

# Low-Thrust Spacecraft Trajectory Optimization with Gravity-Assist Maneuver using Dymos

Gage W. Harris\* and Ping He†  
*Iowa State University, Ames, Iowa, 50011, USA*

Electric propulsion systems are attractive for space missions because they have the advantage of higher specific impulse and payload capability compared with chemical engines. In an interplanetary mission, one often needs to simultaneously consider the spacecraft’s trajectory and onboard system design, which increases the computational cost for design and optimization. This issue is further exacerbated by various combinations of gravity-assist maneuver (fly-bys) options (e.g., the fly-by sequence and the number of fly-bys). Large-scale coupled low-thrust trajectory and system optimization with variable fly-by options have not been well explored. This study will break this limit by proposing a new optimization framework considering low-thrust trajectory optimization with discrete fly-by options. The proposed optimization framework has two main layers: Dymos and dEGO. The Dymos layer uses NASA’s OpenMDAO/Dymos framework for large-scale gradient-based low-thrust trajectory optimization with a given fly-by option. The Dymos layer will output the optimized objective function (e.g., time-of-flight or fuel burn) to the dEGO layer. The dEGO layer then uses discrete, efficient global optimization to update the fly-by option to send it back to the Dymos layer. This paper introduces the framework and showcases our preliminary results for low-thrust trajectory optimization with fly-bys using Dymos. We use the Earth-Venus-Mars case as a demonstration and optimize the fuel consumption by changing the thrust magnitude, direction, and dates for the departure, fly-by, and arrival. The proposed optimization framework has the potential to design more efficient spacecraft missions.

## I. Introduction

As we continue to advance our understanding of the universe, a scheme to optimize large-scale spacecraft missions efficiently is of growing importance. Cost is the most crucial factor when designing a spacecraft mission. Optimizing a spacecraft’s trajectory with discrete fly-bys helps minimize this cost. This, in combination with the design of more efficient electric propulsion systems, makes space missions more feasible. Therefore, increasing research interests in spacecraft optimization have led to an increase in mission efficiency.

Impulsive trajectory optimization with chemical propulsion systems has been a mainstay of space mission design. The goal of these optimizations is to minimize a spacecraft’s energy consumption (often quantified as velocity change;  $\Delta V$ ) or flight time for a mission sequence. Fly-by maneuvers or specific launch windows are used to reduce these objective functions. Genetic algorithm (GA) is a popular approach to optimize spacecraft trajectory and has been used in many studies [1–4]. The standard GA assumes the design variables of a solution as genes in a fixed-length chromosome. By applying the evolutionary operations of selection, mutation, and crossover, the population of these chromosomes converges to the global optimal solution [5]. Differential evolution is another heuristic method that aims to solve the trajectory optimization problem like genetic algorithms. Labroquère et al. [6] utilized a differential evolution technique and implemented four constraint-handling techniques to optimize a multi-gravity assist interplanetary mission to Jupiter. Three of the four techniques could find feasible solutions within a respectable time. In our previous work, we considered coupled spacecraft trajectory and system designs with high-thrust propulsion using gradient-based optimization methods [7, 8].

With the advent of efficient solar electric propulsion systems (SEP), the problem of trajectory optimization has migrated from a domain of finite design variables (impulses) to a more challenging land of infinite continuous design variables. These problems require the thrust magnitude and direction to be given continuously instead of selecting specific instantaneous velocity changes. A benefit of doing this is that there are now more possible trajectories that were otherwise infeasible. However, the object function is littered with local minima and requires numerous continuous-thrust

---

\*PhD Student, Department of Aerospace Engineering, AIAA Student Member.

†Assistant Professor, Department of Aerospace Engineering, AIAA Senior Member. Email: phe@iastate.edu

trajectories to access. There have been many efforts to solve the simplest low-thrust trajectory cases analytically, either by fixing the direction of thrust over the whole trajectory [9–13] or by neglecting boundary conditions. Analytic perturbation theory is another technique for solving low-thrust trajectory optimization problems. Resulting analytic (or semi-analytic) solutions have lower computational costs than explicit numerical techniques and are suitable for moderate accuracies [14, 15].

Optimizing a low thrust trajectory while fixing the mission sequence limits the design freedom of the mission. Existing studies have conducted coupled low-thrust trajectory and system optimization [16–19]. However, they have not included discrete fly-by options in the optimization problems. This study will break this limit by developing a new framework for low-thrust trajectory optimization with discrete fly-by options. The proposed optimization framework has two main layers: Dymos and dEGO. The Dymos layer uses NASA’s OpenMDAO/Dymos framework for large-scale gradient-based low-thrust trajectory optimization with a given fly-by option. The Dymos layer will output the optimized objective function (e.g., time-of-flight or fuel burn) to the dEGO layer. The dEGO layer then uses discrete, efficient global optimization to update the fly-by option to send it back to the Dymos layer. This paper introduces the framework and showcases our preliminary results for low-thrust trajectory optimization with fly-bys using Dymos. We use the Earth-Venus-Mars case as a demonstration and optimize the fuel consumption by changing the thrust magnitude, direction, and dates for the departure, fly-by, and arrival.

The rest of this paper is organized as follows. In Section II, we elaborate on the trajectory optimization framework, followed by the detailed optimization results in Section III. Finally, in Section IV, conclusions are drawn and perspectives and future improvements are provided.

## II. Method

### A. Optimization framework

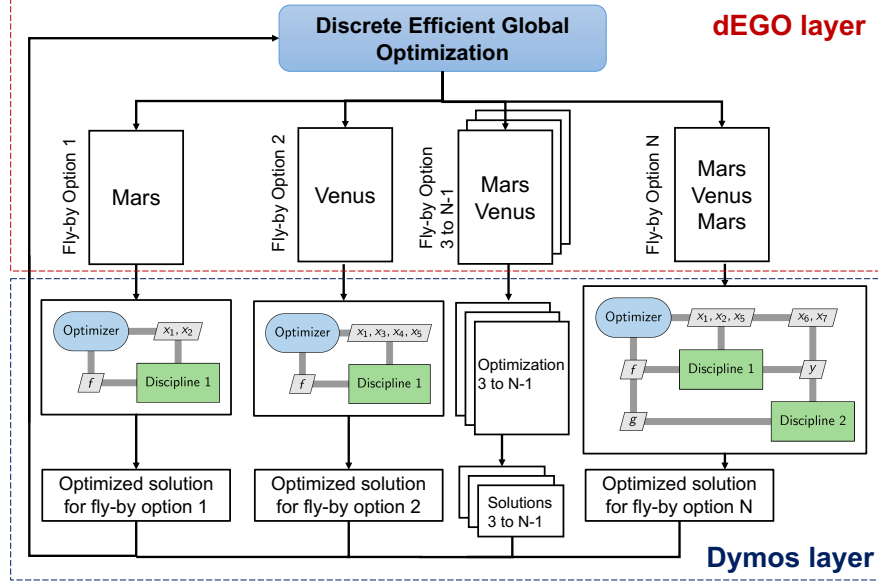
The proposed optimization framework consists of two main layers: dEGO and Dymos, as shown in Fig. 1. The Dymos layer uses NASA’s OpenMDAO/Dymos framework for gradient-based low-thrust trajectory optimization with a given fly-by option, which is the primary focus of this paper. OpenMDAO [20] is an open-source multidisciplinary design, analysis, and optimization framework developed by NASA. OpenMDAO has been widely used for large-scale design optimization in aerospace engineering applications. Dymos [21] is an OpenMDAO derivative for optimizing the control of dynamic systems. Dymos has been mostly used for aircraft MDO [22–24]. This study will extend its application for low-thrust spacecraft trajectory optimization with fly-bys.

