UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

Using genetic algorithm as a multi-gravity assist trajectory design tool

TFM TITLE: Using genetic algorithm optimization as a multi-gravity assist trajectory design tool

DEGREE: Master of Aerospace Science and Technology

AUTHOR: Daniel Huterer Prats

ADVISOR: Sebastian Andreas Altmeyer

DATE: July 9, 2023

Title : Using genetic algorithm optimization as a multi-gravity assist trajectory design tool
Author: Daniel Huterer Prats

Advisor: Sebastian Andreas Altmeyer

Date: July 9, 2023

Overview

This master thesis presents the development of a genetic algorithm for optimizing interplanetary trajectories using multi-planetary gravity assists while considering time as an additional objective in the fitness evaluation. The objective of the research is to address the challenges of designing efficient trajectories that minimize both delta-v and travel duration. The aim of this thesis is to develop an all-encompassing deep space mission trajectory design tool where a trade-off between the total delta-v used and the arrival time can be made, in the interest of the overall mission profile. The results obtained highlight the effectiveness of genetic algorithms in finding optimal multi-planetary gravity assist trajectories and contribute to the advancement of trajectory optimization techniques for future space missions.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

In the realm of modern space exploration, where missions like BepiColombo[6], New Horizons[7], and Juno[8] continue to push the boundaries, one crucial challenge remains constant: designing the spacecraft's journey to its destination. Gravity assist maneuvers, integral to interplanetary travel, have been employed extensively to optimize trajectories and minimize fuel consumption. These maneuvers involve close approaches to celestial bodies, altering the spacecraft's speed and direction through gravitational interactions. A comprehensive exploration of gravity assist maneuvers will be delved into in the following section.

These maneuvers serve not only to propel spacecraft to distant bodies with prohibitively high delta-v requirements but also to decelerate them for rendezvous or injection orbits. Even the illustrious Voyager-I mission relied on gravity assists to reach its ultimate target, Saturn, due to the limitations of its launch vehicle. Voyager-I's incredible feat of escaping the solar system entirely stands as a testament to the power of this technique[9].

While the mechanics of gravity assist maneuvers are relatively understood, the mission design complexity lies in determining the optimal trajectory, especially when employing multiple gravity assists (MGA). Finding a viable interplanetary path can be achieved with effort, but attaining the most fuel efficient trajectory—the one with the least overall delta-v requirement—is a challenge. Delta-v, the primary factor driving mission cost and spacecraft complexity, becomes the cornerstone of trajectory design. In fact, a significant portion of the work presented here is built on top of the code developed by Iker Diaz Cilleruelo (UPC alumni), whose work was considering only the delta-V as the single optimizing parameter [3]. Minimizing delta-v translates to smaller launchers and lighter spacecraft, directly influencing other subsystems. However, the launcher's inherent limitations impose a cap on the available delta-v, necessitating careful optimization with other objectives in mind. Therefore, in this work time, in the form of the arrival time, is added into consideration as it is a crucial variable in mission design including deep space missions to the outer solar system. As deep space missions may travel for over a decade to their final destination as for example Voyager-II[10], a small difference in delta-v can make years difference in terms of the arrival time at the final destination as will be seen in this thesis.

The search space involved is tremendously vast, encompassing available bodies for flybys and the corresponding flyby times. To mitigate the complexity of the optimization problem, certain constraints are imposed. In many instances, the problem is divided into two subproblems. The first part entails determining the optimal sequence of planets for flybys, while the second part focuses on optimizing the timing of each flyby. Although these optimizations are distinct, they are interconnected and sometimes referred to as 'outer' and 'inner' loop [4]. The first optimization (outer loop) serves as the groundwork for the second (inner loop). Once the planet sequence has been established, the algorithm proceeds to optimize the trajectory based on the given sequence. This iterative process continues until convergence is achieved for the best sequence and trajectory [11][12]. The approach introduces an additional layer of complexity to the problem as it requires a lot of computations. To mitigate this complexity, simplifications are usually implemented in the form of constraints by for example limiting the number of planets involved in flybys and their potential arrangements. Moreover, constraints are also imposed on the duration of each leg in the trajectory. However, it is crucial to not shrink the search space too much with these

constraints, as they can potentially exclude the optimal trajectory, i.e. avoid pruning out the optimal solution. Thus, the pruning phase must be done with caution and well-informed such that the constraints are sufficiently permissive to avoid pruning the optimal trajectory.

Along with the MGA trajectory model, there exists a sub-variant known as MGA-Deep Space Maneuvers known as 'MGA-DSM' which considers the utilization of deep space delta-v maneuvers to enhance spacecraft speed at any point in the trajectory. This is in contrast to MGA, which introduces a delta-v impulse at the closest approach to the flyby planet. While this method shows promise to further optimize the trajectory by introducing greater flexibility, it also introduces additional complexity. This thesis focuses solely on the MGA model, thereby permitting delta-v impulses exclusively at perigee passages during flybys. However, more important than fine-tuning the trajectory once a given sequence has been predefined, it is more beneficial in terms of total delta-v to choose the most appropriate flyby sequence in the first place [4]. This topic will be further expanded on in Section 7.1.. In this study, a large variety of potential sequences were tested including the option of having flybys around the same planet consecutively as well as changing the number of potential flybys. The sample results for 20 of the potential flyby sequences the Galileo spacecraft could have potentially taken are shown below:

Table 1.1: Top 20 candidate sequences for MGA Earth to Jupiter mission [4]

No.	Sequence	Total ΔV, $\mathrm{km/s}$
1	EVVEEJ	3.68
2	EVVEJ	4.72
3	EVVEEEJ	5.24
4	EVVVEJ	5.52
5	EVEJ	5.83
6	EVVEVJ	5.91
7	EEMEJ	5.96
8	EEVEEJ	7.03
9	EEMEEJ	7.38
10	EEEEJ	7.61
11	EEEJ	7.61
12	EEEEEJ	7.67
13	EEVVEEJ	7.77
14	EVVEMEJ	7.80
15	EVVVJ	7.91
16	EVEEJ	7.92
17	EEEVEJ	8.07
18	EVJ	8.14
19	EVVVVJ	8.26
20	EEJ	8.30

As can be seen from Table 1.1, rather than introducing complexities in the 'inner' loop to further optimize the trajectory and flyby times, determining the right sequence is paramount. Moreover, the actual Galileo mission flew sequence $No.16$, which requires about twice as much delta-v compared to the $No.1$ flyby sequence which included an additional gravity assist around Venus. However, while mission designers aim to minimize the delta-v as much as possible, it also has to be balanced with various other factors, most notably the

travel and arrival time (The difference between the two stemming from the time window of the launch date, since the launch window can also be spanning over several years, waiting for too many years likely contradicts the point of aiming to collect valuable data to do science today. Therefore, even a short travel time that necessitates a certain geometric arrangement of the planets in the solar system that only arises in decades may not be an acceptable approach. Hence, both, the arrival time and travel time need to be taken into consideration). Moreover, as the mission profile can vary significantly from one trajectory to another, there are also scientific as well as engineering advantages to targeting an earlier arrival date compared to the minimum delta-v solution. For example, arriving when the 'local' geometry is appropriate for a given mission that leverages resonance of moons and planets such as Europa Clipper [13]. With an earlier arrival, data starts being gathered potentially years before and more may be obtained given the total lifetime of the spacecraft at the mission destination can be expanded. Additionally, as interplanetary space is exposed to unprotected solar radiation as the spacecraft is not within another planet's magnetic field, it requires a certain amount of radiation hardening, temperature control systems, battery etc. The size of those may be able to be reduced which means a reduction in launch mass and hence mission cost, as the spacecraft spends a reduced time in those harsher regions where it is not in the operating conditions.

Therefore, this work will focus on optimizing in particular the inner loop using specifically genetic algorithms and further take into account time in order to incorporate realistic deep space mission planning capability to this interplanetary trajectory design tool. Moreover, the algorithm developed will structured to allow for smooth incorporation of the outer loop optimization through the use of the same genetic algorithm. In an effort to not overreach the scope of this thesis and go into a detailed analysis of the effects of introducing time at different priority rate (as will be shown in Equation 1.4), the final implementation of the outer loop optimization is de-prioritized. As the previous work by Iker Diaz Cilleruelo focused exclusively on the minimization of delta-v of a trajectory with a predefined flyby sequence, this thesis will add time as a parameter to the objective function to optimize the inner loop [3]. Moreover, the outer loop may also be optimized using the same genetic algorithm technique in the future.

The thesis is concentrated on the second problem: determining the trajectory based on the leg intervals. Consequently, the planet sequence within the trajectory is predetermined and serves as an input to the problem.

As this is a global optimization problem, it involves variables and constraints, which can be defined using the variable to be optimized, in our case the time events of the trajectory, t_i:

$$\vec{X} = [t_1, t_2, \ldots, t_n]. \tag{1.1}$$

The $\vec{X}$ vector comprises all the mission's time intervals, where t_1 denotes the departure date. Each $t_i, \forall i > 1$, signifies the date of a flyby. Finally, t_n indicates the arrival date. Utilizing this vector enables the inference of the trajectory by computing the transfers between planets at the corresponding times.

The objective function is composed as follows:

$$C = f(\vec{X}) + g(\vec{X}), \tag{1.2}$$

where $g(\vec{X}$ represents the penalty function, $f(\vec{X})$ are the delta-v injections during the flybys

in addition to the departure delta-v and $t(\vec{X})$ is the time taken to the final destination. The hard constraints consist of bounds on the optimized variables:

$$t_{min} < t_i < t_{max}. \tag{1.3}$$

Soft constraints, such as requiring the perigee passage height to be at least 10% above the radius of the flyby planet, are incorporated through the penalty function.
Solving this optimization problem necessitates leveraging heuristic global optimization algorithms, which are combined with deterministic search space pruning algorithms to reduce complexity and enhance performance [14]. Nonetheless, these algorithms are not straightforward and demand intricate optimization techniques to solve the problem effectively.
In the past, deterministic optimization algorithms were the only viable choice. However, thanks to advancements in computational power, new algorithms can now be employed for trajectory optimization such as evolutionary algorithms. Evolutionary algorithms encompass various variants, including genetic algorithms, differential algorithms, particle swarm optimization, Gaussian adaptation, and others. Despite their differences, these algorithms share the same fundamental principles of modeling the natural processes of evolution and selection, converging a given population to an optimal solution through the principle of survival of the fittest. The 'fitness' of each individual solution, i.e. a trajectory composed of a unique time event vector, $\vec{X}$, is then evaluated based on the function:

$$fitness = \alpha \cdot (deltaV + penalty) + \beta \cdot \ sf \cdot time \tag{1.4}$$

where sf is the scaling factor used to normalize the units of time such that a comparable value to the delta-v term is obtained. The α and β parameters are introduced to allow the user to set weights to the delta-v and time terms of the fitness evaluation respectively. This creates a simple method for mission designers to fine-tune how they want to prioritize the fuel vs the time taken to the destination of the mission. Moreover, the different penalties are introduced as an artificial inflation of the delta-v to make the individual trajectories less fit and hence less likely to be selected for reproduction.

Evolutionary algorithms require greater computational power in comparison to deterministic optimization algorithms. However, they offer a less complex approach, as they only require "solutions" to the problem, i.e. an initialization of the $\vec{X}$ time vector rather than needing an extensive array of variables, constraints, and accurate modeling of functions and gradients that are challenging to determine for nonlinear problems.

CHAPTER 2. OBJECTIVES

This thesis embarks on the development of a comprehensive trajectory optimizer that encompasses all the essential steps required for modeling an MGA interplanetary trajectory which are the computation of the planet's ephemeris (position and velocity at a specific point in time), the calculation of transfer orbits (Lambert transfer), and the determination of flybys. Each individual trajectory's values are subsequently employed in conjunction with a genetic algorithm to evolve the population and converge to the optimal solution. In this work, the goal is to minimize the delta-v as well as the travel time (depending on the prioritization parameters α and β) given by Equation 1.4, hence the one with the lowest fitness value is the fittest of the population. Similarly, the same evolutionary optimization approach can be used to find the optimal flyby sequence, therefore having both the inner and outer loop optimized using genetic algorithms. To facilitate this process, the structure of the algorithm including the genetic algorithm is aimed to be designed such that it is reusable and modular. In the future once both loops are completed, the algorithm first determines the best sequence and then we optimize that sequence further and can aim for a specific arrival time using the prioritization variables α and β seen in Equation 1.4. Overall, the collection of the various parts shall make for a complete deep space mission trajectory design tool, which only requires a final destination as an input along with a prioritization of delta-v vs time through the parameters α and β. However, to not overextend the scope of this thesis, only the inner loop will be optimized as ellaborated and the sceleton code architecture for a similar incorporation of the outer loop will be completed.

This thesis aims to demonstrate how genetic algorithms can be employed to discover near-optimal MGA interplanetary trajectories, albeit at the cost of increased computational power, offset by the reduced complexity when compared to deterministic optimizers. Moreover, this work intents to highlight the ability to tune the desired trajectory depending on how delta-v versus time is prioritized. Part of this endeavor entails the study between the additional fuel cost as a method to arrive at an earlier date to the final destination. To conclude the analysis, the algorithm is tested using the real-case missions of Voyager-II and also compared to the previous iteration of the algorith in [3] (only minimizing delta-v) to illustrate the improvements made.

CHAPTER 3. INTERPLANETARY TRAJECTORY

Interplanetary trajectory refers to the path followed by a spacecraft as it travels between different celestial bodies, such as planets. The design of this trajectory plays a crucial role in optimizing the spacecraft's delta-v requirements.

3.1. Transfer Orbits

Transfer orbit refers to the trajectory between two celestial bodies. Various types of transfer orbits exist. The most prominent ones include Hohmann transfers, bi-elliptic Hohmann transfers, and low-energy transfer orbits. Each type has unique characteristics that make it suitable for different scenarios. These transfers involve carefully calculated trajectories and precise engine burns to optimize energy and time requirements. The Hohmann transfer is known for its efficiency, as it minimizes the required delta-v for the transfer. Another type of transfer is the bi-elliptic Hohmann transfer, which involves two elliptical orbits to achieve the desired transfer. This transfer is more efficient when the ratio between the outer and inner orbit radii is greater than 15.

3.1.1. Hohmann Transfer

This transfer involves a two-impulse maneuver between orbits that share the same focus point, typically orbiting the same body. The transfer begins with a burn at the perigee of the departure orbit and concludes with a burn at the apogee of the target orbit. The spacecraft is boosted into the transfer orbit through these burns, which increase its energy due to the higher semi-major axis.

The delta-v required for a Hohmann transfer departing from a point D in a planet with a circular orbital speed V_1 is given by the equation:

$$\Delta V_D = V_D^{(v)} - V_1 = \sqrt{\frac{\mu_{SUN}}{r}}\left(\sqrt{\frac{2r'}{r+r'}} - 1\right), \tag{3.1}$$

where μ_{SUN} represents the standard gravitational parameter of the Sun, V_D and $V_D^{(v)}$ are the departure velocities in the planet's and heliocentric frame respectively. Moreover, r and r' are the radii of the departure and target orbits respectively, shown in Figure 3.1.

The delta-v required at the arrival orbit is given by:

$$\Delta V_A = V_2 - V_A^{(v)} = \sqrt{\frac{\mu_{SUN}}{r'}}\left(1 - \sqrt{\frac{2r}{r+r'}}\right), \tag{3.2}$$

where V_A and $V_A^{(v)}$ are the arrival velocities in the planet's and heliocentric frame respectively at r'.

Through this procedure, the original inner circular orbit of radius r is changed into an elliptical orbit with the apogee being tangent to the target outer circular orbit using the first

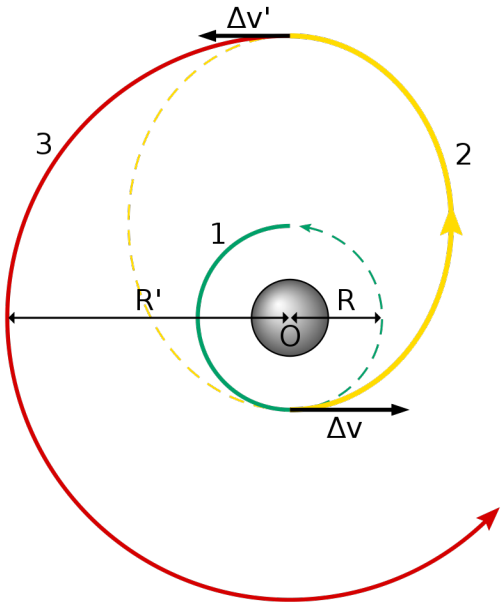

Figure 3.1: Hohmann transfer orbit (Credit: Hubert Bartkowiak)

delta-v burn. This causes the motion depicted with the yellow line marked '2' above. Secondly, once the spacecraft is at apogee, another burn is performed in order to circularise the orbit with a new radius of r', resulting in the final motion shown in red (marked as '3') in Figure 3.1.

Similarly, the procedure for the bi-elliptic Hohmann transfer is detailed in [3].