Dymos uses the collocation method to solve user-prescribed dynamic equations. Then, it uses a gradient-based optimization algorithm to solve optimization problems. Users can define analytical partial derivative computation in Dymos, which enables adjoint computation of total derivatives for a large number of design variables. This feature is particularly desirable for our problem because the low-thrust trajectory problem requires a large design freedom. Once the optimization problem is solved, the Dymos layer will output the optimized objective function (e.g., time-of-flight or fuel burn) to the future dEGO layer. The dEGO layer then uses discrete, efficient global optimization to update the fly-by option to send it back to the Dymos layer. To be more specific, the dEGO layer will use the optimal objective functions computed from the Dymos layer as the surrogate model output. The surrogate model’s input is the fly-by options (the fly-by sequence and the number of fly-bys). Then, the dEGO layer will use the discrete, efficient global optimization algorithm to compute the next fly-by option to explore. We will give this new fly-by option as the input for the Dymos layer and start the next iteration. We will use the dEGO module implemented in the surrogate model toolbox (SMT) [25].

### B. Open-source Multidisciplinary Analysis Design And Optimization Tool: OpenMDAO

As mentioned above, Dymos is built based on the OpenMDAO framework. OpenMDAO [26] is an open-source computing platform for systems analysis and multidisciplinary optimization written in Python. OpenMDAO allows users to decompose large-scale optimization problems into small components. Each component contains basic computation, making them easier to build and maintain. The benefit of OpenMDAO is its focus on gradient-based optimization with analytical derivatives, allowing one to explore large design spaces with thousands of design variables. This salient feature is achieved using the modular analysis and unified derivatives (MAUD) architecture proposed by Hwang and Martins [27].

As previously stated, OpenMDAO can decompose complex models into smaller disciplines or components. Each



**Fig. 1 Proposed coupled optimization framework with gravity-assist maneuver.**

component is defined by its specific inputs and outputs. In a component, one must implement how to compute the outputs based on the inputs. All OpenMDAO components for our formulation define these input-output relationships with analytical expressions. These include components for the electric propulsion system, trajectory analysis, analytical ephemeris, and fly-by calculations. Lastly, one needs to define how the derivatives of the outputs with respect to the inputs will be calculated for each component. Again, this can be done with analytical expressions as was done for each of the previously stated components. Once all the individual components are set, one can link variables (i.e., inputs and outputs) across components to achieve a coupled model. Then, one can assign any of the variables in the coupled model as the objective function, design variables, and constraints.

### C. Fourier Approximation of States

Generating initial conditions for a low-thrust trajectory is a known challenge because the optimization problem has a large number of design variables. This study uses a so-called Fourier Approximation of States (FFS) method to generate initial conditions. The FFS approach is a shaped-based method for preliminary trajectory design and begins with defining the spacecraft's position as a function of time. Using a polar coordinate system, expressions for the radius  $r$  and the polar angle  $\theta$  are approximated as follows:

$$r(\tau) = \frac{a_0}{2} + \sum_{n=1}^{n_r} a_n \cos(n\pi\tau) + b_n \sin(n\pi\tau), \quad (1)$$

$$\theta(\tau) = \frac{c_0}{2} + \sum_{n=1}^{n_\theta} c_n \cos(n\pi\tau) + d_n \sin(n\pi\tau), \quad (2)$$

where  $0 \leq \tau = t/T \leq 1$  is the scaled time,  $T$  is the corresponding time of each segment, and  $n_r$  and  $n_\theta$  determine the number of Fourier terms. Using more Fourier terms improves accuracy but costs more design variables and, thus, computational time. Since we are constraining the escape phase as a rendezvous problem, the corresponding Fourier series approximations for  $r$  and  $\theta$  are constrained to satisfy eight boundary conditions:

$$\begin{aligned} r_i &= r(\tau = 0) & r_f &= r(\tau = 1) \\ r'_i &= r'(\tau = 0) & r'_f &= r'(\tau = 1) \\ \theta_i &= \theta(\tau = 0) & \theta_f &= \theta(\tau = 1) \\ \theta'_i &= \theta'(\tau = 0) & \theta'_f &= \theta'(\tau = 1) \end{aligned} \quad (3)$$

where the prime denotes derivative with respect to the scaled time  $\tau$ , and the subscripts "i" and "f" refer to the initial and final time, respectively.

With eight boundary conditions known, eight unknown Fourier coefficients can be computed. Therefore, the number of design variables left for the non-linear program solver (NLP) is known by:  $2(n_r + n_\theta - 6)$ . The design variables must also be initialized, which can be done with a simple spiral approximation, such as a cubic polynomial. The Fourier series must then be mapped to the approximation. Taheri and Abdelkhalik [28] outline this process in detail, with the general concept being to evaluate the approximated spiral at many discrete points and then curve fitting to the finite Fourier expansions. Satisfaction of the EoM at discrete points is satisfied by minimizing the squared residual errors through the objective function. Additional constraints and penalty terms can be added to the non-linear programming problem as necessary.

#### D. Optimal dynamic control tool: Dymos

As mentioned above, Dymos is a library for the optimal control of dynamic multidisciplinary systems [21]. While Dymos can optimize typical optimal control problems, its key feature is the ability to optimize systems in which a trajectory is only one part of the overall optimization. To do this, Dymos relies on the ability of OpenMDAO to compute accurate derivatives.

Solving optimal control problems using classical techniques is typically a matter of finding a continuous control that moves a system to a desired state while extremizing some objective. To solve these problems with non-linear optimizers, the continuous problem needs to be discretized. This process is called transcription. Dymos supports two forms of direct transcription: collocation and explicit shooting. Collocation was chosen to avoid the numerical integration of explicit shooting.

Collocation-based optimal control methods are implicit techniques. The entire state and control history is proposed at some discrete points in time. One can then calculate how accurately the proposed state and control history obey the governing system dynamics - this is a residual called the defects. By varying the states, controls, and time the defects can be reduced to zero, meaning the state trajectory is a solution to the ODE of the system.

In this work, Dymos will be used to simulate and optimize the trajectory of a continuous thrust spacecraft. To do this, two explicit OpenMDAO components are defined, containing the differential equations for a two-body planar formulation. The dynamics between the two components will be identical; the only difference will be the boundary condition constraints. Inputs and outputs for the two Dymos components can be seen in Table 1. One component will be used for rendezvous, i.e., both position and velocity terminal constraints are fixed. While the second component will be used for an intercept, i.e., the terminal position is fixed, but the terminal velocity is free. The rendezvous component will be used in the final leg of the mission to arrive at the determined destination, while the intercept component will enable fly-bys. Specific fly-by maneuvers are calculated using the planar formulation shown by Curtis [29], which uses the spacecraft velocity, fly-by planet velocity, and a fly-by altitude to return the heliocentric velocity components of the spacecraft after the maneuver. Equations implemented for both Dymos components were taken from Enright et al. [30], which use polar coordinates with canonical units for the Earth-Sun system.

$$\dot{r} = v_r \quad (4)$$

$$\dot{\theta} = \frac{v_t}{r} \quad (5)$$

$$\dot{v}_r = \frac{v_t^2}{r} - \frac{1}{r^2} + \frac{F}{m} \sin u \quad (6)$$

$$\dot{v}_\theta = -\frac{v_t v_r}{r} + \frac{F}{m} \cos u \quad (7)$$

$$\dot{m} = -\frac{F}{c} \quad (8)$$

where  $u$  is the thrust steering angle.

**Table 1 The inputs and outputs for the Dymos trajectory component.**

| Function/Variable   |                      | Description                                                  | Quantity |
|---------------------|----------------------|--------------------------------------------------------------|----------|
| Inputs              | $r$                  | Radius from center of attraction (DU)                        | 1        |
|                     | $\theta$             | Polar angle (rad)                                            | 1        |
|                     | $v_r$                | Radial velocity component (DU / TU)                          | 1        |
|                     | $v_\theta$           | Polar velocity component (DU / TU)                           | 1        |
|                     | $F$                  | Thrust magnitude (DU * kg / TU <sup>2</sup> )                | 1        |
|                     | $m$                  | Total mass of spacecraft (kg)                                | 1        |
|                     | $u_1$                | Thrust angle (rad)                                           | 1        |
|                     | $I_{SP}$             | Specific impulse (TU)                                        | 1        |
| <b>Total Inputs</b> |                      |                                                              | <b>8</b> |
| Outputs             | $\dot{r}$            | rate of change of radius from center of attraction (DU / TU) | 1        |
|                     | $\dot{\theta}$       | rate of change of polar angle (rad / TU)                     | 1        |
|                     | $\dot{v}_r$          | rate of change of radial velocity (DU / TU <sup>2</sup> )    | 1        |
|                     | $\dot{v}_\theta$     | rate of change of polar velocity (DU / TU <sup>2</sup> )     | 1        |
|                     | $\dot{m}$            | rate of change of spacecraft mass (kg / TU)                  | 1        |
|                     | <b>Total Outputs</b> |                                                              | <b>5</b> |

**E. Analytical Ephemeris**

To obtain the position and velocity boundary constraints of the Dymos model, an OpenMDAO component was created containing an analytical ephemeris model. An analytical model was chosen because of the computational efficiency and the ability to utilize the analytic gradients for gradient-based optimization. The model is a combination of Meeus [31] and Standish et al. [32] and initially computes the position and velocity of a planet in the J2000 ecliptic frame. Tables of coefficients are provided by [32] to obtain the updated Keplerian elements for a specific planet. The elements given in the tables are:

$$\begin{aligned}
a_0, \dot{a} &: \text{sem-major axis [au, au/century]} \\
e_0, \dot{e}_0 &: \text{eccentricity [radians, radians/century]} \\
I_0, \dot{I}_0 &: \text{inclination [degrees, degrees/century]} \\
L_0, \dot{L}_0 &: \text{mean longitude [degrees, degrees/century]} \\
\bar{\omega}_0, \dot{\bar{\omega}} &: \text{longitude of perihelion [degrees, degrees/century]} \\
\Omega_0, \dot{\Omega} &: \text{longitude of the ascending node [degrees, degrees/century]}
\end{aligned}$$

**Table 2 The inputs and outputs for the analytical ephemeris component.**

| Function/Variable |                      | Description                           | Quantity |
|-------------------|----------------------|---------------------------------------|----------|
| Inputs            | JD                   | Julian Date                           | 1        |
|                   | Index                | Planet Index                          | 1        |
|                   | <b>Total Inputs</b>  |                                       | <b>2</b> |
| Outputs           | $r$                  | Radius from center of attraction (DU) | 1        |
|                   | $\theta$             | Polar angle (rad)                     | 1        |
|                   | $v_r$                | Radial velocity component (DU / TU)   | 1        |
|                   | $v_\theta$           | Polar velocity component (DU / TU)    | 1        |
|                   | <b>Total Outputs</b> |                                       | <b>4</b> |

To obtain the coordinates of one of the planets at a given Julian Ephemeris Date,  $T_{eph}$ , we will

- 1) Compute the value of each orbital element:

$$a = a_0 + \dot{a}T, etc \quad (9)$$

where T, the number of centuries past J2000.0, is  $T = (T_{eph} - 2451545.0)/36525$ .

- 2) Compute the argument of perihelion,  $\omega$  and the mean anomaly, M:

$$\omega = \bar{\omega} - \Omega \quad (10)$$

$$M = L - \bar{\omega} + bT^2 + c \cos fT + s \sin fT \quad (11)$$

where the last three terms must be added for Jupiter through Pluto, and are provided in the tables for each respective planet.