3.2. Patched conic

Interplanetary trajectories involve complex interactions between spacecraft, the sun, planets and other celestial bodies. To simplify these trajectories, the patched conic approximation is commonly employed [1] Sect 8.5 which assumes that when a spacecraft is outside the sphere of influence r_{SOI} of a planet [1] Sect. 8.4, it is primarily influenced by the Sun's gravity, following an unperturbed Keplerian orbit around the Sun. When the spacecraft enters the sphere of influence, it transitions to an unperturbed Keplerian orbit around the planet. The sphere of influence represents a region around a celestial body, such as a planet, where the body's gravitational force dominates over the Sun's gravitational force. In this work the sphere of influence of a planet is calculated as:

$$r_{SOI} = R\left(\frac{m_p}{m_s}\right)^{\frac{2}{5}}. \tag{3.3}$$

As the radius of the sphere of influence also depends on the force of gravity from the sun, the planets with larger semi-major axis tend to have larger ones as shown below:

For our purposes, focusing on interplanetary transfers without interactions from other bod-

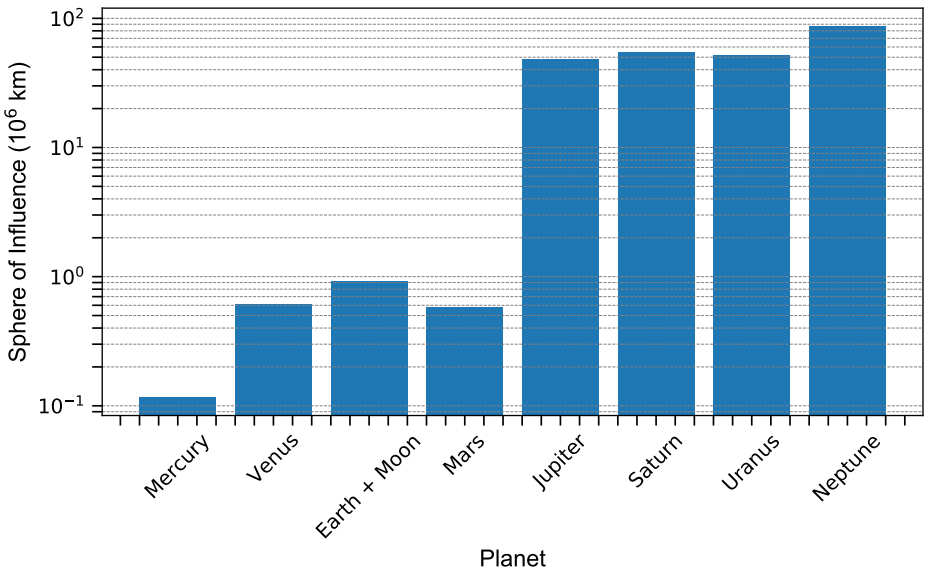

Figure 3.2: Sphere of Influence of Planets, constructed using NASA Planetary Fact Sheet

ies (thus treating it as a set of two-body problems), known as the patched conic approximation, suffices. This allows us to break down the trajectory into three steps: departure, transfer, and arrival. Heliocentric speeds are defined relative to the planets inside their spheres of influence, as well as relative to the Sun outside these spheres. By patching together the relative speeds, we can analyze interplanetary transfers effectively. While this simplification is generally accurate, it is important to note that in situations where two bodies are in close proximity or for systems like the Sun-Earth-Moon, the more complex 3-body problem must be considered [15]. See Figure 3.3 for a visual representation of planetary departure under the sphere of influence of a planet (adapted from [1] Chap. 8, Figure 8.8).

3.3. Planetary gravity assist flyby

3.3.1. Departure

In this work all departures originate from the earth and by definition, the departure velocity needs to be larger than the escape velocity. To position the spacecraft into the desired transfer orbit, its heliocentric velocity at the crossing of the sphere of influence should align with the asymptote of the departure hyperbola as seen in Figure 3.3. At that point, the velocity should match the desired hyperbolic velocity beyond the sphere of influence, v_∞, which means we need velocity v_p at the perigee of the hyperbola to be:

$$v_p = \frac{h}{r_p} = \sqrt{v_\infty^2 + \frac{2\mu_1}{r_p}} \tag{3.4}$$

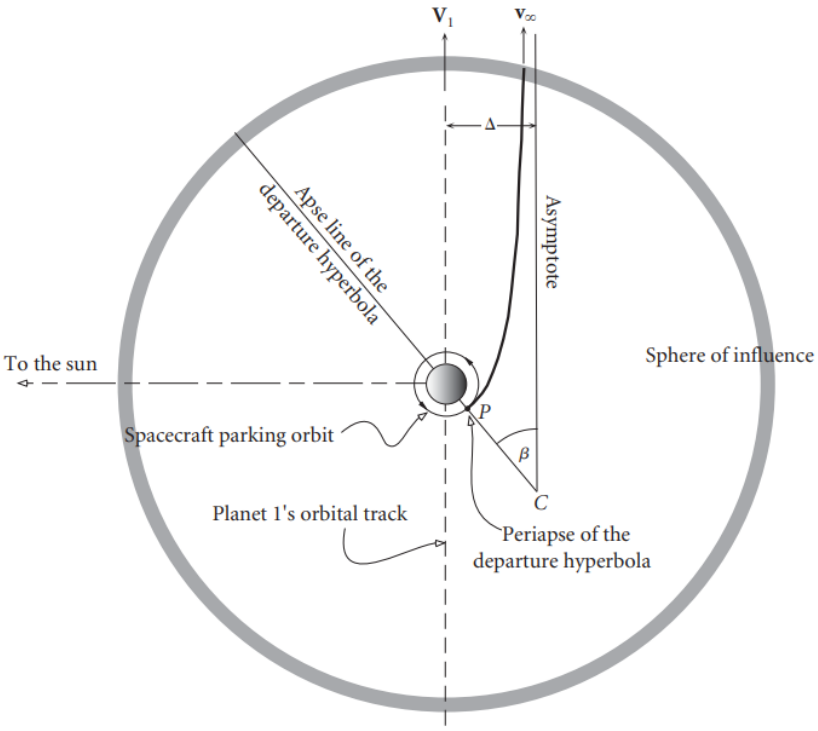

Figure 3.3: Planetary departure under the sphere of influence of a planet (from [1] Chap. 8, Figure 8.8)

Assuming that the spacecraft is originally in a circular parking orbit defined by v_c:

$$v_c = \sqrt{\frac{\mu_1}{r_p}} \tag{3.5}$$

To achieve this final hyperbolic velocity from a parking orbit as illustrated in Figure 3.3, the required delta-v is given by [1] (Eq. (8.42)):

$$\Delta v = v_p - v_c = v_c \left(\sqrt{2 + \left(\frac{v_\infty}{v_c} \right)^2} - 1 \right) \tag{3.6}$$

3.3.2. Flyby maneuver

Having understood the transfers required as well as the departure of the journey, the individual legs of the trajectory need to be patched using adequate gravity assists with a delta-v maneuver at the perigee of the flyby.

In this process, the spacecraft transitions from the Sun's sphere of influence to the planet's and a change of velocity as well as deflection occurs. It is important to note that in the planet's reference frame, the speed of the spacecraft remains unchanged, i.e. it enters and exits its sphere of influence with the same speed. However, when considering the sun's perspective there has been a change in the velocity. Typically, the spacecraft enters from the trailing side in order to increase its speed because of the gravitational pull of

the planet. Subsequently, in the planet's view, it decreases at an equivalent rate after the perigee passage (at the planet's leading side), the only difference being that a deflection occurs denoted with a turning angle δ. A depiction of a gravity assist is shown in Figure 3.4 below.

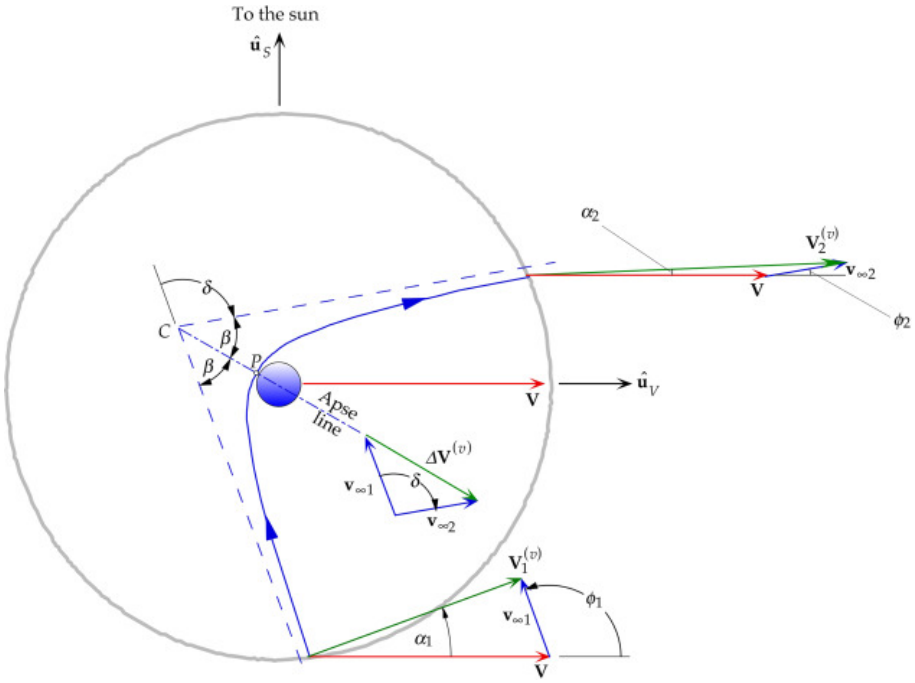

Figure 3.4: Trailing-side flyby (from [1] Chap. 8, Figure 8.19)

Here, the planets velocity is $\vec{V}$ shown with a red vector whereas $\vec{v}_{\infty_1}$ and $\vec{v}_{\infty_2}$ are the entry and exit velocity of the spacecraft in the planet's frame with an angle ϕ_1 and ϕ_2 respectively, creating a turning angle $\delta = \phi_2 - \phi_1$. The heliocentric velocities are $\vec{V}_1^{(v)}$ and $\vec{V}_1^{(v)}$ respectively which make an angle to the planets velocity vector $\vec{V}$ of α_1 and α_2 respectively.

In the sun's view, the spacecraft gains speed from the planet's motion, meaning the heliocentric speed has increased due to the flyby around the planet [1]. During this maneuver, it is important that the perigee passage altitude is sufficiently above the surface of the flyby planet in order to prevent a potential reduction in velocity due to atmospheric drag. This is implemented in the form of a delta-v penalty if the distance at perigee is smaller than $1.1 r_{planet}$.

As seen in Figure 3.4,$\vec{V}_1^{(v)}$ and $\vec{V}_2^{(v)}$ represent the heliocentric velocities at the inbound and outbound crossing points respectively. $\vec{v}_{\infty_1}$ and $\vec{v}_{\infty_2}$ are the excess velocities with which the gain in delta-v (in the heliocentric reference frame) can be calculated as follows:

$$\Delta\vec{V}^{(v)} = \vec{V}_2^{(v)} - \vec{V}_1^{(v)} = (\vec{V} + \vec{v}_{\infty_2}) - (\vec{V} + \vec{v}_{\infty_1}) \tag{3.7}$$

$$\Delta\vec{V}^{(v)} = \vec{v}_{\infty_2} - \vec{v}_{\infty_1} = \Delta\vec{v} \tag{3.8}$$

Note that the delta-v can be positive or negative depending on which side the gravity assist occurs from. When the flyby maneuver is conducted by entering from the leading side of

the planet's rotation, the speed of the spacecraft decreases as it travels in the opposite direction. Conversely, if both the spacecraft and the planet move in the same direction, i.e., the spacecraft enters from the trailing side, the delta-v is positive, which is the more common case.
Considering an unpowered flyby without a delta-v injection at the perigee, the relative hyperbolic excess velocities are:

$$\vec{v}_{\infty_1} = \vec{V}_1^{(v)} - \vec{V}_p \tag{3.9}$$

$$\vec{v}_{\infty_2} = \vec{V}_2^{(v)} - \vec{V}_p \tag{3.10}$$

Next, we determine the eccentricity of the flyby parabola and the angular momentum using the following equations ([1] Eq. 8.83)

$$e = 1 + \frac{r_p v_\infty^2}{\mu} \qquad h = r_p \sqrt{v_\infty^2 + \frac{2\mu}{r_p}}. \tag{3.11}$$

Using the eccentricity the turning angle is calculated as follows:

$$\delta = 2\sin^{-1}\left(\frac{1}{e}\right). \tag{3.12}$$

In the context of this work, as the turning angle and delta-v requirements from the various legs of the trajectory (that is the incoming and outgoing velocity) need to be patched, a delta-v thrust is injected at the perigee of the hyperbolic trajectory around the planet.

3.4. MGA missions - an overview

Multi-gravity assist trajectories have been pivotal in numerous space missions. Notable examples include NASA's Voyager program, which utilized Jupiter and Saturn's gravity to achieve remarkable speeds and capture stunning images of and from the outer solar system. The Galileo mission employed gravity assists to explore Jupiter and its moons, while the BepiColombo mission is utilizing multiple gravity assists to reach and study Mercury. Looking ahead, the upcoming Europa Clipper mission by NASA will employ gravity assists to investigate Jupiter's moon Europa in search of potentially habitable environments.

3.4.1. Time-sensitivity in space missions

While in the scope of [3], multi-gravity assist trajectories can be designed with the minimum delta-v requirement, which essentially just provides one final trajectory. However, in certain space missions, the timing of arrival becomes a critical factor, surpassing the sole consideration of fuel efficiency in trajectory planning. This is particularly relevant when time-sensitive observations or rendezvous with specific celestial bodies are involved.

3.4.1.1. Past mission examples:

For instance, the Galileo mission to Jupiter required precise timing to study key events such as the impact of Comet Shoemaker-Levy 9 [16]. Similarly, the BepiColombo mission to Mercury had to account for the planet's complex gravitational field and orbital dynamics, necessitating specific launch windows for optimal arrival [17].

3.4.1.2. Future MGA mission

Future missions like the Europa Clipper also prioritize timing to ensure close flybys of Jupiter's moon Europa during favorable conditions for scientific investigations [13].

These examples illustrate how mission objectives and scientific priorities can influence the choice of multi-gravity assist trajectories, where arriving at the desired destination at the right time takes precedence over minimizing fuel consumption alone. Therefore in this work a comparison will be made between the real mission trajectory, the minimum fuel trajectory and a third trajectory taking time into consideration arriving at a similar date compared to the actually flown spacecraft.

3.4.2. Voyager II mission

To give an example of what the trajectory and the addition of gravity-assists to the velocity of a spacecraft looks like, a brief summary of the Voyager-II mission is presented below.

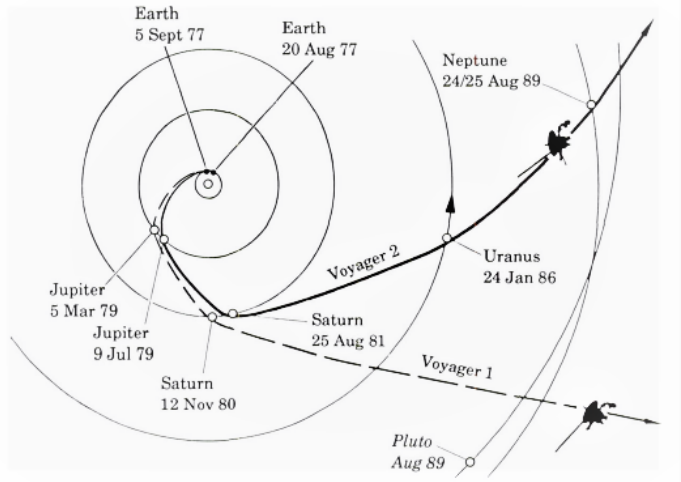

Figure 3.5: Voyager I and II interplanetary trajectories (from [2] Figure 1-4)

A remarkable aspect of the Voyager-II mission is also the number of flybys that were undertaken which allowed it to become one of the first man-made objects to leave the solar system in November of 2018 [18].

	Time	Perigee (km)
Launch	20 Aug. 1977	-
Jupiter Flyby	9 Jul. 1979	721670
Saturn Flyby	26 Aug. 1981	161000
Uranus Flyby	24 Aug. 1986	107000
Neptune Flyby	25 Aug. 1989	29240

Table 3.1: Timeline of Voyager II events [2]

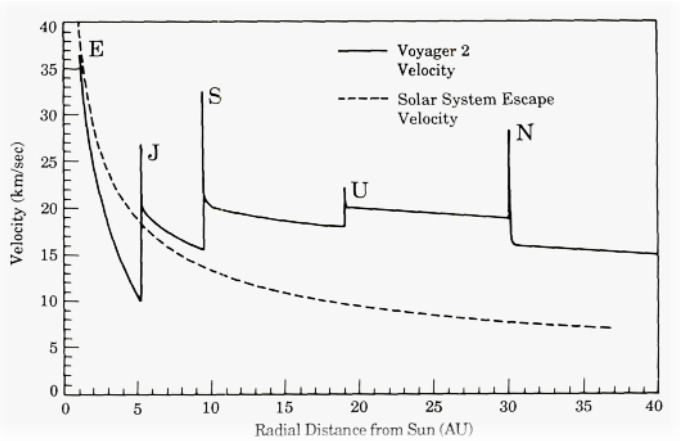

Figure 3.6: Voyager II velocity evolution (from [2] Figure 11-6)

CHAPTER 4. EVOLUTIONARY ALGORITHMS

Evolutionary algorithms (EAs) are heuristic optimization methods which, as the name suggests, aim to replicate the processes of nature such as selection, reproduction, and other evolutionary mechanisms to optimize problem solutions. By adapting to the problem environment over time, individuals (which in our case are trajectories) evolve and improve the overall optimization process. Although EAs are non-deterministic, they can provide either the optimal solution or a close approximation.

The study of evolutionary algorithms dates back to the 1950s and 1960s, pioneered by computer scientists [19]. Although evolutionary algorithms are considered artificial intelligence/machine learning (AI/ML) techniques, they differ significantly from traditional AI/ML approaches like neural networks. EAs rely on a known problem model instead of using data to model and solve the problem for new solutions, which means that prior knowledge of the problem is required for optimization.

In particular, evolutionary algorithms are well-suited for optimizing complex problems with large search spaces that are impractical to exhaustively search using traditional methods. In our case, the search spaces are defined by the different time windows allowed for the legs of the interplanetary trajectory. This abstraction comes at the cost of increased computational requirements and no guarantee of converging to the optimal solution, which made them hard to implement some time ago when the computational power was significantly limited. Nowadays, this approach provides a rather simple way to solve world problems of high complexity by trialing a population of solutions.

Interplanetary trajectories involve an extremely large search space and are challenging to model with deterministic algorithms that require gradients and mathematical constraints. While the optimality of the solution is assumed, different inputs yield nearly the same solution, as shown in Section 6. Although the computational demands increase, one advantage of evolutionary algorithms is that certain steps can be parallelized through the use of multithreading, which alleviates the computational processing time.

4.1. Genetic algorithms

Genetic algorithms (GAs) are a widely used type of evolutionary algorithms developed by John Holland in the 1970s [20] to study adaptive behaviours. They aim to mimic natural selection/evolution in artificial systems [21] and traditionally require individuals to be represented as bit strings [**?**] (CITE THIS PROPERLY). While GAs select the fittest solution from an initial set of random values, their performance, robustness, and convergence can be improved by incorporating other evolutionary processes which in this case revolve around 'gene' mutation, crossover, etc. The fittest individual is determined by maximizing a user-defined objective function. Although GAs are commonly used for single-objective optimization, as was done in [3], they can also be extended to handle multi-objective problems [22]. In astrodynamics multiple objective GAs have been applied already [23], but in

their case delta-v along with the final spacecraft mass was optimized. In this work, both time and delta-v can be simultaneously minimized by defining an appropriate objective function as shown in Equation 1.4.

Individuals, which in the scope of this work are the aforementioned $\vec{X}$ vector containing the time events, are represented as strings of bits, known as chromosomes. These chromosomes are composed of genes, which influence specific parts of the individual (in this case the flyby dates) and its corresponding solution. Although this representation simplifies nature, it captures the key processes required for evolution. The individuals are evaluated based on the objective function:

$$\begin{aligned} i : 001011001010 \quad &\longmapsto \quad \vec{x} \\ y = f(\vec{x}), & \end{aligned}$$

where y represents fitness and f represents the objective function in Equation 1.4 and $\vec{x}$ is the vector of time events mapped into a binary string.

The initialization of individuals involves assigning random values within the predefined search space. It is crucial to keep the search space large as to not prune out the optimal solution. Therefore, without proper initialization and diversity, the algorithm may stagnate and converge to local minima or maxima instead of finding the global minima/maxima. After initialization, each individual's fitness is evaluated which is subsequently used to generate a likelihood to be chosen for selection. Genetic operators (described in Section 4.1.1.) are then applied to generate new individuals, simulating various natural processes that transmit genetic material and drive population evolution. This process is repeated for multiple generations until a predetermined number of generations is reached or a given convergence criterion terminates the algorithm. The evolution is not solely dependent on the convergence between consecutive runs, but also on the emergence of random values that enhance individual fitness and subsequently improve the fitness of future generations.

4.1.1. Genetic operators

Genetic operators are the core of GA and distinguish them from random search space algorithms. These operators facilitate intelligent selection and transmission of beneficial genetic information between generations, driving the population towards the optimal solution. The main operators used in this work are selection, mutation and recombination which are described in the following section.

4.1.1.1. Selection

The selection operator is the first one used in GA and models the principle that individuals with traits that are more favorable for their environment have a higher chance of surviving and reproducing, thereby passing on their advantageous traits to future generations. In the context of this work, this concept is applied to the population of individuals (solutions) that are being evolved to solve for the optimal interplanetary trajectory by leveraging multiple gravity assists. Two commonly used methods for selection are:

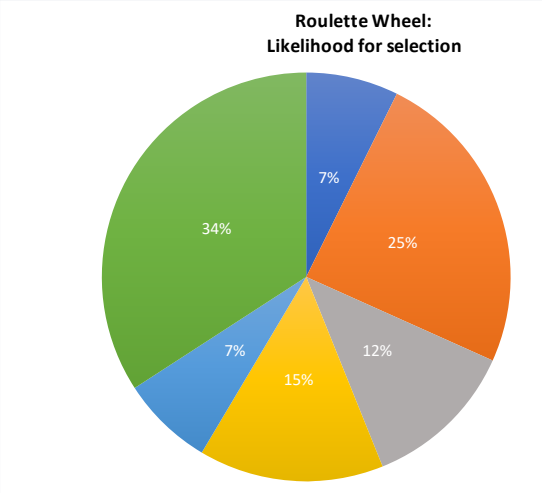

Figure 4.1: Illustration of roulette selection.

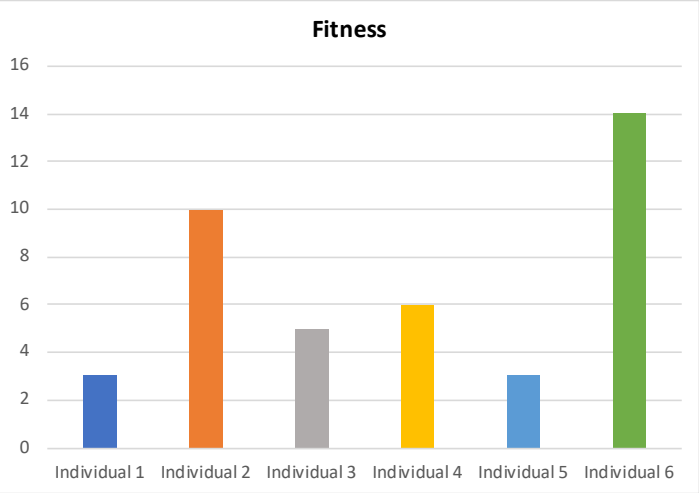

Figure 4.2: Fitness of 6 Individuals.

1. **Roulette selection:** In this method, the probability of being selected is proportional to the fitness values of the individuals as follows:

$$p_i = \frac{f_i}{\sum_{j=1}^{Npop} f_j}, \tag{4.1}$$

Here, f_i is the fitness of individual i and $Npop$ is the population size used to express the percentage chance of selection p_i. While the fittest individuals have the highest probability of being selected, weak individuals still have a small chance of being selected as well, which is crucial to avoid the algorithm rapidly converging in local minima [24]. The name 'Roulette Selection' stems from the selection being analogous to a roulette as shown below with an example population of six individuals:

2. **Tournament selection:** This method involves an initial selection phase followed by the tournament. Firstly, multiple individuals are chosen using for example the aforementioned roulette selection. In the subsequent tournament phase, the selected individuals are compared, and the fittest one is chosen. In Table 4.1 taken from [3], four individuals were selected using roulette selection, and the third individual is selected as the fittest in the tournament phase. Note that in the implementation the fitness is renormalized to be within 0 and 1, such that the comparison is of numbers with similar magnitude.

Individual	Fitness
01011	0.12
10110	0.04
10001	**0.23**
10101	0.013

Table 4.1: Illustration of the tournament selection method [3]

Tournament selection has the advantage of faster convergence since fitter individuals are selected more frequently compared to the roulette selection. However, it can lead to a loss of diversity within the population due to its greedy nature, potentially hindering the attainment of the optimal solution.

4.1.1.2. Elitism

Although not considered a traditional genetic operator, elitism ensures the retention of highly fit individuals in the population, allowing them to directly pass their genetic material to the next generation without any modifications. This approach aims to maintain and protect the best solutions found so far during the evolutionary process as there is potential to lose the fittest individuals during reproduction or omission in the selection process. Hence, introducing elitism ensures that the fittest solution of a given generation is at least as good as the best individual of the previous generation, making the solution always improve or remain unchanged. Given that there could be various local minima, the number of elites in a given population is a function of the size itself in order to account for the possibility of having multiple different individuals that are of similar competitiveness.

4.1.1.3. Reproduction and Crossover

Reproduction and crossover operators dictate how the selected genetic material is transmitted and mixed to the next generation once individuals have already been selected. They facilitate the exchange and recombination of favorable genetic traits to generate new chromosomes which represent a new individual.

Reproduction is a straightforward process where the selected parents are directly carried over to the next generation without any modifications to their chromosomes. It helps maintain diversity by preserving the existing genetic material and preventing its loss but does not add new variations into the population

On the other hand, crossover is the primary operator responsible for altering the chromosomes of the parents to create offspring with a combination of their genetic material creating a new individual called offspring or children. During crossover, specific points in the parent chromosomes, called crossover points, are chosen randomly. The genetic material between these crossover points is exchanged between the parents, resulting in offspring with a combination of genetic information from both parents. The purpose of crossover is to explore new areas of the search space by combining favorable traits from different individuals and potentially generating offspring with improved fitness.

There are different types of crossover methods used in GAs, including:

Single-point crossover: A single crossover point is selected, and the genetic material beyond that point is exchanged between the parents' chromosomes. This method creates two offspring by combining genetic information before and after the crossover point. Here, an example where the crossover point is the fifth bit.

$$\begin{array}{llcll} \text{Parent 1: } & 11 & 110 & 0111 & \rightarrow \quad \text{Child 1: } 11 \quad 111 \quad 0110 \\ \text{Parent 2: } & 11 & 011 & 0110 & \rightarrow \quad \text{Child 2: } 11 \quad 010 \quad 0111 \end{array}$$

Two-point crossover: Similar to single-point crossover, but two crossover points are chosen, and the genetic material between these two points is exchanged. In this case, the children often remain more similar to their parents as more of the genetic material is shared across them. Similarly, in the following example, the crossover starts at the fifth bit and

ends after the 7th bit.

Parent 1: 11 110 0111 → Child 1: 11 111 0111
Parent 2: 11 011 0110 → Child 2: 11 010 0110

Uniform crossover: Each bit or gene in the offspring is randomly selected from either parent with a 50% chance. This method provides a more diverse recombination of genetic material. A coin is flipped whenever the bits at the same positions are different from the two parents. An example would be as follows:

Parent 1: 11 110 0111 → Child 1: 11 010 0110
Parent 2: 11 011 0110 → Child 2: 11 111 0111

In this case, the third, fifth and ninth bit are different and the coin flips for those positions was ”heads - tails - heads”, as the bit in fifth location did not flip since ”tail” was chosen.

Single gene uniform crossover: Similar to uniform crossover, this approach operates on a single gene rather than the entire chromosome. A specific gene, such as a range of bits, is selected, and the coin flip method is applied to determine the inherited gene. An example of this method is illustrated in [3].

The choice between reproduction and crossover for two parents is determined by a random number generated between 0 and 1. Fundamentally, it is important that crossover is assigned a higher probability as it introduces new genetic material. Moreover, over-reliance on reproduction can significantly slow down the convergence of the algorithm as crossover between fit individuals plays a vital role in driving evolution and convergence towards the optimal solution. It enables the transmission of shared genes from fit individuals to new individuals while generating new genetic combinations from differing genes. This process promotes the prevalence of ”good” genes present in many fit individuals, leading to their eventual dominance in the population. For instance, if the most fit individuals consistently have the first bit of a 5-bit chromosome set to 1, indicating a decoded departure date above 32 (since 2^5), it suggests that the optimal date is likely to be higher than this value. Consequently, this gene tends to become dominant, with the entire population inheriting it through the crossover operation.

4.1.1.4. Mutation

Mutation is the final genetic operator in the evolutionary process of genetic algorithms and involves modifying a portion of randomly chosen individuals' genetic material (chromosome), after the reproduction and crossover stages have been completed. While its effects may not be immediately noticeable in a single generation, it plays a significant role in the long-term evolution of the population by ensuring diversity. As the population evolves, it tends to lose diversity and become more homogeneous, through the process described in Section 4.1.1.3.. This phenomenon is a consequence of the selection operator, as the fittest individuals are more likely to be selected, and their genetic material dominates the population, which slows down further evolution. By introducing new mutations and changes to the chromosomes, new genetic material can be generated, increasing the diversity of the population. Morover, mutation also serves to get a population unstuck from local maxima or minima by changing a bit that is commonly shared amongst the fittest individuals.

There are four main types of mutation methods:

Flip Bit mutation: Here a randomly chosen bit is flipped.

$$11 \quad 011 \quad 0110 \xrightarrow{\text{Bit 5}} 11 \quad 11\mathbf{0} \quad 0110$$

Boundary mutation: In this method, a sequence of n bits (a single gene) is randomly flipped entirely to 0 or 1 using a coin flip.

$$11 \quad 011 \quad 0110 \xrightarrow{\text{Gene 3 to 5}} 11 \quad \mathbf{111} \quad 0110$$

Uniform mutation: In this case, each bit within a randomly selected gene is determined by individual coin flips while the other remains untouched. This method is similar to uniform crossover but applies to the bits within a gene rather than the entire gene.

$$11 \quad 011 \quad 0110 \xrightarrow{\text{Gene 3 to 5}} 11 \quad \mathbf{001} \quad 0110$$

Inversion mutation: Here, a gene is selected and the bits are flipped.

$$11 \quad 011 \quad 0110 \xrightarrow{\text{Gene 3 to 5}} 11 \quad \mathbf{100} \quad 0110$$

These four mutation methods contribute to the creation of new genetic material and maintain diversity within the population. By introducing random changes to the chromosomes, mutation allows for the exploration of different regions of the search space and helps overcome stagnation. The selective application of mutation ensures that the population continues to evolve and avoids premature convergence to suboptimal solutions.

CHAPTER 5. ALGORITHM IMPLEMENTATION

5.1. Overview

This section presents the algorithm developed to optimize interplanetary spacecraft trajectories within the solar system which has been original developed by [3] primarily using C++ and then modified. Some non-core visual components were coded in Python. C++ was chosen for its optimization, efficiency, standard library support, compilation benefits (faster execution), and multithreading capabilities which are particularly useful since genetic algorithms are computationally expensive.

The algorithm can be decomposed into two main parts: the trajectory solver itself, involving the design and computation of the interplanetary trajectory, and the genetic algorithm used to optimize the trajectory. Although both parts are incorporated into the algorithm, they can be used separately, allowing the reuse of the genetic algorithm independently from the problem to optimize. This enables the use of the same genetic algorithm for the optimization of the flyby sequence which is used to optimize the order of planetary flybys that will be used.