- 3) The true anomaly can be found as:

$$\theta = M + C_{cen} \quad (12)$$

where

$$C_{cen} = \left(2e - \frac{e^3}{4} + \frac{5}{96}e^5\right) \sin(M) + \left(\frac{5}{4}e^2 - \frac{11}{24}e^4\right) \sin(2M) + \left(\frac{13}{12}e^3 - \frac{43}{64}e^5\right) \sin(3M) + \frac{103}{96}e^4 \sin(4M) + \frac{1097}{960}e^5 \sin(5M) \quad (13)$$

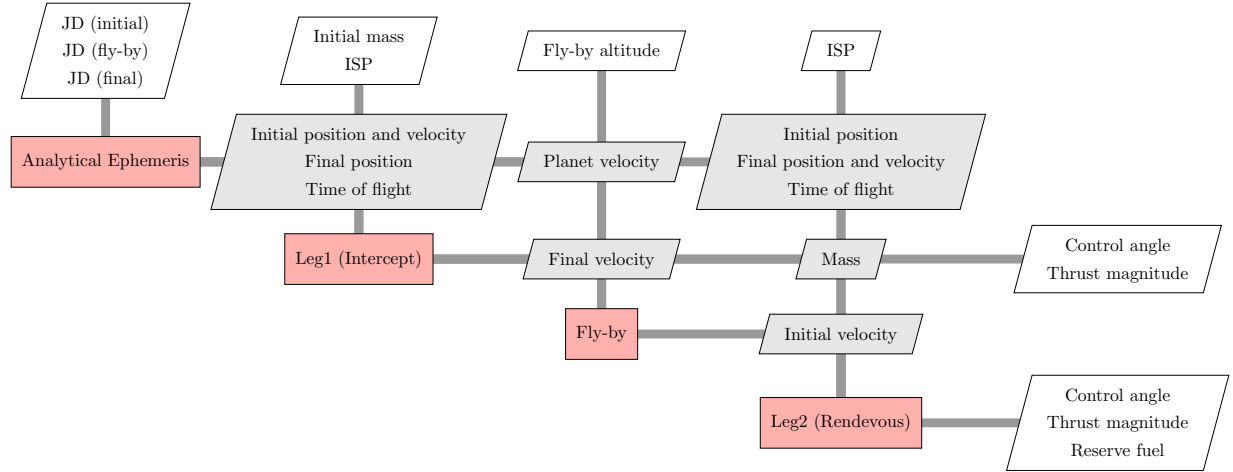
Inputs and outputs for the analytical ephemeris model can be seen in Table 2. All of the internal computations are not shown here because once the orbital elements ( $a, e, i, \Omega, \omega, \theta$ ) are known, the position and velocity of the planet can be calculated traditionally. This position and/or velocity will be thrown into the Dymos component as boundary conditions for the rendezvous or intercept segment.

## F. Spacecraft Mission Optimization Tool: SMOT

Space mission optimization tool (SMOT) is a Python interface we are developing to combine Dymos, analytical ephemeris, fly-by calculations, and OpenMDAO for design optimization of spacecraft trajectory with discrete fly-bys. OpenMDAO acts as the top-level driver and is where all the various components of the mission will be defined (e.g., spacecraft trajectory and fly-by model). Dymos is the trajectory optimization tool, while the analytical ephemeris and fly-by calculations are implemented as OpenMDAO components.

Before the optimization can begin, initial conditions must be generated to be thrown into the optimizer. As mentioned above, the FFS method is our way of generating these initial conditions. The FFS method uses a polar coordinate system, which makes the connection to our dymos component straightforward. A specific discrete trajectory must be chosen for the FFS method to map a trajectory. The specific Earth-Venus-Mars (EVM) case used in this study was found using a traditional genetic algorithm (GA). The goal of the GA was to find the most efficient impulsive Earth-Mars (EM) mission with certain departure/arrival constraints. The impulsive mission solution was converted to a continuous thrust mission using the abovementioned FFS method. With initial guesses generated, the optimization of the continuous thrust trajectory could be run.

As mentioned, there are two separate Dymos components: a rendezvous segment and an intercept segment. All boundary conditions for a specific segment are known because the departure date, time of flights, and arrival date are design variables, which can then be turned into position and velocity vectors with the analytical ephemeris model. SMOT will use both of these segments to complete a spacecraft trajectory. For example, the mission used in this study is an interplanetary mission to Mars from Earth with a planetary fly-by around Venus. To formulate this mission, SMOT will first call the intercept component with initial conditions (position and velocity) fixed to Earth at a departure date ( $t_d$ ) and the final position fixed to Venus at a given time of flight ( $t_1$ ). The solution of that segment will give us the correct position, with a certain  $v_\infty$ , to model a fly-by as an instantaneous boost. Next, SMOT will call the rendezvous segment with the initial position fixed to Venus at ( $t_1$ ) and the initial velocity chosen by the output of the fly-by component, with



**Fig. 2 SMOT framework with the Dymos segment, illustrated using the extended design structure matrix (XDSM) diagram [33].**

the terminal position and velocity fixed to Mars at the arrival date ( $t_a$ ). The rendezvous segment's solution signals the simulation's completion for a specific optimization iteration. This process will continue until the optimizer converges to a solution.

A basic overview of the coupled model can be seen in Fig. 2. The diagonal blocks are the components, and the off-diagonal blocks denote the data transfer between components. A component receives the inputs from the vertical direction and outputs data in the horizontal direction. The gray blocks in Fig. 2 denote the data transfer between the four components. To be more specific, the analytical ephemeris uses each planet's arrival/departure dates as inputs and outputs the necessary boundary conditions for each trajectory component. The trajectory component then uses the control parameters (e.g., control angle and thrust magnitude) to meet these boundary conditions. There is no feedback from the trajectory components to the analytical ephemeris model; the components are one-way coupled.

In the future, we plan to add an upper-level discrete EGO module to choose the discrete fly-by sequences rather than having a predetermined mission selected. SMOT will then call the corresponding Dymos components to execute the sequence given by the dEGO. The solution to each sequence will be similar to what was described above, each ending with a rendezvous segment with  $n$  fly-by and intercept components.

### III. Results and Discussion

Before the problem was put together, each component was tested individually for accuracy. Every component mentioned above was created using the OpenMDAO framework, and implemented analytic gradients. When dealing with user-defined analytic gradients, testing against a numerical method is necessary. OpenMDAO has an easy way to do this test with the check-partials function. This function takes the output equations defined in a component and numerically computes the input-output gradients. Next, the function will compute the same input-output gradients using the user-defined analytic gradients and compare the relative difference between the two. The complex step was chosen for numerical computation due to its improved accuracy over the finite difference. This method tested each component, totaling over 250 unique analytic gradients. Through testing, the largest relative difference between a single gradient was  $10^{-14}$ , which confirmed the correct implementation of all analytic gradients.