The overall diagram of the code can be summed up as follows:

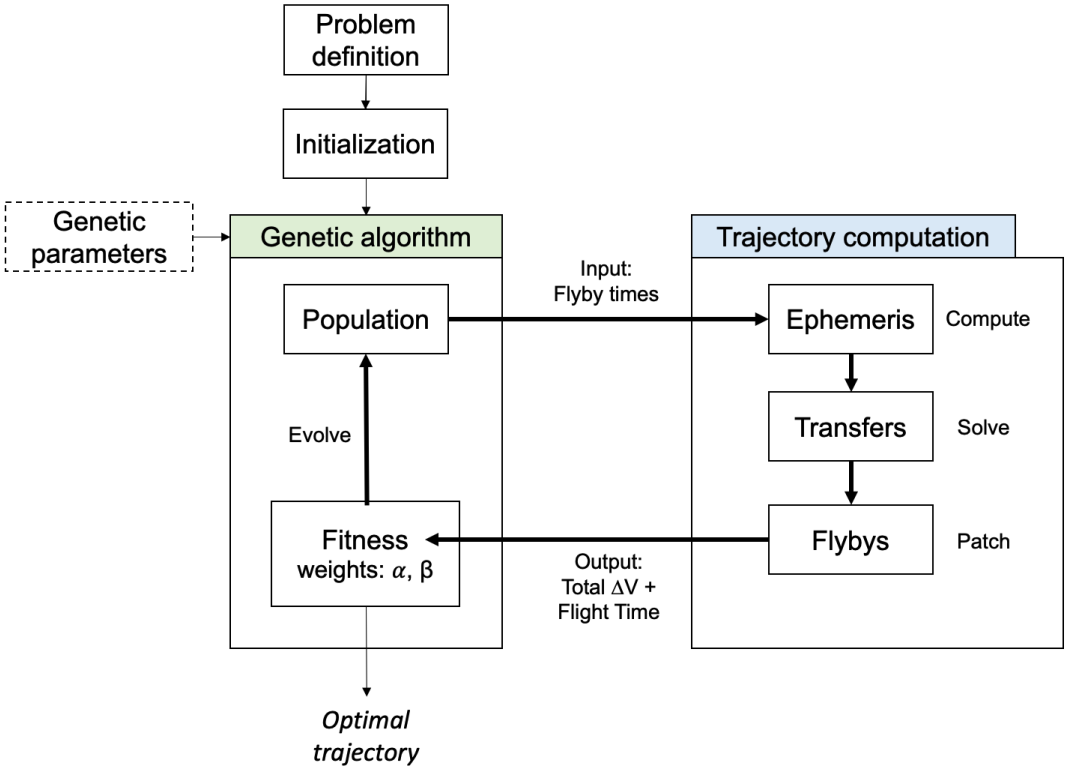

Figure 5.1: Code diagram

5.2. Process Chronology

The first step of the algorithm is in charge of initiating as many individuals as specified by the GA parameters N_{POP}, meaning a vector $\vec{X}$ is created for each individual containing randomly generated flight times within the prespecified time intervals for each leg as described in Section 1. Then, the genetic algorithm solves the trajectory for each individual and before evaluating the delta-v cost of the obtained solution. Initially, using the specific fly-by dates that are computed using $\vec{X}$, the position and velocity of the planets at those time events are calculated, which is known as the planet's ephemeris. This information is then used to solve for the transfer from one planet to another, which is conducted using a Lambert solver. Finally, as each leg now has a specified entry and exit velocity, those two need to be patched with the appropriate gravity assist and delta-v thrust. Finally, this delta-v is taken as the cost along with the time taken since the first day of the departure window to obtain a fitness for each trajectory. As opposed to the previous iteration of the code which directly used the delta-v as the fitness of the individual, thus making the one with the least delta-v teh fittest individual, in this case the 'units' of the objective function may be in delta-v but it is composed of multiple terms. At this step, the user of this trajectory design tool can specify the weights of importance set to delta-v and time (through α and β respectively), in order to find a family of solutions rather than being limited to the minimum delta-v trajectory. This fitness is used to evolve the algorithm until a specific number of generations is reached. The global problem is defined at the beginning, along with parameters for the genetic operators to modify their behavior and tweak the algorithm's performance, through the processes described in Section 4.1.

5.3. Trajectory Design

5.3.1. Ephemeris

The ephemeris of a planet is the two state vectors $\vec{R}$ and $\vec{V}$ representing the position and velocity in a heliocentric ecliptic frame of a planet at a given time [1] Sect. 8.10. Those vectors are used to determine the location of the planet at the flyby time in the trajectory, as well as its velocity. The heliocentric velocity of the planet is required to compute the flyby maneuver and subsequently patch the conics.
The ephemeris are computed using the orbital parameters of the planets and their rate of change which are measured and determined by the space agencies like NASA. The six elements, shown in Table 5.1 below, are the Keplerian orbital elements of each planet:

- a: semi-major axis [au].
- e: eccentricity.
- i: inclination to elliptic plane [degrees].
- L: mean longitude [degrees].
- w: longitude of perihelion [degrees].
- Ω: longitude of the ascending node [degrees].

Using these six elements, $\vec{R}$ can be calculated. Moreover, the elements with a dot ($\dot{a}$, $\dot{e}$,...) are the rate of change of each element per Julian century, with which $\vec{V}$ is computed. However, these are only accurate for a given time period, as the values of the orbital parameter and in particular their rate of change are not constant. Therefore, after some time they need to be calibrated again with new observations. For the scope of this work the Table 5.1 of orbital elements from 1850 to 2050 is used, as it provides sufficiently accurate values. It is important to note that for dates beyond 2050, other tables must be used with less accuracy. However, none of the missions discussed in this paper go beyond that year.

	a [au, au/Cy]	e [rad, rad/Cy]	i [deg, deg/Cy]	L [deg, deg/Cy]	w [deg, deg/Cy]	Ω [deg, deg/Cy]
Mercury	0.38709927	0.20563593	7.00497902	252.25032350	77.45779628	48.33076593
	0.00000037	0.00001906	-0.00594749	149472.67411175	0.16047689	-0.12534081
Venus	0.72333566	0.00677672	3.39467605	181.97909950	131.60246718	76.67984255
	0.00000390	-0.00004107	-0.00078890	58517.81538729	0.00268329	-0.27769418
Earth	1.00000261	0.01671123	-0.00001531	100.46457166	102.93768193	0.0
	0.00000562	-0.00004392	-0.01294668	35999.37244981	0.32327364	0.0
Mars	1.52371034	0.09339410	1.84969142	-4.55343205	-23.94362959	49.55953891
	0.00001847	0.00007882	-0.00813131	19140.30268499	0.44441088	-0.29257343
Jupiter	5.20288700	0.04838624	1.30439695	34.39644051	14.72847983	100.47390909
	-0.00011607	-0.00013253	-0.00183714	3034.74612775	0.21252668	0.20469106
Saturn	9.53667594	0.05386179	2.48599187	49.95424423	92.59887831	113.66242448
	-0.00125060	-0.00050991	0.00193609	1222.49362201	-0.41897216	-0.28867794
Uranus	19.18916464	0.04725744	0.77263783	313.23810451	170.95427630	74.01692503
	-0.00196176	-0.00004397	-0.00242939	428.48202785	0.40805281	0.04240589
Neptune	30.06992276	0.00859048	1.77004347	-55.12002969	44.96476227	131.78422574
	0.00026291	0.00005105	0.00035372	218.45945325	-0.32241464	-0.00508664

Table 5.1: Keplerian elements and rates (valid from 1800 - 2500). Table from [5]

Note, that it is commonly assumed that all the planets are in the same plane in the solar system. However, that is not accurate as the heliocentric ecliptic is defined as the orbital plane of the earth. Hence, other planets like Mercury can have an inclination i of up to 7 degrees as seen in the first row of Figure 5.1. A planet's orbit in the heliocentric reference frame is illustrated in Figure 5.2 with the orbital parameters introduced above.

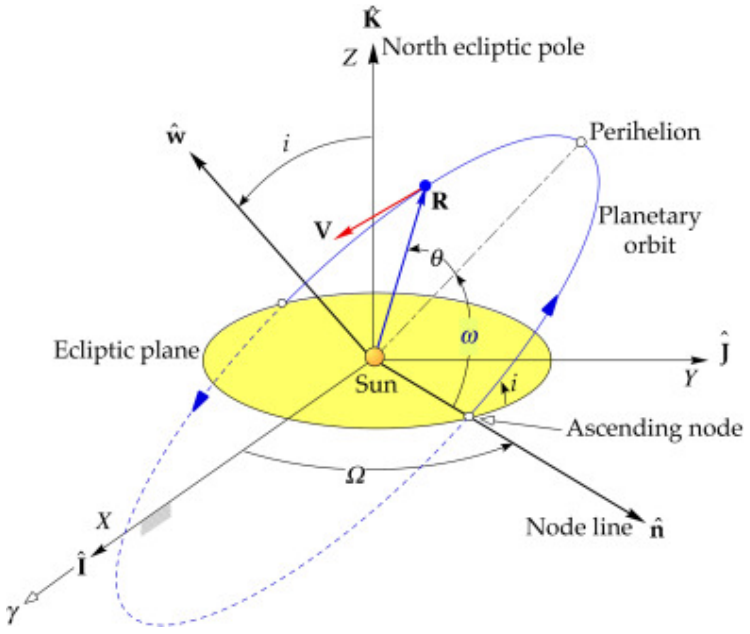

Figure 5.2: Planetary orbit in heliocentric frame reference (from [1] Fig. 8.25)

The algorithm used to obtain the state vectors $\vec{R}$ and $\vec{V}$ are described step-by-step on NASA website in [25]. Moreover, as the implementation of these steps have been covered by [3], they will not be repeated here.

5.3.2. Interplanetary Transfer Orbit - Lambert Solver

At this stage, the state vectors $\vec{R}$ and $\vec{V}$ of the two planets composing a transfer leg have been obtained, hence, the transfer orbit between them has to be determined, which is known as Lambert's problem. This problem was first posed by J. H. Lambert in the 18th century and states that the transfer time of a body moving between two points on a conic trajectory is solely dependent on the sum of the distances of the points from the origin of force, the linear distances between the points, and the semi-major axis of the conic [26]. While this is an assumption as we are ignoring the influence of all other objects in the solar system, for the purposes of this work, the perturbations caused by tertiary bodies are negligible.

Lambert's problem can be formulated as follows:

$$E = \frac{1}{2}V^2 - \frac{\mu}{r}, \tag{5.1}$$

where E represents the energy/mass of the body (called the specific orbital energy), r is the distance from the Sun to the center of the body, and V is the velocity of the body at r. By determining the time of flight T between two given points P_1 and P_2, Lambert's problem aims to compute the trajectory connecting these points. This trajectory is determined by finding the departure velocity from point P_1, as the position and velocity at any point along the orbit can be inferred from the initial values [1] Sect 5.3. Note that the planets are not confined on the same 2-dimensional ecliptic plane of Earth, but rather have a third dimension above/below the plane. Nonetheless, the interplanetary transfer orbit will still be confined in a 2-dimensional plane that usually differs from the ecliptic.

Moreover, the sign of the specific orbital energy E in Equation 5.1 also serves to classify the orbit as bound or unbound:

$$\begin{aligned} E < 0 &\rightarrow \text{Elliptical orbit (bound)} \\ E = 0 &\rightarrow \text{Parabolic orbit (unbound)} \\ E > 0 &\rightarrow \text{Hyperbolic orbit (unbound)} \end{aligned}$$

Equation 5.1 can also be used to determine the departure speed required for parabolic and hyperbolic transfer between orbits P_1 and P_2, as it is the formula for the escape velocity:

$$V_1 \geq \sqrt{\frac{2\mu}{r_1}}. \tag{5.2}$$

In this case, the equality is the velocity required for a parabolic transfer whereas any velocity above is the excess hyperbolic velocity.

As this problem has already been established for centuries, in this work a Lambert solver from the European Space Agency (ESA) was implemented [27]. This solver, available in

the Pykep repository[1], utilizes advanced techniques for accurate and efficient computation of hyperbolic velocities v_1 and v_2 required to connect the initial points r_1 and r_2 within a specified time of flight T.

The solver implementation involves the following key components:

- Selection of an iterative variable to invert the time of flight curve.
- Iteration method for solving the problem.
- Initial guess to start the iteration process.
- Reconstruction methodology to compute v_1 and v_2 from the iteration results.

5.3.3. Patched conic

Now that the position of the planets at the fly by time are known and the transfers between are computed the trajectory segments need to be connected. More precisely, the arrival velocities from the transfer orbits to the departure velocities obtained from the Lambert solver need to be matched.

To accomplish this, we employ the patched conic approximation meaning that within the sphere of influence of each planet, the spacecraft follows a hyperbolic planetocentric trajectory and the influence of the Sun's gravitational pull is neglected during this phase [28]. The matching process yields the difference in velocity from incoming and outgoing which will be the delta-v impulse, required to connect the trajectory segments (since the velocities are with respect to the planet, applying no thrust would entail no difference in delta-v in the frame of the planet due to conservation of energy). It is important to note that the delta-v maneuver in this work is performed at the perigee passage, where it is most efficient. Nonetheless, there might be additional delta-v impulses done in one of the trajectory segments, known as deep space maneuvers which are not included in this work. In fact, the Cassini made use of deep space maneuver twice in between the Earth-Venus and Jupiter-Saturn legs [29].

To match the incoming and outgoing velocities, we find the required turning angle and perigee radius. We begin by converting the velocities relative to the planet, considering the heliocentric velocities provided by the Lambert solution and the planet's velocity [30].

$$\vec{v}_{\infty-in} = \vec{V}_{in} - \vec{V}_p \tag{5.3}$$

$$\vec{v}_{\infty-out} = \vec{V}_{out} - \vec{V}_p, \tag{5.4}$$

where $\vec{v}_\infty$ are the incoming and outgoing velocities in the planet's frame, $\vec{V}$ the heliocentric velocities respectively and $\vec{V}_p$ the planet's velocity.

The perigee radius of the hyperbolic trajectory can then be determined by computing the semi-major axes of the pre- and post-perigee passages a_{in} and a_{out} respectively [30] along

[1] https://github.com/esa/pykep/

with the respective eccentricities e_{in} and e_{out}.

$$r_p = a_{in}(1-e_{in}) = a_{out}(1-e_{out}), \tag{5.5}$$

After a couple more manipulation steps, for which the exact procedure can be found in [30] and in [3], the final expression for the delta-v thrust required at the perigee is:

$$\Delta V = \left| \sqrt{v_{\infty -in}^2 + \frac{2\mu_p}{r_p}} - \sqrt{v_{\infty -out}^2 + \frac{2\mu_p}{r_p}} \right|. \tag{5.6}$$

It is important to note that the perigee radius and turning angle are unconstrained, meaning that in some cases, they may lead to physically impossible configurations, such as the perigee radius passing through the planet's interior. After the full trajectory is calculated, these cases are handled in the algorithm using a penalty function.

Having completed these calculations the entire trajectory of an individual is now well defined and all the required values are available to be used as inputs for the genetic algorithms. With each set of dates defining events in the trajectory, the full trajectory can be computed, allowing for the determination of the total delta-v required. This total delta-v includes the cumulative sum of the departure delta-v and the delta-v for each flyby.

5.4. Genetic algorithm

This section provides a detailed explanation of the genetic algorithm part of the code used in this study shown in Figure 5.1.

5.4.1. Objective function

Unlike previously where the objective of the program was to minimize the delta-v cost for an interplanetary trajectory with multiple gravity assist maneuvers, now we are looking for a more nuanced solution where we also consider time as part of the equation as previously shown in Section 1:

$$C = f(\vec{X}) + g(\vec{X}) + t(\vec{X}) \tag{5.7}$$

The objective function in this case is the sum of $f(\vec{X})$, which represents the sum of the departure and flyby delta-v values, and $t(\vec{X})$, which is a measure of the time taken to the final destination. Both of these terms are a function of a vector of time events $\vec{X} = [t_0, t_1, \ldots, t_n]$, where t_0 is the departure date and t_n the date of the n^{th} flyby. C is the objective function that we aim to minimize using the evolution of the individuals in the population through the genetic algorithm. Since, the function $f(\vec{X})$ is just the sum of the departure and flybys delta-v, it is already well defined:

$$f(\vec{X}) = \Delta_{dep}(t_0) + \Delta_{fb}(t_1) + \cdots + \Delta_{fb}(t_n). \tag{5.8}$$

where $\Delta_{dep}(t_0)$ is the initial delta-v cost of the departure.

In order to be able to compare and weigh an arrival time to a total delta-v cost of a mission several steps had to be completed.

Firstly, it has to be decided how time is taken into consideration as there are multiple ways that can be done. In this work, the 'quantity' of time that will be considered is the days taken to the final destination from the first date of the departure window. This is done to assist with the interests of space mission development phases, in particular, the aspect of having a satellite ready but the code for example finding a trajectory that only departs over half a decade after the satellite is constructed. Therefore, only having a short time of flight is not a good measure of time as that flight may be scheduled deep into the future. Hence, using the number of days since the first day of the departure window was chosen to be able to target a specific arrival date with the right calibration of β, which is also what is important to mission designers beyond having a low thrust consumption as mentioned in Section 3.4.. The calculation of the time taken since the first day of the departure window is simple, due to the way it has been implemented in the algorithm, which is simple the sum of the $\vec{X} = [t_0, t_1, \ldots, t_n]$:

$$t_{total} = \sum_{i=0}^{n} (t_i) \tag{5.9}$$

where t_{total} is the number of days since the start of the departure window.

Secondly, once an adeqaute way of measuring time has been established it needs to be non-dimensionalized appropriately. Since the proper way to compare time with delta-v in a meaningful way, the two terms have to be of similar magnitude which also allows α and β to be in more palatable dimensions between 0 and 1. While this redimensionalization is the most challenging part, it is worth noting that there are multiple ways to do it and there is no single correct answer as it can be adjusted with the prioritizing weights α and β later. In this work, the sum of the mean time of each time interval was used to make the time term dimensionless:

$$\sum_{i=0}^{n} \text{mean } (T_i) \tag{5.10}$$

where T_i is the i^{th} time window and example may be as follows:

$$T_i = [250, 750] \rightarrow \text{mean } T_i = \frac{750 + 250}{2} = 500 \tag{5.11}$$

here the units of time intervals are in days. Now, the term can become dimensionless through simple division:

$$\Pi_{time} = \frac{t_{total}}{\sum_{i=0}^{n} \text{mean } (T_i)} \tag{5.12}$$

Here, the Π_{time} term is the dimensionless time component, which was named in reference to the Buckingham Pi theorem that revolves around creating groups of non-dimensional variables (CITE Dimensional Analysis and Numerical Experiments for a Rotating Disc or Buckingham pdf)

Thirdly, as the unitless term has to be compared to the delta-v term, which is in units of m/s, Π_{time} will be multiplied with the value of the missions minimum delta-v cost, V_{min}. That is to say that first the code will be run using $\alpha = 1$ and $\beta = 0$ to have a delta-v quantity

of adequate proportion. This is important since using just a fixed delta-v scaling parameter for all different types of missions will not have the same scaling effect as some missions require in the order $7000m/s$ while others require almost $15000m/s$ as will be seen in the following sections of the thesis. Therefore, this ensures that the scaling of Π_{time} to a "delta-v-like" term is tailored for each specific mission trajectory. The final term that makes up the second part of the objective function, $t(\vec{X})$, is now well defined as follows:

$$t(\vec{X}) = V_{min}\Pi_{time} = V_{min}\frac{\sum_{i=0}^{n}(t_i)}{\sum_{i=0}^{n}\text{mean }(T_i)} \tag{5.13}$$

It is important to note that V_{min} is not required to be the minimum possible delta-v trajectory, instead, it suffices to be close to it which means the code can be run for a smaller number of generations in a shorter time frame. Hence, now Equation 1.4 can be presented in more detail as:

$$C = \alpha \cdot \left(\Delta_{dep}(t_0) + \Delta_{fb}(t_1) + \cdots + \Delta_{fb}(t_n) + penalty\right) + \beta \cdot \left(V_{min} * \frac{\sum_{i=0}^{n}(t_i)}{\sum_{i=0}^{n}\text{mean }(T_i)}\right) \tag{5.14}$$

Moreover, to make this interplanetary design tool particularly user friendly, the prioritization variables α and $beta$ are related as follows:

$$1 = \alpha + \beta \tag{5.15}$$

Hence, the two can be utilized in a practical fashion by assigning them percentages that add up to 100%.

5.4.2. Trajectory constraints

At this point, the penalty term, $g(\vec{X})$, needs to be addressed as it serves to eliminate non-physical solutions by making the affected individuals appear extremely unfit through the artificial inflation of the total delta-v. Two main constraints are considered.

The first constraint prevents physically unrealistic scenarios where the perigee radius passes through the interior of a planet during a gravity assist maneuver. To address this, a piece-wise function is employed, assigning a large constant penalty if the perigee radius is smaller than 110% of the planet's radius. The condition implemented in the code is when $r_p > 1.1r_{pl}$, where the penalty cost $g(\mathbf{x})$ is a large constant value of $10^{10}m/s$.

$$g(x) = \begin{cases} 0 & r_p \geq 1.1r_{pl} \\ 1e10 & r_p < 1.1r_{pl}. \end{cases}$$

The penalizing value is chosen very high on purpose, as the delta-v maneuver usually is in the order of 0 to 10 m/s for the optimal values and it even has a significant effect on the overall cost for cases where time is heavily prioritized over delta-v for example when $\alpha = 0.05$ and $\beta = 0.95$. This ensures that solutions with excessively low perigee radii are discarded in all cases. To further emphasize this point, the only case where the penalty will not be part of the cost function is in the 'invalid' case when $\alpha = 0$ and $\beta = 1$. However, this case when interpreted in terms of a mission design means getting to the final destination as fast as possible irrespective of the cost. In such a scenario gravity-assist trajectories

are not used but rather direct transfers to the final destination through a massive initial departure delta-v.

The second constraint prevents low-velocity flybys between the same planet, as it could result in the spacecraft being captured by the planet's gravity. This is for a similar reason to the first constraint as a planet's atmosphere which is usually not included in the radius of a planet, causes drag on the spacecraft and hence slowing it down. To enforce this constraint, a penalty function based on the excess energy (E) of the spacecraft after the flyby is used as follows (taken from [31]):

$$E = \frac{v_{\infty-in}^2}{2} - \frac{Gm_{pl}}{r_{SOI}}, \tag{5.16}$$

where

$$r_{SOI} \approx \left(\frac{m_{pl}}{m_{sun}}\right)^{\frac{2}{5}} r_{pl,sun}. \tag{5.17}$$

and the implemented penalty function is

$$g(x) = \begin{cases} 0 & E \geq 0 \\ \frac{1}{v_{\infty-in}} & E < 0. \end{cases}$$

A penalty is applied to the solution if the excess energy is negative, as it indicates a potential capture. This penalty guarantees that the spacecraft's velocity remains sufficiently high to avoid capture. It is also worth noting that the sphere of influence is assumed to be 'spherical' which is technically not the case as the far side of the planet experiences a weaker gravitational pull from the Sun compared to the Sun-facing side. However, this effect is minor and is therefore neglected in this work.

Now, individuals violating either of these constraints receive high cost and fitness values, making them unlikely to be selected for reproduction and crossover in the genetic algorithm. Therefore, over time these individuals that represent non-physical trajectories will be driven out of the population.

5.4.3. Genetic evolution structure

The genetic algorithm is defined by the various parameters discussed in Section4.1.. The main ones are the population size and number of generations but also important are the operator methods, and probabilities. These values can be adjusted to modify the algorithm's behavior, and changing the population size and number of generations significantly affects the output. The structure of the genetic algorithm used in this study is illustrated in Figure 5.3 [3].

The algorithm begins with population initialization where each individual is represented by a list of time events:

$$\vec{X} = [t_0, t_1, \ldots, t_n]. \tag{5.18}$$

For each leg of the trajectory, there is a prespecified minimum and maximum time. To increase the input resolution, the dates are cumulative and converted to real dates using the provided bounds. In the scope of this work Bayesian Julian Date convention was used

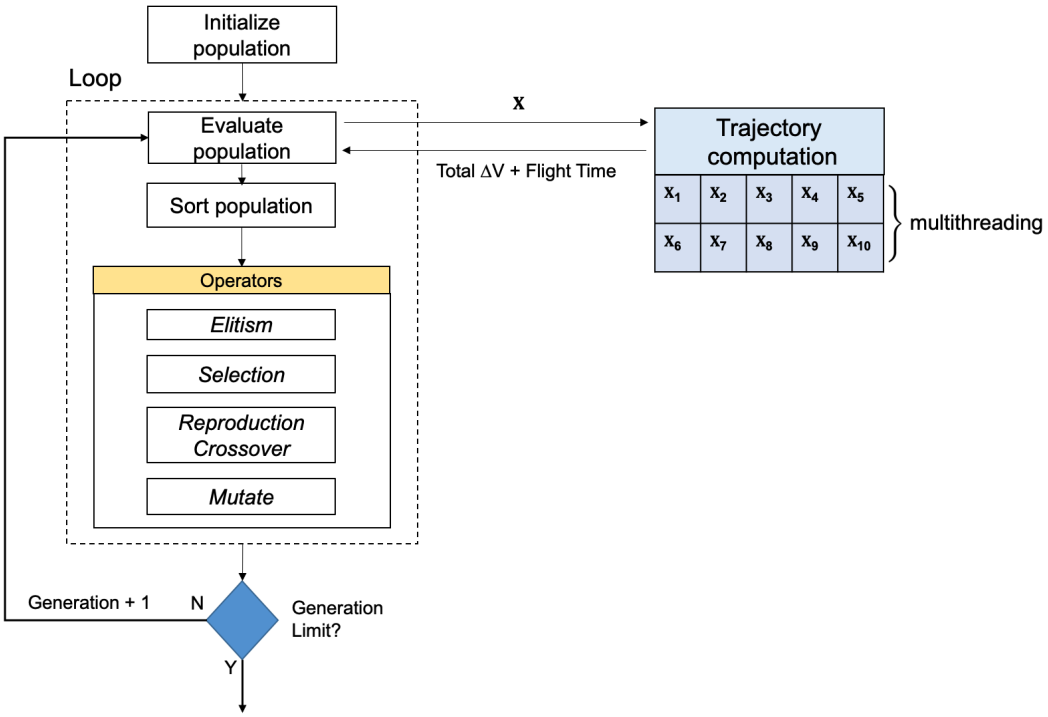

Figure 5.3: Genetic algorithm diagram

to also match the ephemeris data seen in Table 5.1 before. One example for the time boundaries for t_0 (the departure) date is:

$$T_0 = [2440587.5 \leq t_0 \leq 2441317.5] \tag{5.19}$$

where T_0 is a range of two calendar dates. Note that for T_1 to T_n the time ranges are days, as shown in Equation 5.11 and not dates. Nonetheless, in the Bayesian Julian calendar system you can subtract the two dates to obtain the number of days between. In the case above that would be:

$$\delta t_0 = 2441317.5 - 2440587.5 = 730 \tag{5.20}$$

which means the departure window δt_0 starts on 2440587.5 (which represents January 1st, 1970) and is open for 2 years (730 days). For t_0, a random value between [0,730] will be chosen and added to the first date of the arrival window. For the other time events, a random number within the time range is assigned to each t_i in the vector. To calculate the date of the flyby, required to compute the ephemeris, the following equation is used:

$$t_n = t_{0,min} + \sum_{i=0}^{n} t_i. \tag{5.21}$$

These t_is which compose $\vec{X}$ are then used to calculate the trajectory. The first step after initialization (from the genetic algorithm perspective) is therefore the evaluation of the fitness of every individual in the population. Each individual's trajectory, encoded by their dates, is evaluated using the objective function (Equation 5.14) to derive a fitness value. Now, the fitness values are normalized, considering it is a minimization problem. First, we conduct a step to calculate the adjusted fitness of an individual:

$$a(i) = \frac{1}{1 + s(i)}, \tag{5.22}$$

where i is an individual with a raw fitness $s(i)$ and $a(i)$ is the adjusted fitness. The fitness will now lay between 0 and 1. Next, the fitness for the whole population is normalized such that the sum of all fitness values is 1:

$$n(i) = \frac{a(i)}{\sum_j^n a(j)}. \tag{5.23}$$

After this step, each individual i will have normalized fitness value $n(i)$ and the fittest individuals also have higher normalized fitness values. Now, the population is sorted from the fittest to the weakest individual using the inbuilt C++ $std :: sort$ function. In order to run a genetic algorithm the various time events need to be converted into a binary string to follow the procedures described in Section 4.1.. The detailed breakdown of the number of bits used for the integral and decimal part of a number can be found in [3]. The various genetic algorithm operators shown in the yellow box in Figure 5.3 are now applied:

1. As described before, a predefined number of ”elite” individual directly survives to the next generation in order to preserve the best genetic material of the current generation.

2. Selection is performed next, using one of the implemented methods chosen by the user. One of the specific selection methods discussed in Section 4.1.1.1. is chosen by the user at the start. Selected individuals are copied to the new population vector until it reaches its maximum size, which is the predefined population size N_{POP}.

3. Reproduction and crossover operations are then performed on the new population vector. Pairs of consecutive individuals are crossed using the user-chosen method, and the offspring replace their parents.

4. The last operator, mutation, is applied to each individual using the selected mutation method. Note that here the elite individuals are excluded in order to maintain them unchanged. Once completed, each new individual's trajectory is calculated again and the process recommences.

This loop continues until the maximum number of generations is reached and the program displays the exact trajectory details of the best trajectory that was found.

5.4.4. Multithreading

To enhance the algorithm's execution time, multithreading was implemented which allows parallel execution of independent operations that consume significant time. Choosing the code's language as C++ in order to allow for multithreading was also deliberate as other programming languages such as Python, which are perhaps easier to work with, do not support it.

For this thesis, multithreading is utilized by computing multiple (at the moment ten) trajectories in parallel instead of sequentially. This is down by splitting the population of individuals into ten groups of size $N_{POP}/10$, and solving each of those concurrently. As a means to also optimize the outer loop optimization mentioned in Section 1, multithreading has also been implement there for future application, to test different sequences at once.

Parallely computing is only feasible during the computation of the fitness because it depends solely on $\vec{X}$ and is an independent operation for each member of the population. However, multithreading is not applicable to other parts of the genetic algorithm, such as the crossover operation, which involves interdependencies among individuals and is time-consuming. The mutation operation, although independent, is already fast and inexpensive, so multithreading is not considered to avoid unnecessary code complexity. However, in the case of trying different sequences at once as the code is written for, then the interdependencies do not go beyond a single sequence, meaning that multithreading can indeed be used for the GA part of the algorithm. Note that the speed-up might be smaller in this application, as there is already multithreading used within each sequence and using even more threads is applicable only to computers with many cores that are designed to handle the high computational load at once.

The impact of multithreading is not noticeable for short trajectories or small population sizes. For missions with more complex trajectories involving multiple planets and a larger search space, such as Galileo or Voyager II, multithreading reduces execution time by approximately 50%. For these missions, the execution time for fitness computation and other operations was reduced from 7 to 8 minutes using a single thread to around 4 minutes with multithreading given the same parameters used for the genetic algorithm. The convergence to an optimal value depends mainly on the number of individuals and generations, and substantial gains and stability are achieved when both are sufficiently large for lengthy missions. Moreover, as the objective function has become more complex, the required dimensions of the number of generations and the population size in order to converge has also increased.

At this stage, it is tempting to create a relation looking at the time efficiency of a code to reach an optimal trajectory as a function of the total number of computations. This may look as follows:

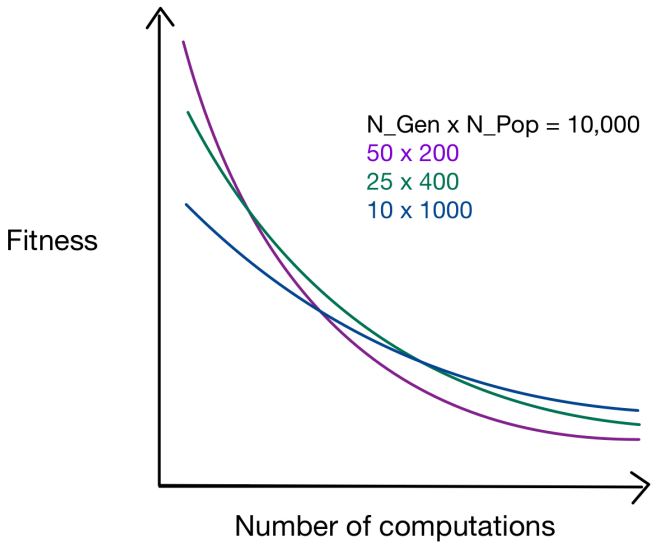

Figure 5.4: Fitness convergence as a function of computations

Here, the three curves all represent a total civilization of 10,000 individuals (counting up all the generations). One would expect that on average, when running each of the three variants (50x200, 25x400 and 10x1000) multiple times, the lowest starting fitness is likely

the one with most initializations, meaning 10x1000. Remember that since we aim to minimize the total delta-v quantity a lower fitness score is better in the illustration above. As the genetic algorithm evolves 50x200 the most, it is probable that it will find the best optimal trajectory after a high number of computations. However, not every computation has the same 'costs' in terms of the time taken to complete it. As mentioned before, for every evolution that occurs all the individuals are needed at the same time, meaning that only one computation can occur at a given moment. On the other hand, in the case of 10x1000 it may take more computations to reach the same level of fitness, but more of those computations can be done in parallel. As seen in Figure 5.3, the multithreading allows for the computation of 10 individuals' trajectories at once. Therefore, when looking at how fast the fitness convergences as a factor of time, it is possible for the optimal solution to be found faster using a higher ratio of N_{POP}/N_{GEN} given a fixed civilization size.

CHAPTER 6. TEST CASES AND RESULTS

The complete algorithm has undergone validation at various stages, specifically focusing on three key components: ephemeris, Lambert solver, and patched conic computations. The accuracy of the ephemeris and Lambert solver results was confirmed by comparing them with the Pykep library, a standard and easily verifiable source. Additionally, the patched conic computations were cross-checked with data from [30]. While the individual parts of the algorithm align with expected outcomes, it is important to acknowledge that the computation of the entire trajectory relies on the Lambert solver and flyby technique employed. Therefore, it should be noted that while our algorithm produces optimal solutions, they may differ from those obtained by other heuristic or deterministic trajectory optimizers.

As this thesis is built as an adaptation and extension to the previously constructed interplanetary trajectory design algorithm by [3], the test case trialed will aim to use the same or similar genetic algorithm parameters in order to achieve a holistic comparison between the real mission versus the minimum delta-v solution ($\alpha = 1$ case) versus a time prioritized trajectory (where $\beta \neq 0$). Moreover, as there is only limited data known on the exact flybys and delta-v injections performed by the real missions, the flyby dates are used as means to construct the equivalent delta-v costs using this thesis' algorithm. This means that instead of the actual delta-v used in for example Voyager-II, the delta-v calculated using the real flyby dates of the mission will be used to ensure a just comparison by converting the real data into the predictions of the deep space mission trajectory design tool presented in this work. That way a fair and direct comparison can be made across the three without an inherent bias in comparing 'observations' to 'predictions'.

6.1. Established scaling relationships

In this section, some of the analysis that was conducted in [3] is summarized to demonstrate the scrutiny that this algorithm has already gone through. Thanks to the previous work, it has already been established which genetic operators are considered to be favorable for selection, reproduction and mutation. Moreover, an analysis of the run time of the algorithm as well as the progression of the delta-v over the course of the evolutions have been studied and are summarized below.