Most component output equations did not need to be tested because they were not new formulations. However, the analytical ephemeris model combined two different models, and there was no guarantee that it was accurate. A loop was created to call the model every day from March 2022 through March 2023 to test the analytical ephemeris model. This was done for each planet in the solar system. For comparison, MATLAB's numerical planet-Ephemeris function was also run in the same loop. MATLAB was assumed to produce the correct ephemeris of a planet, so a percent difference was then calculated for each position and velocity term between the two models. Table 3 shows the results of this comparison and presents the average percent difference of the analytical model for three different planets,

**Table 4 Optimization formulation for the trajectory optimization with fly-bys.**

|            | Function/Variable             | Description                                   | Quantity  |
|------------|-------------------------------|-----------------------------------------------|-----------|
| Minimize   | $m_i$                         | Initial spacecraft mass (kg)                  | 1         |
| w.r.t      | $JD_{initial}$                | Earth departure date (day)                    | 1         |
|            | $JD_{fly-by}$                 | Venus fly-by date (day)                       | 1         |
|            | $JD_{final}$                  | Mars arrival date (day)                       | 1         |
|            | $r_p$                         | Fly-by altitude (km)                          | 1         |
|            | $u_1$                         | Steering angle (rad)                          | 42        |
|            | F                             | Thrust magnitude (DU * kg / TU <sup>2</sup> ) | 42        |
|            | <b>Total Design Variables</b> |                                               | <b>88</b> |
| Subject to | $\Delta r$                    | Radial position boundary constraints          | 6         |
|            | $\Delta \theta$               | Polar position boundary constraints           | 6         |
|            | $\Delta v_r$                  | Radial velocity boundary constraints          | 4         |
|            | $\Delta v_\theta$             | Polar velocity boundary constraints           | 4         |
|            | $m_f \geq 625$                | Forced final mass (kg)                        | 1         |
|            | <b>Total Constraints</b>      |                                               | <b>21</b> |

as well as the execution time for a single run of a specific model.

|                                      | Analytical ephemeris | MATLAB numerical |
|--------------------------------------|----------------------|------------------|
| Execution Time (s)                   | 0.0025               | 0.1562           |
| Average Percent Difference (Earth)   | 0.6%                 | NA               |
| Average Percent Difference (Mars)    | 0.1%                 | NA               |
| Average Percent Difference (Jupiter) | 0.35%                | NA               |

**Table 3 Analytical Ephemeris Comparison**

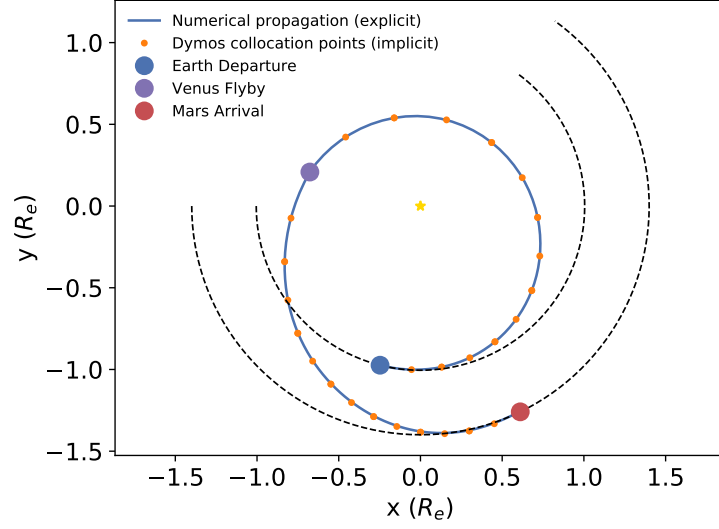
It can be seen from the table that the analytical ephemeris model matches closely with MATLAB's numerical model and gives us confidence in the accuracy of the formulation. In addition, the analytical model is two orders of magnitude faster than the numerical model, which will help reduce computational costs. This, coupled with the analytical derivatives of the ephemeris model, will greatly increase the efficiency of the gradient-based optimization.

Once each component was tested, putting together the formulation was straightforward. As described in the method section, a pre-defined EVM case was used. With only two legs needed in the trajectory, one intercept, fly-by, and rendezvous component were defined. The analytical ephemeris model for each component also connected all necessary boundary conditions. All connections and creation of components were done automatically through the creation of external functions. The reason for doing this automatically comes in the future when the dEGO layer will choose the trajectory sequence and create the runscrip itself rather than using a pre-defined sequence.

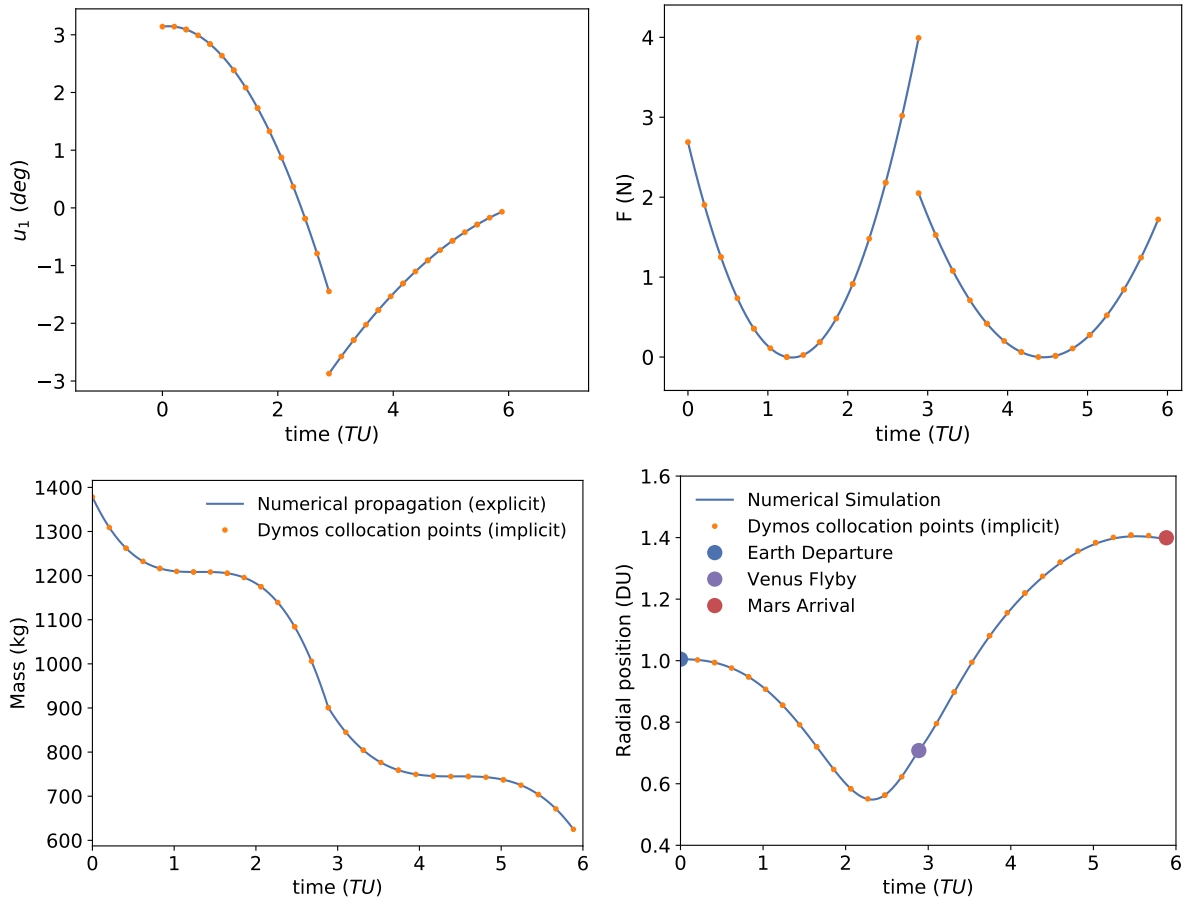
Table 4 shows the optimization formulation. The optimization goal was to reach Mars with the lowest fuel burn possible, subject to the boundary conditions given by the analytical ephemeris model for the intercept and rendezvous segments. In addition, the final fuel mass had to be more than some reserve amount chosen to be 50kg for this study. The dry mass and  $I_{sp}$  of the spacecraft were chosen to be 575kg and 3500s, respectively, which was a combination of the DAWN [34] and DART [35] spacecrafts. The optimizations were run using the Sparse Nonlinear OPTimizer (SNOPT) package [36] and converged in less than ten minutes on a Dell workstation.