6.1.1. Population and Generations

The population size and number of generations play a crucial role in optimizing interplanetary trajectories using genetic algorithms. Larger search spaces require bigger populations and more generations for effective evolution. Figure 6.1 compares the convergence of 20 algorithm solutions with population sizes of 1500 and 15000 for the first three time events. Those time events correspond to the optimal departure, first and second flyby dates after

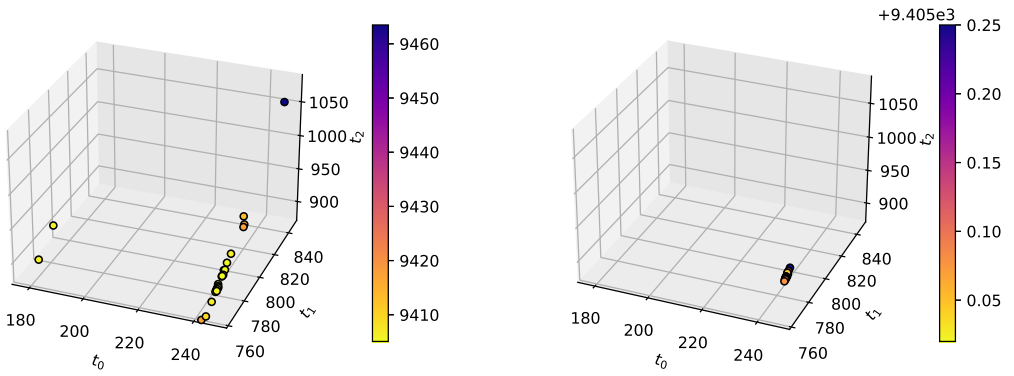

Figure 6.1: Comparison of 20 run solutions: 25 generations with 1500 individuals in the left and 15000 in the right [3]

evolving for 25 generations.

Moreover, if the population size is insufficient, convergence to a local minimum may occur. In such cases, solutions that reached a local minimum in early generations dominate the population, hindering diversity. To overcome this, larger populations and more generations are necessary to foster diversity and guide the population towards the global minimum. This can be seen in Figure 6.2, where small population may be stuck in a local minimum like the green or red line which are composed of 1,000 and 500 individuals respectively.

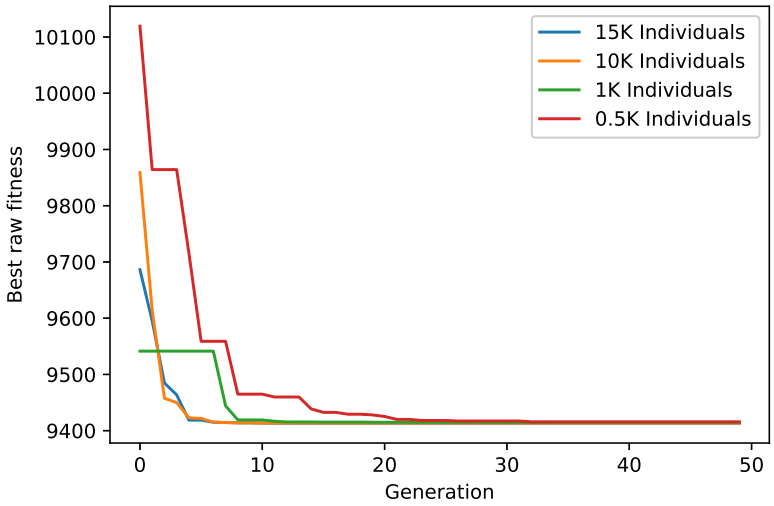

Figure 6.2: Convergence of Voyager-I's mission for different population sizes [3]

As can be seen in Figure 6.2, runs with 15000 to 10000 individuals find optimal values around the 10th generation, whereas a population of 500 individuals converges around the 30th generation. Naturally, the population size and number of generations also impact the algorithm's speed as will be discussed in further detail later.

6.1.2. Genetic Operators

The following genetic operators were tested in the optimization process in [3]:

- **Elitism**: A value of ten was set for every mission, meaning that only the fittest ten will survive from one generation to another. While its impact on overall performance is minimal with large populations as found by [3], it does have noticeable impact on the run time of the code as will be seen later.
- **Selection**: In order to understand which selection is more efficient a comparison of the two was made. Looking at Figure 6.3, the tournament selection was employed as

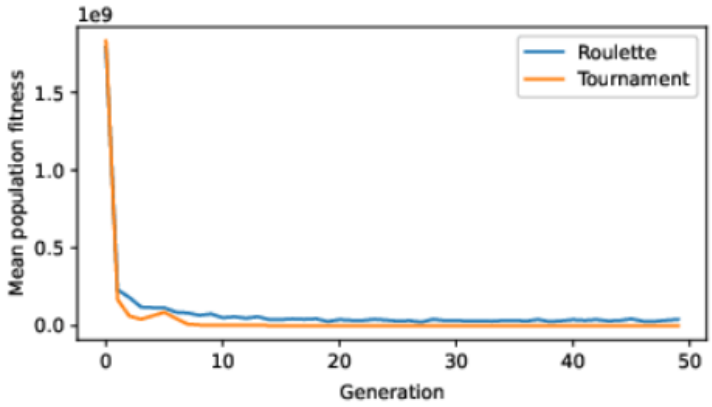

Figure 6.3: Comparison of selection methods for Voyager-I [3]

  the selection operator. This ”greedy” operator, as it always chooses the fitter of two individuals being compared, accelerates convergence. Due to the large populations used in the missions, tournament selection maintains diversity adequately.

- **Reproduction/Crossover**: Different crossover methods were compared in the optimization process and specifically their run time was analyzed. It was found that all the methods except Uniform crossover are quick and viable options.

6.2. Case Study - Voyager-II mission

6.2.1. Initial conditions overview

Voyager-II, the sister mission of Voyager-I, extended its trajectory beyond Saturn towards Uranus and Neptune, resulting in a larger search space and a greater number of potential solutions. To accommodate the increased search space, the population size and number of generations were increased to 50000 and 500, respectively. Although this leads to longer execution times, it ensures convergence to the optimal value and consistency across different runs. The GA parameters used for this test are the same ones used in [3] and are presented in Table 6.1. Moreover, the original time intervals for the various legs of the trajectory are presented in Table 6.2.

GA parameters	
Population	50000
Generations	500
Elitism	10
Selection	Tournament
Crossover	Double Point $(P = 0.9)$
Mutation	Flip Bit $(P = 0.2)$

Table 6.1: Genetic Algorithm Parameters Voyager-II

Trajectory parameters	
$T_{0,\,\text{min}}$	2443145
T_0 (Earth)	$01-01-1977$
T_1 (Jupiter)	$[0, 1095]$
T_2 (Saturn)	$[50, 2000]$
T_3 (Uranus)	$[500, 2500]$
T_4 (Neptune)	$[250, 2500]$

Table 6.2: Trajectory Parameters: Voyager-II

6.2.2. Performance comparison

A side-by-side comparison of the real mission data versus the minimum delta-trajectory versus the $\alpha = 0.7$ and $\beta = 0.3$ case is presented below to demonstrate the difference across the three:

As seen in Table 6.3, the lowest delta-v is of course the $\alpha = 1$ trajectory whose single objective in the fitness evaluation is the minimization of the delta-v. Naturally, this also comes with the longest time to reach Neptune being required out of the three cases since there is the least thrusting involved. In this case, the mission arrives over two years behind the mission that was actually flown and almost 2.5 years after the $\alpha = 0.7$ trajectory. While the real mission departed ten days before the $\alpha = 0.7$ case, it already required more initial departure delta-v and even took longer to reach Jupiter. This can be traced back to not conducting the most efficient interplanetary transfer and arriving at Jupiter over two months after the $\alpha = 0.7$ case.

In the next phase, the first flyby occurs around Jupiter which was similar in terms of the turning angle for the real and $\alpha = 0.7$ case, where both turned at around $\delta \approx 97^\circ$. This turn can also be seen in Figure 6.4, but note that this trajectory is a 2-dimensional representation of patched conics that are 3-dimensional when put together. Here, the incoming velocity of the real mission is slower than for $\alpha = 0.7$, which can be explained as it aims at a meeting with Jupiter further along in the planet's trajectory. Therefore, it has been going against the gravitational pull of the Sun for a longer period meaning work has been done on the spacecraft by the Sun to reduce its kinetic energy. This has occurred only to a lesser degree for in the $\alpha = 0.7$ trajectory, which is why it is seen to interact with Jupiter earlier in its path (further clockwise in Jupiter's trajectory as the planets move counterclockwise around the Sun) as seen in Figure 6.4.
Already in this phase, it is clear where a lot of the time will be gained with respect to the minimum ΔV case. While the departure velocity is only 4.7% higher when leaving Earth, the incoming velocity at Jupiter is over 40% larger when entering its sphere of influence. Moreover, the perigee passage of $\alpha = 0.7$ is less than half the distance compared to the $\alpha = 1$ case, at $r_p = 4.8026 \times 10^8 m$ as compared to $r_p = 1.1118 \times 10^9 m$ respectively. Nonetheless, as the two trajectories arrive at a different time and hence a different geometry of the planets, the delta-v thrust injected in this point is very small for both a factor of 57 times smaller than the real mission for the same flyby planet.

By the time the flyby around Saturn occurs in the different scenarios, there is already more

Trajectory Result			
	Real	minimum ΔV ($\alpha = 1$)	ΔV and Time Tradeoff ($\alpha = 0.7$)
Earth Departure			
	$20-08-1977$	$04-09-1977$	$01-09-1977$
Date T_0	2443375.5	2443391	2443388
Departure $\Delta v[\,\mathrm{m/s}]$	10230.7	9413.34	9855.66
Jupiter Flyby			
	$09-07-1979$	$17-10-1979$	$03-05-1979$
Date T_1	2444063.5	2444163.5	2444000
$v_{\infty -in}[\,\mathrm{km/s}]$	7901.55	6563.52	9298.66
$v_{\infty -\text{ out}}\,[\mathrm{km/s}]$	7757.95	6559.38	9295.96
$\delta[\mathrm{deg}]$	96.9863	93.0884	97.7442
$r_p[\,\mathrm{m}]$	7.058×10^8	1.1118×10^9	4.8026×10^8
$\Delta v[\,\mathrm{m/s}]$	57.844	0.34356	1.01265
Saturn Flyby			
	$26-08-1981$	$24-04-1982$	$18-03-1981$
Date T_2	2444842.5	2445084.125	2444680
$v_{\infty -in}[\,\mathrm{km/s}]$	10790.6	8254.31	13088.7
$v_{\infty -\text{ out}}\,[\mathrm{km/s}]$	9052.45	8255.15	13090.4
$\delta[\mathrm{deg}]$	85.656	83.3468	85.269
$r_p[\,\mathrm{m}]$	2.18×10^8	2.805×10^8	1.05455×10^8
$\Delta v[\,\mathrm{m/s}]$	815.66	0.378272	0.758372
Uranus Flyby			
	$24-08-1986$	$26-05-1987$	$19-02-1985$
Date T_3	2446666.5	2446942.125	2446120
$v_{\infty -\text{ in}}\,[\,\mathrm{km/s}]$	12877.1	11900.2	17378.6
v_{∞}-out $[\,\mathrm{km/s}]$	17661.6	11900.2	17379.4
$\delta[\mathrm{deg}]$	22.164	18.1143	26.0872
$r_p[\,\mathrm{m}]$	9.21×10^7	2.583×10^8	7.76536×10^7
$\Delta v[\,\mathrm{m/s}]$	3728.92	0.025	0.654346
Neptune Arrival			
	$25-08-1989$	$15-09-1991$	$25-04-1989$
Date T_4	2447763.5	2448484.125	2447277
Arrival $V_{in}[\,\mathrm{m/s}]$	21551.0	15349.0	21862.2
Result			
Total cost $[\Delta v]$	**14830.12**	**9414.075**	**9856.34**

Table 6.3: Voyager-II Results Comparison

than a year's difference across them. Here, again the delta-v injections for the $\alpha = 0.7$ and $\alpha = 1$ is below $1m/s$, whereas for the real mission it is $\delta v = 815.66m/s$. The turning angle is similar for the real and $\alpha = 0.7$ trajectory as those are closer in date compared to the $\alpha = 1$ case, with $\delta = 85.656°$, $\delta = 85.269°$ and $\delta = 83.3468°$ respectively. There is also a stark difference noticeable in the incoming and outgoing velocity across the three cases. As expected the velocities can be sorted from lowest to highest as follows: $\alpha = 1 < real < \alpha = 0.7$. Interestingly, the velocity that $\alpha = 0.7$ carries is over 20% higher compared to the actual Voyager-II spacecraft.

This velocity differential also explains the increasing gap in arrival time for the final flyby around Uranus, as $\alpha = 0.7$ arrives six months ahead of the real mission. At this perigee passage, the real Voyager-II conducted a massive maneuver of $\Delta v = 3728.92m/s$, which of course is very expensive fuel-wise, but allows it to have a faster outgoing velocity compared to $\alpha = 0.7$. In fact, the outgoing speed differential between the two is $282.2m/s$, which means it is 'catching up' but at a high fuel cost.

Finally, the arrival at Neptune occurs and the results may be summarized as follows:

	$M/D/Y$	total ΔV$[\mathrm{m/s}]$	extra days*	% less thrust vs real
Voyager-II	$8/25/89$	14830.12	751	-
minimum ΔV	$9/15/91$	9414.075	-	-36.5%
Tradeoff	$4/25/89$	9856.34	873	-33.5%

Table 6.4: Voyager-II Results Summary; *the extra days at the final destination are calculated compared to the minimum ΔV solution

An interesting aspect that may be non-intuitive is that the $\alpha = 0.7$ mission arrives at Neptune at a higher incoming velocity than the real spacecraft, despite having left with a big velocity differential from Uranus' sphere of influence. However, this can be explained due to the different trajectories that the two cases take in order to reach the final destination as seen in Figure 6.4. In the case of the real mission the motion after departing from Uranus is much more radial with respect to the sun compared to its $\alpha = 0.7$ counterpart. This is also reflected by the larger turn angle that $\alpha = 0.7$ took compared to the real mission at $\delta = 26.0872°$ and $\delta = 22.164°$ respectively. As seen in Figure 6.4, the $\alpha = 0.7$ trajectory is more circularized with respect to the sun (meaning it moves more counterclockwise) compared to the real mission which has a higher eccentricity for its final transfer to Neptune (meaning it more work is done to reduce its kinetic energy as it moves more radially). This effect causes the real mission to arrive at a slower velocity, despite having cut the time differential from a maximum of six months down to four.

Overall, the implementation of time as a consideration has caused massive changes in the ultimate trajectory of the spacecraft. Using a balance of priorities distributed as 70% ($\alpha = 0.7$) on delta-v and 30% on time, a similar trajectory to the real mission flown, in terms of overall arrival time, has been designed. It is obvious that simply minimizing the overall delta-v and thereby having flybys with practically no thrust injection, leaves a lot of mission potential untapped and should, therefore, not be the design strategy of deep space missions.Moreover, the total ΔV cost in order to arrive four months ahead of the

real mission, and almost 2.5 years ahead of the minimum ΔV solution, is less than $2/3$ of the true Voyager-II ΔV requirement. Additionally, this massive increase in time available at the final destination compared to the results obtained in [3] comes only at an additional cost of about 4.7% of the minimum total ΔV.