Figure 3 shows the final  $x - y$  trajectory plot of the transfer in canonical units when the departure and initial dates are fixed. These dates were initially fixed to help the optimization converge faster. The goal here was to test the capability

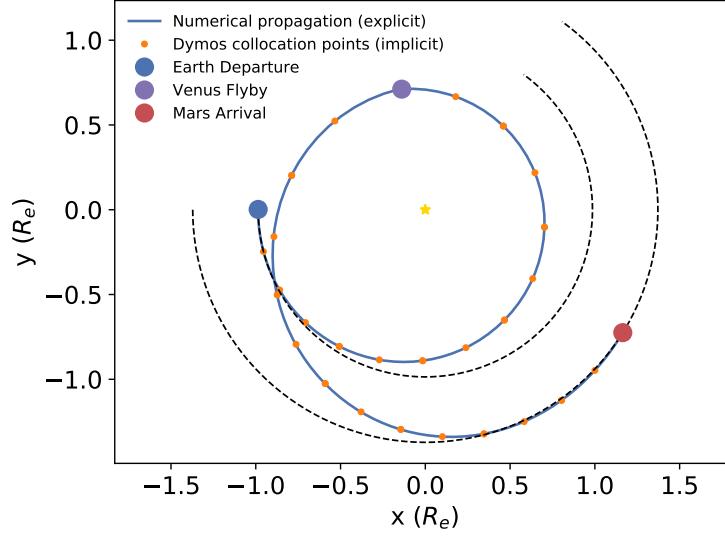




**Fig. 3**  $x - y$  trajectory plot of the optimized transfer plotted in the polar canonical Sun ecliptic frame with fixed departure/arrival dates.



**Fig. 4** Plots showing how the steering angle, thrust magnitude, spacecraft mass, and radial distance change over the trajectory with fixed departure/arrival dates. Time is plotted in the Earth-Sun canonical time units.



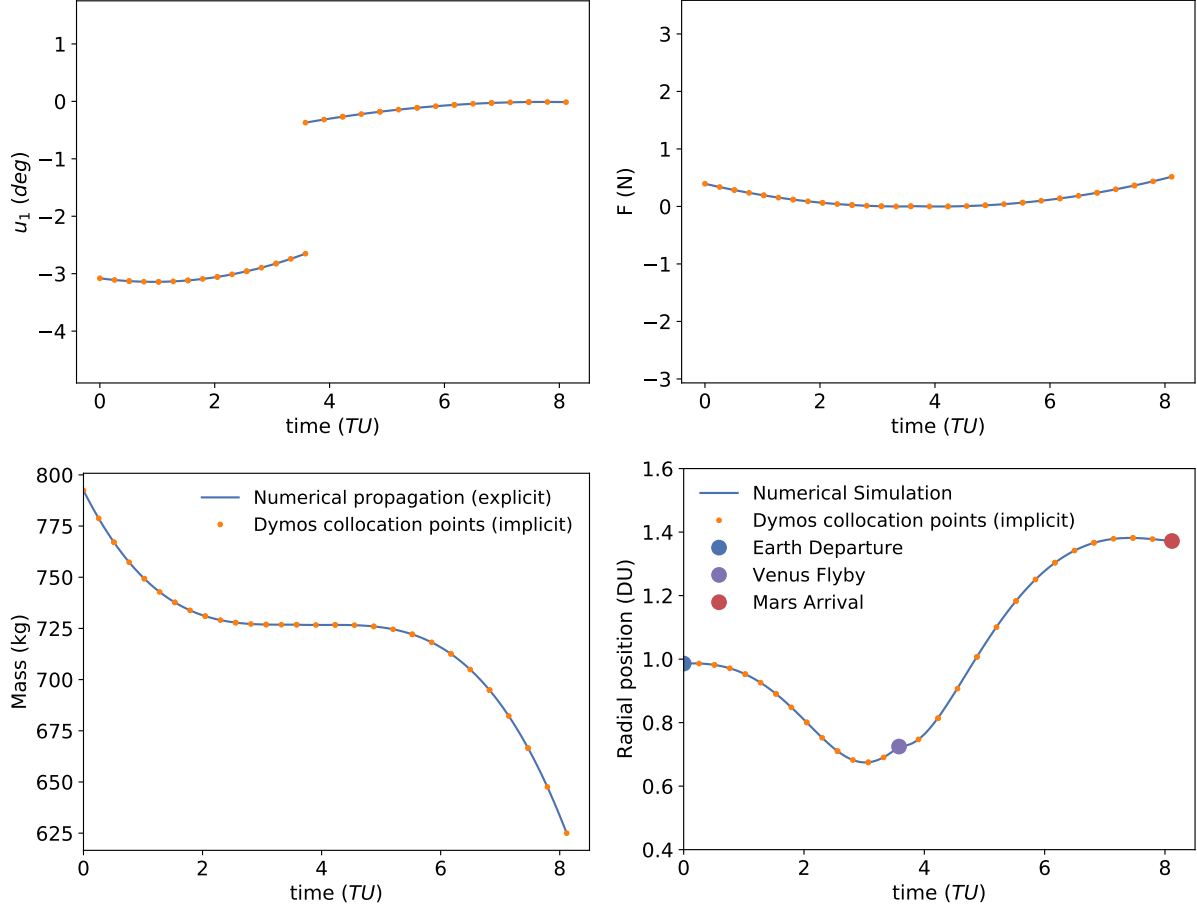
**Fig. 5** XY trajectory plot of the optimized transfer plotted in the polar canonical Sun ecliptic frame with variable departure/arrival dates.

of the Dymos layer. The implicit collocation solution by Dymos (yellow dots) and the numerical propagation simulation (blue curves) agree at every point, showing the agreement of the collocation method to the equations of motion. All boundary conditions are also met, meaning the rendezvous was completed. The spacecraft leaves Earth on June 6<sup>th</sup>, 2004 and arrives at Venus on November 20<sup>th</sup>, 2004. Fly-bys are modeled as instantaneous impulsive maneuvers, so the spacecraft leaves Venus on the same day it arrives while rendezvousing with Mars on May 5<sup>th</sup>, 2005. Before the fly-by, the spacecraft's radial and polar velocities were 12.84 and 40.47 km/s, respectively, while after the maneuver, they became 10.27 and 44.55 km/s.