The complete trajectories of all three cases are illustrated here:

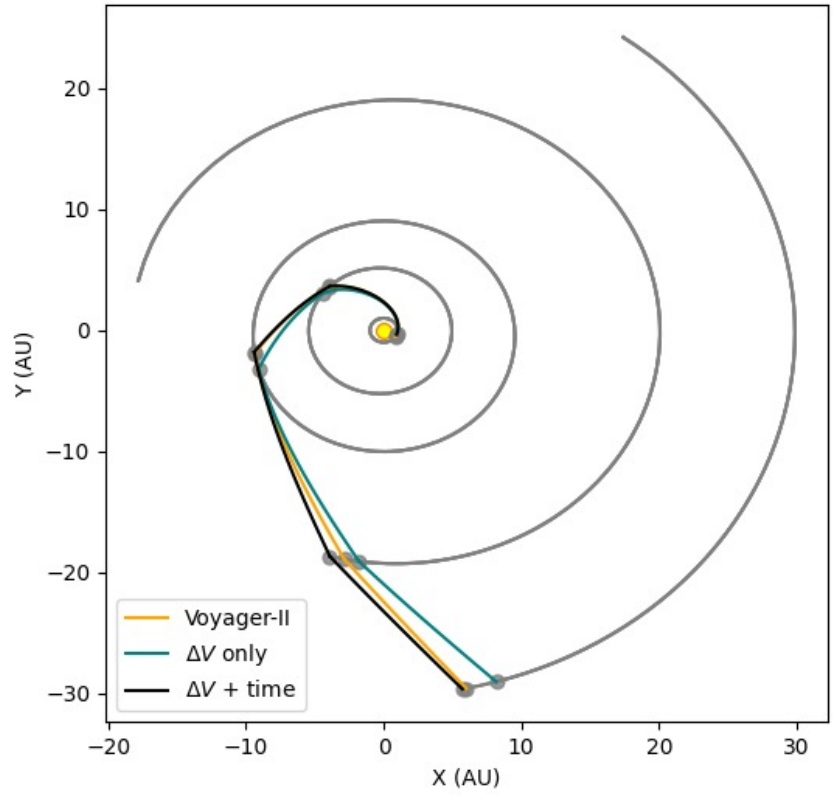

Figure 6.4: Voyager-II trajectories: real vs $\alpha = 1$ (minimum ΔV) vs $\alpha = 0.7$ (Tradeoff)

In Figure 6.4, the Earth's orbit is only the small gray circle with a radius of $1AU$ right around the Sun. The order of flyby planets is naturally Jupiter, Saturn, Uranus and finally Neptune, which are the circles with increasing radii illustrated above. As analyzed using the Table 6.3, the minimum ΔV trajectory can be seen to be the slowest at every stage as it arrives always further down counter-clockwise to every planet, as they circle around the Sun in that orientation. Here, the difference between the $\alpha = 0.7$ and the real mission can be most notably seen in the second and third flyby as it causes large differences in the arrival to the third and fourth planet respectively. The maximum time difference between the two of six months is also evident by the difference in where they arrive to Uranus (the second largest circle depicted). The massive delta-v injection in the real mission to reach Neptune and shorten the gap is also visible as the location of Neptune at the arrival of those two cases is almost the same. Note also the that the angle between the two cases for the final leg, is a reflection of the eccentricity of their conic around the Sun.

6.2.2.1. Speed Evolution

Finally, to get a complete picture of the gravity assists and their addition to the speed of the spacecraft at minimal fuel cost, a comparison of the minimum ΔV trajectory speed evolution, $\alpha = 1$, and the $\alpha = 0.7$ trajectory speed evolution is shown in Figure 6.5.

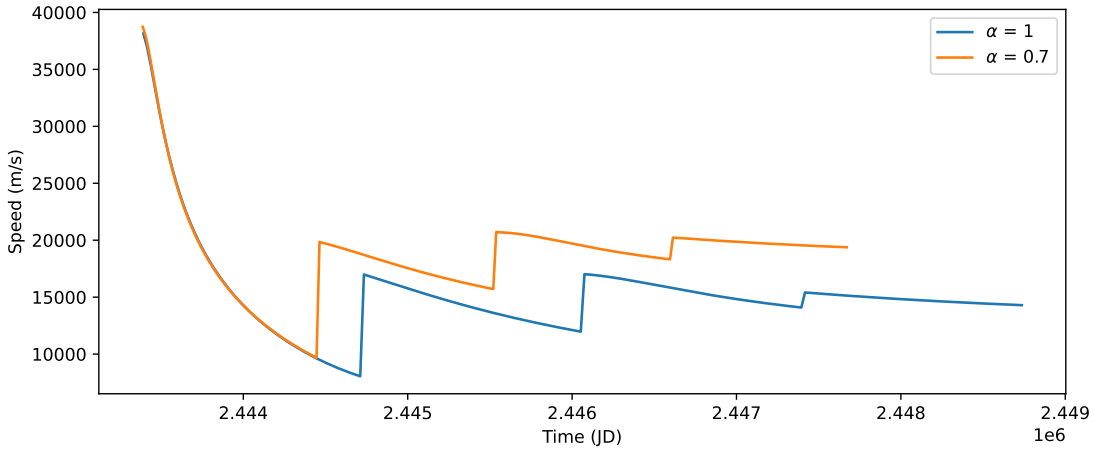

Figure 6.5: Speed Evolution: $\alpha = 1$ (minimum ΔV) vs $\alpha = 0.7$ (Tradeoff)

Each vertical spike in Figure 6.5 is the effect of the gravity assist allowing the spacecraft to utilize the energy of the planet (instead of its own fuel) in order to accelerate. It can be seen that the $\alpha = 1$ trajectory already prefers to depart with a slightly lower speed, which due to the large distance to Jupiter, accounts for a massive time difference in arrival to the first planet. As the $\alpha = 0.7$ trajectory arrives earlier at the final destination of Neptune, the orange line indicating its speed evolution also stops significantly before the blue line which represents the speed of the $\alpha = 1$ trajectory. The arrival speed at Neptune is also significantly higher for the $\alpha = 0.7$ trajectory, as can be seen by the vertical difference at the end of the orange and blue line in Figure 6.5. However, using the right entry angle, atmospheric drag can be used (without spending fuel) to slow down the spacecraft if needed. All of this additional time was gained using just 4.7% additional fuel across the two, which is a very small fraction of the delta-v cost of the real mission flown.

6.2.3. Sensitivity of weighting parameters

While prior to this, the only cases analyzed have been $\alpha = 0.7$ and $\alpha = 1$ in the context of the Voyager-II mission, now a spectrum of αs will be analyzed to study the effect on the optimal trajectory found.

To remain familiar with the overall mission this analysis is also conducted using the Voyager-II mission as the trial case. A range of $\alpha = [1.0, 0.5]$ has been used in conjunction with the same GA parameter and trajectory parameters found in Table 6.1 and Table 6.2 in order to complete this analysis. A particular focus is given to the number of days prior to the arrival of the minimum delta-v solution as well as the additional fuel spent as those are key factors taken into account by mission designers. The results in Table 6.5 were obtained.

alpha	Date	total ΔV	extra days	additional fuel %
1.0	4/28/91	9415.35	-	-
0.9	9/16/90	9433.63	224	0.19%
0.8	2/25/90	9482.86	427	0.72%
0.7	4/25/89	9856.34	733	4.68%
0.6	8/31/87	10136.6	1336	7.66%
0.5	5/1/87	10533.5	1458	11.88%

Table 6.5: Overall trajectory performance at different levels of α

Using these results, two plots are created to visualize the evolution of the trajectory as a function of α.

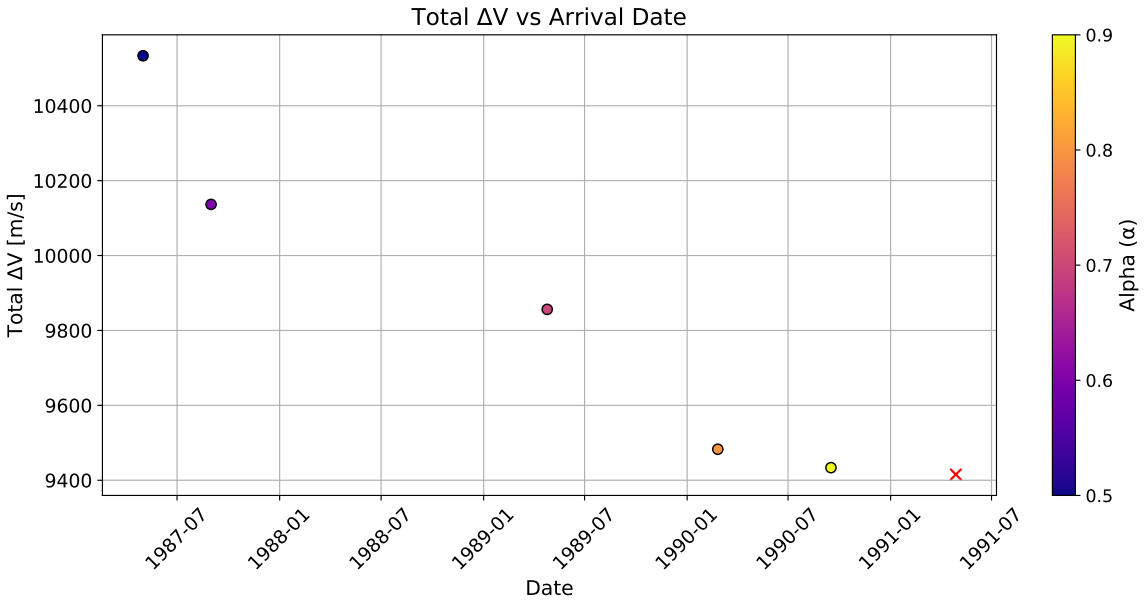

Figure 6.6: Total Delta-V vs Arrival Date

In Figure 6.6, the plot "Total Delta-V vs Arrival Date" shows the relationship between the total delta-v and the arrival date for different alpha values. The color map represents the alpha values where darker colors indicate smaller alpha values. As shown in the figure, a lower alpha value indicates a lower prioritization of minimizing the delta-v (higher prioritization of an earlier time of arrival), resulting in a higher total delta-v requirement.

Figure 6.7 displays the "Trade-off between Extra Days and Fuel Cost." which illustrates the relationship between the extra days spent at the final destination and the extra fuel cost relative to the minimum delta-v for different alpha values. This is done to show how the use of marginal extra fuel can lead to significantly earlier arrival dates at the final destination. As seen in Figure 6.7, a lower alpha value represents a higher priority to reaching the destination earlier, resulting in increased extra fuel costs. It can be seen that the $\alpha = 0.7$ case is slightly above the general trend which shows that there full convergence is not yet achieved, as there is no physical reason for the trend not to be smooth and monotonic. This will be more evident in Figure 6.8, where different fitting curves are applied.

These figures demonstrate the trade-off between the time of arrival and the delta-v cost

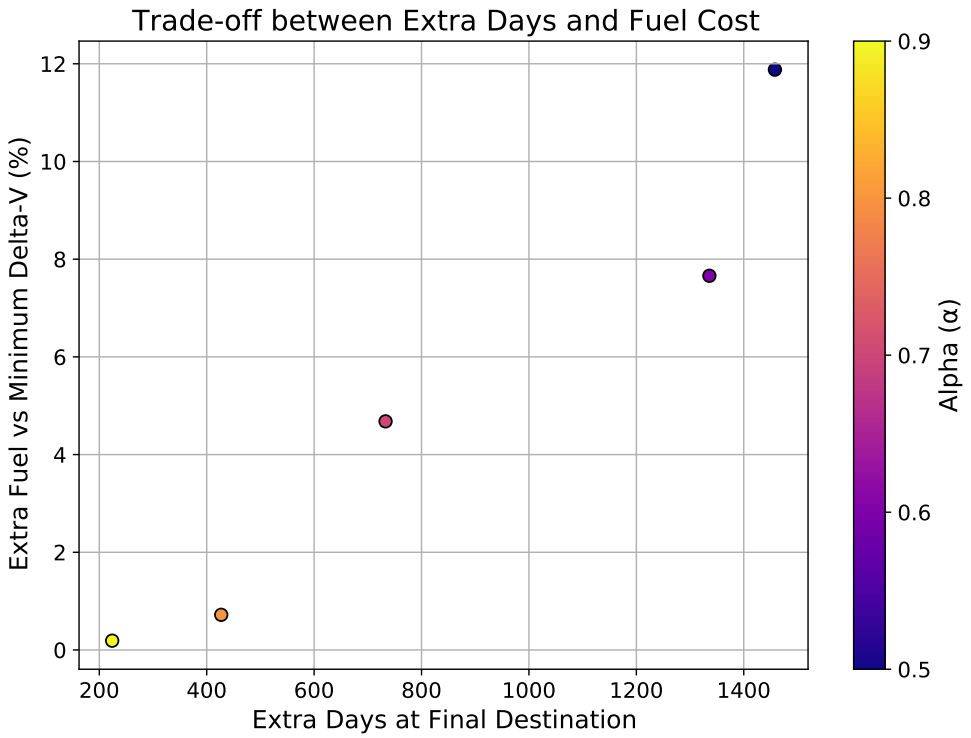

Figure 6.7: Trade-off between Extra Days and Fuel Cost

in space mission planning. By assigning lower alpha values, spacecraft can prioritize an earlier time of arrival but at the expense of increased fuel consumption. Mission planners must carefully consider this trade-off to achieve scientific objectives within resource constraints. In fact, when looking at a potential linear and quadratic fit, there seems to be a point of diminishing returns as illustrated by the blue curve in Figure 6.8. However, at what stage increasing the prioritization weight of β has diminishing returns is a subjective matter that each mission designer has to consider.

However, since both correlations are considerably strong, as seen by the respective R^2 values of $R^2_{quadratic} = 0.940$ and $R^2_{linear} = 0.936$, this trend of diminishing returns should be analyzed with caution. Moreover, as all this data has been generated using the GA parameters from Table 6.1 before in order to match the same inputs used by [3], in reality, the algorithm with the increased complexity in the objective function is not as stable when $\alpha \neq 1$ as compared to when $\alpha = 1$. This is to say that while a population size and number of generations of 50000 and 500 respectively were sufficient to continuously converge to practically the same optimal trajectory when only minimizing delta-v, in the case where time is also considered the parameters of the genetic algorithm need to be further increased.

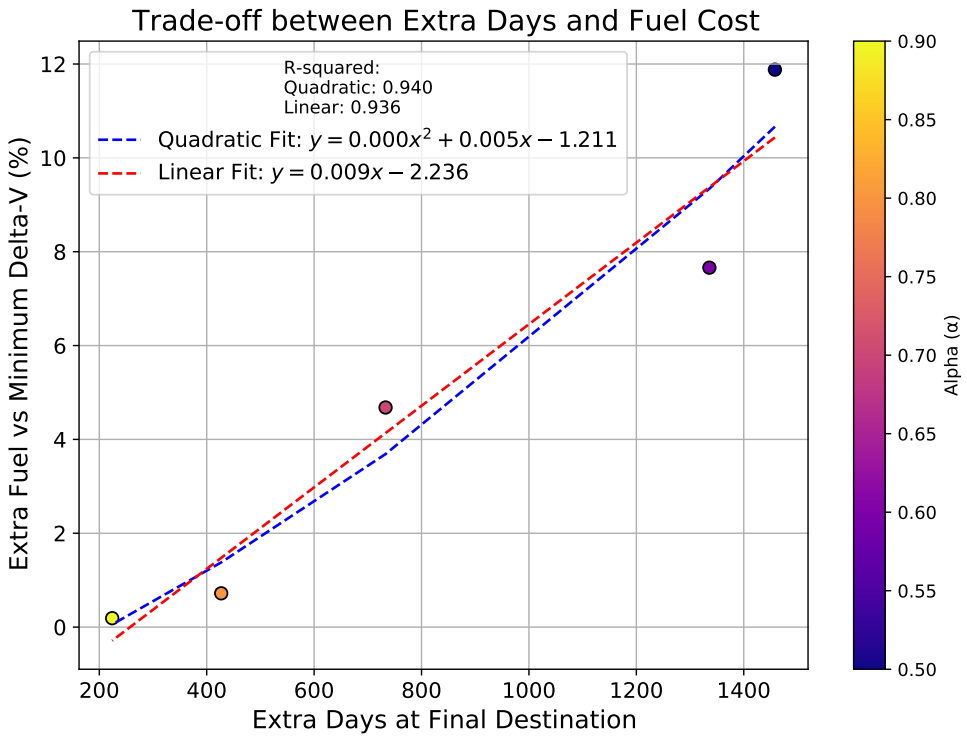

Figure 6.8: Linear vs Quadratic best fit

6.3. Analysis of genetic algorithm behaviour

6.3.1. Convergence of population

The following analysis will consist in studying the stability of convergence in the case for $\alpha = 0.7$, which is the value used to achieve a similar arrival date to the real Voyager-II mission given the aforementioned algorithm inputs. This time around the size of N_{POP}, N_{GEN} and elitism will be increased and the spread of the optimal trajectory arrival date of different runs will be studied.

Seeing the different spread of arrival times, it can be seen that the trend of earlier arrival time to increasing delta-v cost appears to be linear when looking at a short time frame of arrivals. As expected the higher variety of results comes from the runs with a smaller civilization size, that is roughly $N_{POP} \times N_{GEN}$. Clearly, this convergence, which is achieved at 50,000 individuals and 50 generations for the minimum delta-v trajectory, is not present using those GA parameters. This is related to the increased complexity of the objective function detailed in Equation 5.14. While on average the runs with higher population size and larger number of evolutions are more clumped up, some of the lowest delta-v has been found even using just 50,000 individuals and 200 generations.

It is worth noting that since this is a mission that departs on 1/9/1977, having an arrival window of several months in 1980 is not that large considering the overall journey is well over a decade long. Moreover using 75,000 individuals and 500 generations the range of arrival months was just from mid-May of 88' to August of 88'. Interestingly, when increasing

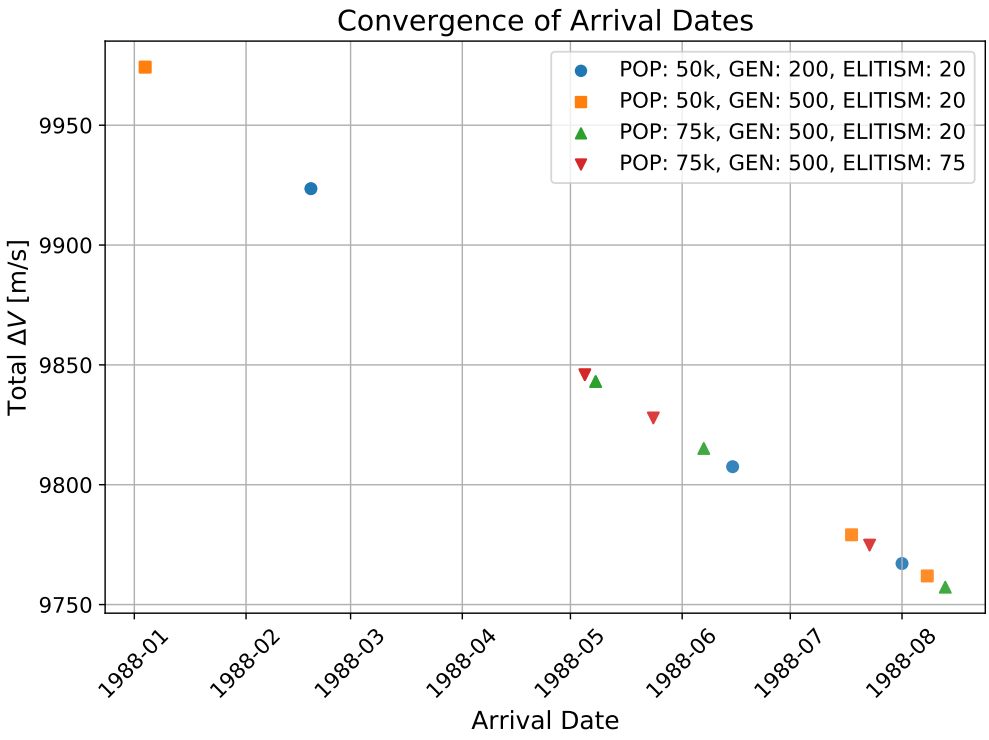

Figure 6.9: Arrival Date Convergence at different N_{POP}, N_{GEN} and elitism sizes

the number of elite individuals which are maintained across generations, the arrival window is shrunk further from May to July, which is less than 2% of the overall travel time. The specific range of delta-v required for the mission given different GA parameters is summed up below:

Population	Generations	Elitism	Delta-V Range (m/s)
50k	200	20	[9767.13, 9923.52]
50k	500	20	[9757.26, 9974.25]
75k	500	20	[9755.04, 9843.07]
75k	500	75	[9774.80, 9845.87]

Table 6.6: Delta-V Range for Different GA Parameter Set-ups

Moreover, changing the number of elites has another effect beyond simply shrinking the arrival window. As seen in Figure 6.10 as the number of elites is increased the run time required to complete the algorithm is reduced. This is because a significant lower number of computations are conducted as the elite individuals are always simply copied to the next generation.

Here, it is also seen that the scaling of the number of generations and the run time to complete the algorithm is approximately linear (see first and second bar of Figure 6.10).

Moreover, it is noticeable that increasing the number of individuals in a population increases the run time at a rate faster than linear (see second and third bar of Figure 6.10). However, this can be deduced logically as with increasing population size the algorithm not only has to spend more time creating new individuals through reproduction etc., but also

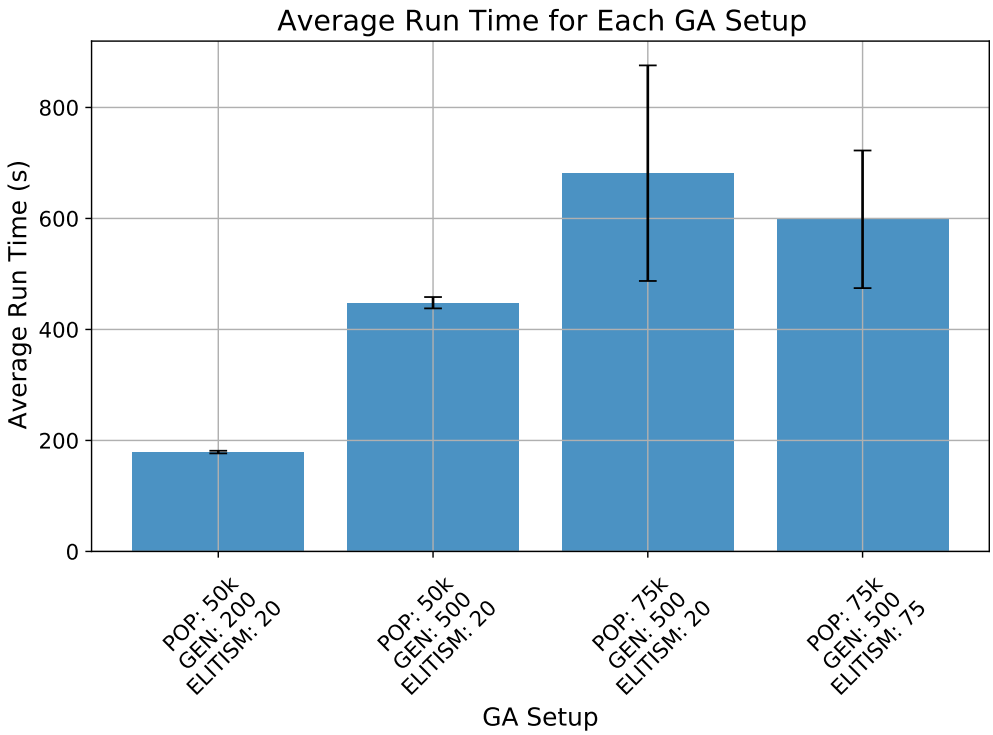

Figure 6.10: Algorithm run time at different N_{POP}, N_{GEN} and elitism sizes

has to compute many more trajectories which are computationally expensive (this is per generation). On the other hand, increasing the number of generations does not increase the number of trajectories that need to be calculated in each generation. Therefore, the run time is increased more when scaling up the population size.

Nonetheless, considering the complexity of the Voyager-II mission, the convergence to an optimal solution given a user-defined α can be achieved with significantly more ease in missions that do not require as many gravity assists or take as long.

CHAPTER 7. CONCLUSION

In this master thesis, an alternative approach to optimizing multi-gravity assist (MGA) interplanetary trajectories was investigated and developed. Previous methods focused solely on minimizing delta-v without considering time, whereas this study incorporates time as a factor in the objective function. This is of particular importance as merely optimizing delta-v has shown to prolong the time taken to reach the final destination by up to several years. As this can be considered a valuable time for space agencies and mission designers, finding a more nuanced solution considering the arrival time entails an improved mission design approach. Through the utilization of evolutionary algorithms, particularly genetic algorithms, the effectiveness of this approach in finding near-optimal solutions for interplanetary trajectories was demonstrated.

By extensively validating and testing against the Voyager-II mission (as well as Voyager 1 for the $\alpha = 1$ case), which followed an MGA trajectory, it was shown that this approach can generate trajectories that closely match the expected results and convergence criteria. Notably, the choice of genetic operators had minimal impact on the algorithm's performance, while the population size and number of generations emerged as critical factors influencing the outcomes. With respect to run time, the population size, number of generations as well as elites was found to be relevant. Moreover, selecting large values for these compared to the previous study [3] was necessary to ensure convergence to the optimal solution and avoid local minima.

Overall, the total ΔV cost in order to arrive four months ahead of the real mission, and over two years ahead of the minimum ΔV solution, was found to be only $2/3$ of the true Voyager-II ΔV requirement. The comparison of trajectories for flybys at Jupiter, Saturn, and Uranus reveals the significant impact of trajectory choices (different levels of α) on arrival times, delta-v injections, and spacecraft velocities. While deviations from the actual Voyager-II trajectory result in variations in arrival times, it is shown that even small changes in departure and flyby conditions can lead to substantial differences in trajectory outcomes. Additionally, the massive increase in time available at the final destination compared to the results obtained in [3] comes only at an additional cost of about 4.7% of the minimum total ΔV.

At this stage, it is clear that a willingness to spend more on fuel to push the spacecraft with a higher delta-v can have significant returns and is "well worth the time". Therefore, the objectives that were discussed in Section 2 have been achieved with the expected outcome that tradeoffs in terms of small additional delta-v costs can bring significant advantages in terms of the overall mission profile by arriving at a much earlier date As mentioned before, this new capability to target a particular arrival window allows users to plan their arrival and thus design missions that take advantage of the local geometry such as Europa-Clipper. In many cases, such as the Rosetta mission the time of intercept of an asteroid also has to be planned precisely which cannot be made when solely focusing on minimizing delta-v. Either way, additional time at the target destination, along with a reduced travel time, implies that data can be collected earlier as well as for an extended period since the lifetime of the spacecraft at the destination is prolonged. Therefore, this mission design tool may

potentially permit a change in the structural composition of future satellite missions as the priority on shielding and batteries, which are required during the interplanetary journey, can be reduced.

By incorporating time as a factor in trajectory optimization, it becomes possible to explore trajectories that minimize delta-v while meeting specific mission duration requirements. This allows for the identification of trade-off solutions that achieve a balance between fuel efficiency and shorter travel durations, in the interest of improving the overall mission in terms of scientific potential and economic investment.

7.1. Future work

While this master thesis has made significant progress in optimizing interplanetary trajectories using genetic algorithms, there are several avenues for future research and expansion of the work.

One interesting area of exploration is the potential to utilize multiple gravity assists around the same planet by leveraging the planet's moons. By considering the gravitational influence of a planet's moons, it becomes possible to perform sequential gravity assists, utilizing each moon's gravitational field to further optimize the trajectory. However, it is important to note that the moons typically have smaller masses compared to the planet itself. This would also require a high-fidelity simulation as the positions of the moons needs to be known with very good precision. While they can contribute to trajectory optimization, other factors such as the perturbations caused by the planet (e.g., Jupiter) in the interplanetary trajectory are likely to have a more significant impact. This would imply converting the interplanetary transfer from a 2-body to a 3-body problem, making it considerably more complicated while becoming more accurate.

From a technical/computational standpoint, the multithreading aspect of this tool can be expanded in order to further shorten the execution time. More details on how the GPU processing may be used are also discussed in [3].

As mentioned in Section 1, the use of Deep Space Maneuvers (DSMs) also present an interesting addition to interplanetary trajectory optimization. The inclusion of DSMs in trajectory design models offers increased flexibility and optimization possibilities to the algorithm. DSMs involve performing delta-v maneuvers at any points in the trajectory, not just at the perigee passage as it is currently implemented in this work. This allows for adjustments and corrections to be made along various legs of the trajectory. By enabling delta-v maneuvers at any point in the trajectory, the algorithm can explore a broader search space and potentially discover more efficient trajectories with improved mission performance and fuel efficiency.

Probably the most applicable and beneficial add-on to the current algorithm would be the optimization of the flyby sequence itself, which can be accomplished using the same genetic algorithm framework described in previous sections. Optimizing the flyby sequence

entails finding the most advantageous order and timing of planetary encounters to minimize delta-v and mission duration. By incorporating the flyby sequence optimization into the genetic algorithm framework, more efficient overall trajectories can be discovered, which can lead to even greater reductions in delta-v and mission duration as shown in Table 1.1.

To implement flyby sequence optimization, the genetic algorithm would be first applied on the outer loop, where each individual represents a flyby sequence. Then each 'outer individual' will then run itself a genetic algorithm of 'inner individual's, which are described using $\vec{X}$ as was seen throughout this thesis. Since the outer loop genetic algorithm continuously generates inner loop genetic algorithms, the parameters used for the outer and inner GA operators will be different. In fact, the backbone to implement this new and improved algorithm has already been coded for the most part, but was not included in the scope of this work to contain the scope the work of this thesis as mentioned in Section 2. Once completed, this makes for a user-friendly end-to-end deep space mission trajectory design tool, which only requires a final destination as an input along with a prioritization of delta-v vs time through the parameters α and β. It is also worth noting that a future study shall also take a deeper dive into analyzing the relationship between the complexity of the objective function and the resultant number of generations and population size required to reach convergence.

By expanding on these future directions and addressing the optimization of the flyby sequence, refining genetic algorithm parameters, incorporating additional constraints and objectives, leveraging computing advancements, and considering mission-specific factors, further advancements can be made in the field of interplanetary trajectory optimization.

BIBLIOGRAPHY

[1] H. Curtis. *Orbital Mechanics: For Engineering Students*. Aerospace Engineering. Elsevier Science, 2015. vii, 8, 9, 10, 11, 12, 22, 23, 24

[2] C. Kohlhase. *The Voyager Neptune travel guide.* 1989. vii, ix, 13, 14

[3] Iker Diaz Cilleruelo. Genetic algorithm for multi-gravity assist interplanetary trajectory optimisation. Master's thesis, Universitat Politècnica de Catalunya, February 2023. Accepted: 2023-02-22T12:13:51Z. vii, ix, 1, 3, 5, 8, 12, 15, 17, 19, 21, 24, 26, 29, 31, 35, 36, 37, 41, 44, 49, 50

[4] Jacob A. Englander, Bruce A. Conway, and Trevor Williams. Automated Mission Planning via Evolutionary Algorithms. *Journal of Guidance, Control, and Dynamics*, October 2012. ix, 1, 2

[5] Myles Standish and James Williams. *Orbital Ephemerides of the Sun, Moon, and Planets*. 01 2006. ix, 23

[6] Alberto Anselmi and George E. N. Scoon. BepiColombo, ESA's Mercury Cornerstone mission. , 49(14-15):1409–1420, December 2001. 1

[7] Yanping Guo and Robert W. Farquhar. New horizons mission design for the pluto-kuiper belt mission. 2002. 1

[8] Steve Matousek. The juno new frontiers mission. *Acta Astronautica*, 61:932–939, 11 2007. 1

[9] A.A. Siddiqi and United States. NASA History Program Office. *Beyond Earth: A Chronicle of Deep Space Exploration, 1958-2016*. NASA SP. National Aeronautics and Space Administration, Office of Communications, NASA History Division, 2018. 1

[10] Charles Kohlhase and Paul Anthony Penzo. Voyager mission description. *Space Science Reviews*, 21:77–101, 1977. 1

[11] Kaijian Zhu, Junfeng Li, and Hexi Baoyin. Multi-swingby optimization of mission to Saturn using global optimization algorithms. *Acta Mechanica Sinica*, 25(6):839–845, December 2009. 1

[12] Matteo Ceriotti. Global optimisation of multiple gravity assist trajectories. 2010. 1

[13] NASA's Europa Clipper. 3, 13

[14] D.R. Myatt S.J.Nasuto J.M.Bishop D. Izzo, V.M. Becerra. Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories. 4

[15] Edward Belbruno and John Carrico. Calculation of weak stability boundary ballistic lunar transfer trajectories. 08 2000. 9

[16] https://www.jpl.nasa.gov. Galileo - Jupiter Missions - NASA Jet Propulsion Laboratory. 13

[17] BepiColombo overview. 13

[18] Nasa's Voyager 2 probe 'leaves the Solar System'. *BBC News*, December 2018. 13

[19] Melanie Mitchell. *An Introduction to Genetic Algorithms*. The MIT Press, 03 1998. 15

[20] John H. Holland. Adaptation in natural and artificial systems. 1975. 15

[21] David E. Goldberg. Genetic algorithms in search optimization and machine learning. 1988. 15

[22] Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering System Safety*, 91(9):992–1007, 2006. Special Issue - Genetic Algorithms and Reliability. 15

[23] Matthew A Vavrina and Kathleen C Howell. MULTIOBJECTIVE OPTIMIZATION OF LOW-THRUST TRAJECTORIES USING A GENETIC ALGORITHM HYBRID. 15

[24] Elisa P. Dos Santos Amorim, Carolina R. Xavier, Ricardo Silva Campos, and Rodrigo W. Dos Santos. Comparison between Genetic Algorithms and Differential Evolution for Solving the History Matching Problem. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Beniamino Murgante, Osvaldo Gervasi, Sanjay Misra, Nadia Nedjah, Ana Maria A. C. Rocha, David Taniar, and Bernady O. Apduhan, editors, *Computational Science and Its Applications – ICCSA 2012*, volume 7333, pages 635–648. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Series Title: Lecture Notes in Computer Science. 17

[25] NASA JPL. Approximate positions of the planets. 24

[26] James F. Jordon. The application of lambert's theorem to the solution of interplanetary transfer problems. Technical report, NASA - Jet Propulsion Lab, 1964. 24

[27] Dario Izzo. Revisiting lambert's problem. *Celestial Mechanics and Dynamical Astronomy*, 03 2014. 24

[28] Victor R Bond. *Matched-conic solutions to round-trip interplanetary trajectory problems that insure state-vector continuity at all boundaries*, volume 4342. National Aeronautics and Space Administration, 1969. 25

[29] Linda Spilker. Cassini-huygens; exploration of the saturn system: 13 years of discovery. *Science*, 364(6445):1046–1051, 2019. 25

[30] Sam Wagner, Brian Kaplinger, and Bong Wie. Gpu accelerated genetic algorithm for multiple gravity-assist and impulsive v maneuvers. 08 2012. 25, 26, 35

[31] Jacob A. Englander, Bruce A. Conway, and Trevor Williams. Automated mission planning via evolutionary algorithms. *Journal of Guidance, Control, and Dynamics*, 35(6):1878–1887, 2012. 29