Figure 4 shows how the control (steering angle), thrust magnitude, cost (spacecraft mass), and radial distance change over the final trajectory. Discontinuities in the control angle and thrust plots are not a worry because this is when the fly-by is executed. The steering angle and thrust magnitude are changed throughout the optimization to reduce the initial spacecraft mass, subject to a final mass constraint. The optimization iterations are not shown here, but the cost does decline as the optimization iterations increase, with a final fuel burn value of about 750kg. The optimality and feasibility are also very small, meaning a local minimum has been found. One might notice that the thrust magnitudes shown in Fig. 4 are much higher than what normal electric propulsion systems can produce. For example, the NEXT-C ion thruster equipped to the DART spacecraft had a max thrust of about .24 N [35]. Excessive thrusts arise because of the fixed departure and arrival dates. The initial flight times between each leg are too small and thus require the continuous thrust propulsion system to exhibit unreasonable thrust magnitudes for current designs. This result is expected because a direct conversion from an impulsive to a continuous trajectory seems far-fetched. The remaining control variables are reasonable, but an infeasible propulsion system is a major problem. In an effort to reduce these thrust magnitudes, the departure and arrival dates were added as design variables, and the optimization results are shown in the following.

Figure 5 shows the final  $x - y$  trajectory plot of the transfer in canonical units when the departure and initial dates are variable. Again, the implicit collocation solution and numerical propagation simulation agree at every point, and all boundary conditions are met. It can be seen from Fig. 5 that the overall structure of the trajectory is very similar to that of Fig. 3. The main difference is that the trajectory in Fig. 5 is elongated. Letting the dates be design variables allows the optimization to push up the Earth departure date and push back the Mars arrival dates while keeping the Venus fly-by date about the same. Doing this stretches out both legs of the trajectory. Now, the spacecraft leaves Earth on March 20<sup>th</sup>, 2004, and arrives at Venus on October 14<sup>th</sup>, 2004. The fly-by is still modeled as instantaneous impulsive maneuvers, so the spacecraft leaves Venus on the same day it arrives while rendezvousing with Mars on July 4<sup>th</sup>, 2005. Before the fly-by, the spacecraft's radial and polar velocities were 4.773 and 36.44 km/s, respectively, while after the maneuver, they became  $-0.5856$  and 40.08 km/s.

By stretching out both legs, the electric propulsion system uses much smaller thrust magnitudes, as shown in Fig. 6. Now, the thrust magnitudes are much more reasonable, always staying under about .5N. The discontinuities in the



**Fig. 6** Plots showing how the steering angle, thrust magnitude, spacecraft mass, and radial distance change over the trajectory with variable departure/arrival dates. Time is plotted in the Earth-Sun canonical time units.

control angle still show up, but again, this is when the fly-by is being executed. The elongated trajectory not only solved the infeasible thrust magnitudes but also vastly improved the cost function. Now, instead of the mission using 750kg of fuel, it only uses about 160kg. This is nearly a 80% fuel burn reduction, showing the importance of having these dates as design variables. The mission duration is increased from 11 months to 16 months by doing this, but the fuel burn reduction is well worth the wait.

#### IV. Summary

This paper presents a low-thrust trajectory optimization framework that can consider gravity-assist maneuver options. The optimization framework shown above is the Dymos layer, the first step in the proposed framework. The Dymos layer combines Dymos, analytical ephemeris, fly-by calculations, and OpenMDAO to optimize a polar trajectory with discrete fly-bys. Initial conditions are derived from a Fourier approximation, which decomposes a discrete impulsive trajectory into a continuous thrust mission. The goal of the optimization was to reach Mars with the lowest fuel burn possible.

Initially, the departure and arrival dates of the selected EVM mission were fixed to help the optimizer converge and demonstrate the feasibility of the Dymos layer. An optimized solution was found, but thrust magnitudes were infeasible regarding current electric propulsion system designs. To alleviate this issue, the departure and arrival dates of the mission were used as design variables, along with the control angles and thrust magnitudes. Doing this reduced the fuel burn by around 80% and solved the infeasible thrust magnitudes. With a fully feasible solution, we can confirm the feasibility of this Dymos layer. It is possible to optimize a continuous thrust trajectory with discrete fly-bys using this method.

In the future, we will further improve the shown framework by adding an upper-level discrete EGO module to choose

the discrete fly-by sequences rather than selecting a predetermined mission. SMOT will then call the corresponding Dymos components to execute the sequence given by the dEGO. The solution to each sequence will be similar to what was described above, each ending with a rendezvous segment with n fly-by and intercept components. This new feature will allow us to use discrete design variables (e.g., departure dates, fly-bys, and deep space maneuvers) and have a better chance of finding the global optimal solution.

### Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Number 2223676. We would like to thank Eliot Aretskin-hariton and Robert Falck for their insightful comments on trajectory optimization and Dymos setup. In addition, we thank Ossama Abdelkhalik's group for sharing the FFS code and the impulsive trajectory solution of the Earth-Venus-Mars case.

### References

- [1] Darani, S. A., and Abdelkhalik, O., "Space trajectory optimization using hidden genes genetic algorithms," *Journal of Spacecraft and Rockets*, Vol. 55, No. 3, 2018, pp. 764–774. <https://doi.org/10.2514/1.A33994>.
- [2] Gad, A., and Abdelkhalik, O., "Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization," *Journal of Spacecraft and Rockets*, Vol. 48, No. 4, 2011, pp. 629–641. <https://doi.org/10.2514/1.52642>.
- [3] Izzo, D., Hennes, D., and Riccardi, A., "Constraint handling and multi-objective methods for the evolution of interplanetary trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 4, 2015, pp. 792–800. <https://doi.org/10.2514/1.G000619>.
- [4] Rauwolf, G. A., and Coverstone-Carroll, V. L., "Near-optimal low-thrust orbit transfers generated by a genetic algorithm," *Journal of Spacecraft and Rockets*, Vol. 33, No. 6, 1996, pp. 859–862. <https://doi.org/10.2514/3.26850>.
- [5] Golberg, D. E., "Genetic algorithms in search, optimization, and machine learning," *Addison wesley*, Vol. 1989, No. 102, 1989, p. 36.
- [6] Labroquère, J., Héritier, A., Riccardi, A., and Izzo, D., "Evolutionary Constrained Optimization for a Jupiter Capture," *International Conference on Parallel Problem Solving from Nature*, Springer, 2014, pp. 262–271. [https://doi.org/10.1007/978-3-319-10762-2\\_26](https://doi.org/10.1007/978-3-319-10762-2_26).
- [7] Harris, G. W., He, P., and Abdelkhalike, O. O., "A coupled spacecraft system and trajectory optimization framework using GMAT and OpenMDAO," *AAS/AIAA astrodynamics specialist conference*, 2022.
- [8] Harris, G., He, P., and Abdelkhalik, O., "Control co-design optimization of spacecraft trajectory and system for interplanetary missions," *Journal of Spacecraft and Rockets*, 2023, p. (under review).
- [9] Tsien, H., "Take-off from satellite orbit," *Journal of the American Rocket Society*, Vol. 23, No. 4, 1953, pp. 233–236.
- [10] Benney, D., "Escape from a circular orbit using tangential thrust," *Journal of Jet Propulsion*, Vol. 28, No. 3, 1958, pp. 167–169.
- [11] Lawden, D., "Optimal escape from a circular orbit," *Astronautica Acta*, Vol. 4, No. 3, 1958, pp. 218–233.
- [12] Schwenzfeger, K. J., "Asymptotic solution to the tangential low thrust energy increase trajectory," Tech. rep., 1973.
- [13] Boltz, F. W., "Orbital motion under continuous radial thrust," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 3, 1991, pp. 667–670.
- [14] Afshari, H., and Taheri, E., "Non-linear solution to the maximum height orbit transfer guidance problem," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, Vol. 226, No. 6, 2012, pp. 620–630.
- [15] Zuiani, F., Vasile, M., Palmas, A., and Avanzini, G., "Direct transcription of low-thrust trajectories with finite trajectory elements," *Acta Astronautica*, Vol. 72, 2012, pp. 108–120.
- [16] Kluever, C. A., and Pierson, B. L., "Vehicle-and-trajectory optimization of nuclear electric spacecraft for lunar missions," *Journal of Spacecraft and Rockets*, Vol. 32, No. 1, 1995, pp. 126–132.

- [17] Koppel, R. C., "Optimal specific impulse of electric propulsion," *Second european spacecraft propulsion conference, proceedings of the conference held 27-29 may 1997 in noordwijk, the netherlands. Edited by michael perry. ESA SP-398. Paris: European space agency, 1997., p. 131*, Vol. 398, 1997, p. 131.
- [18] Arya, V., Taheri, E., and Junkins, J. L., "A composite framework for co-optimization of spacecraft trajectory and propulsion system," *Acta Astronautica*, Vol. 178, 2021, pp. 773–782. <https://doi.org/10.1016/j.actaastro.2020.10.007>, publisher: Elsevier.
- [19] Shimane, Y., Preston, D., and Ho, K., "Concurrent optimization of gravity-assist low-thrust trajectory with power and propulsion subsystem sizing," *2022 AAS/AIAA Astrodynamics Specialist Conference*, 2022.
- [20] Gray, J. S., Hwang, J. T., Martins, J. R. R. A., Moore, K. T., and Naylor, B. A., "OpenMDAO : An open-source framework for multidisciplinary design , analysis , and optimization," *Structural and Multidisciplinary Optimization*, 2019.
- [21] Falck, R., Gray, J. S., Ponnappalli, K., and Wright, T., "dymos: A Python package for optimal control of multidisciplinary systems," *Journal of Open Source Software*, Vol. 6, No. 59, 2021, p. 2809.
- [22] Falck, R. D., Chin, J., Schnulo, S. L., Burt, J. M., and Gray, J. S., "Trajectory optimization of electric aircraft subject to subsystem thermal constraints," *18th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2017, p. 4002. <https://doi.org/10.2514/6.2017-4002>.
- [23] Hendricks, E. S., Falck, R. D., and Gray, J. S., "Simultaneous propulsion system and trajectory optimization," *18th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2017, p. 4435.
- [24] Falck, R. D., Ingraham, D., and Aretskin-Hariton, E., "Multidisciplinary optimization of urban-air-mobility class aircraft trajectories with acoustic constraints," *2018 AIAA/IEEE Electric Aircraft Technologies Symposium (EATS)*, IEEE, 2018, pp. 1–7.
- [25] Bouhlel, M. A., Hwang, J. T., Bartoli, N., Lafage, R., Morlier, J., and Martins, J. R. R. A., "A Python surrogate modeling framework with derivatives," *Advances in Engineering Software*, 2019, p. 102662. <https://doi.org/https://doi.org/10.1016/j.advengsoft.2019.03.005>.
- [26] Gray, J. S., Hwang, J. T., Martins, J. R. R. A., Moore, K. T., and Naylor, B. A., "OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization," *Structural and Multidisciplinary Optimization*, Vol. 59, No. 4, 2019, pp. 1075–1104. <https://doi.org/10.1007/s00158-019-02211-z>.
- [27] Hwang, J. T., and Martins, J. R., "A computational architecture for coupling heterogeneous numerical models and computing coupled derivatives," *ACM Transactions on Mathematical Software (TOMS)*, Vol. 44, No. 4, 2018, pp. 1–39.
- [28] Taheri, E., and Abdelkhalik, O., "Shape based approximation of constrained low-thrust space trajectories using Fourier series," *Journal of Spacecraft and Rockets*, Vol. 49, No. 3, 2012, pp. 535–546.
- [29] Curtis, H. D., *Orbital mechanics for engineering students*, Butterworth-Heinemann, 2013.
- [30] Enright, P. J., and Conway, B. A., "Optimal finite-thrust spacecraft trajectories using collocation and nonlinear programming," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 5, 1991, pp. 981–985.
- [31] Meeus, J., "Astronomical algorithms," *Richmond*, 1991.
- [32] Standish, E. M., Williams, J. G., et al., "Orbital ephemerides of the Sun, Moon, and planets," *Explanatory supplement to the astronomical almanac*, 1992, pp. 279–323.
- [33] Lambe, A. B., and Martins, J. R. R. A., "Extensions to the Design Structure Matrix for the Description of Multidisciplinary Design, Analysis, and Optimization Processes," *Structural and Multidisciplinary Optimization*, Vol. 46, 2012, pp. 273–284. <https://doi.org/10.1007/s00158-012-0763-y>.
- [34] E. Bell, I., "Dawn," , 2022. URL <https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=2007-043A>.
- [35] E. Bell, I., "Double Asteroid Redirection Test," , 2022. URL <https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=2021-110A>.
- [36] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM review*, Vol. 47, No. 1, 2005, pp. 99–131. <https://doi.org/10.1137/S1052623499350013>.