REINFORCEMENT LEARNING FRAMEWORK FOR SPACECRAFT LOW-THRUST ORBIT RAISING

A Thesis by

Lakshay Arora

Bachelor of Technology, Manipal Institute of Technology, India, 2017

Submitted to the Department of Aerospace Engineering
and the faculty of the Graduate School of
Wichita State University
in the partial fulfillment of
the requirements for the degree of
Master of Science

May 2020

REINFORCEMENT LEARNING FRAMEWORK FOR SPACECRAFT LOW-THRUST ORBIT RAISING

The following faculty members have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science, with a major in Aerospace Engineering.

Atri Dutta, Committee Chair

James E Steck, Committee Member

Zahra Nili Ahmadabadi, Committee Member

ACKNOWLEDGMENTS

I would first like to thank my thesis advisor Dr Atri Dutta, for his office door was always open whenever I ran into a trouble spot or had any question related about my research. He consistently allowed this thesis to be my own work, but steered me in the right the direction whenever he thought I needed it. I would also like to thank my committee members for their valuable comments on this research.

Finally, I would like to express my profound gratitude towards my parents, my sister Ankita and my dear friend, Sanober for providing me with unfailing support and continuous encouragement throughout my years of study and research.

ABSTRACT

The use of electric propulsion (EP) in satellites for transfer to geosynchronous equatorial orbit (GEO) is increasingly gaining importance among the space industry all around the world, and is proven a key for new space missions. In a conventional launch, the satellite is placed into a geostationary transfer orbit (GTO) by the launch vehicle and uses chemical propellants to reach GEO. This orbital transfer maneuver typically takes a few days. However, even though EP is far more efficient than the conventional chemical propulsion, its low thrust generation adds the complexity of longer transfer time from an equatorial orbit to GEO. This longer transit time leads to exposure of spacecraft to hazardous radiation of Van Allen belts. Therefore, there is a need to develop a method to determine the minimum transfer time trajectory for all-electric low thrust orbit raising problem.

This thesis proposes a new formulation that facilitates the application of reinforcement learning to the problem of orbit raising. This work is based on the approach that the electric orbit-raising problem is posed as a sequence of multiple trajectory optimization sub-problems. Each sub-problem aims to move the spacecraft closest to GEO by minimizing a convex combination of suitably selected objectives. A mathematical formulation for the orbit-raising problem is proposed in the framework of reinforcement learning to enable adaptive modification of the objective function weights during a transfer. Due to high dimensionality of the planning states of the orbit-raising problem, artificial neural networks are then constructed and trained on orbit-raising scenarios in order to compute the reward functions associated with reinforcement learning. The reward function for a planning state is defined as the time required to reach GEO from that planning state. With the help of numerical simulations for planar and non-planar transfer scenarios, it is demonstrated that there is a reduction in transfer time for low-thrust orbit raising problem with the proposed methodology.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (continued)

NOMENCLATURE

SYMBOLS

$\hat{Q}$	predicted value of transfer time required to reach GEO from a planning state, sidereal days
$\boldsymbol{\theta}$	trainable parameter of the neural network
$\boldsymbol{A}$, $\boldsymbol{B}$	functions of state variables in low-level optimization problem
$\boldsymbol{B_1}, \boldsymbol{B_2}$	biases for the neural network
$\boldsymbol{s}_k$	planning state at k-th revolution
$\boldsymbol{u}$	control thrust vector, N
Δw_h	change in the relative weight of the magnitude of the angular momentum
$\mathbb{G}$	rate of change of state variables due to thrust acceleration
μ	gravitational parameter, km^3/s^2
$\mathbf{F}$	Keplerian rate of change of state variables
$\mathbf{x}$	state vector of the spacecraft
h	magnitude of specific angular momentum, km^2/s
i	inclination angle, deg
k	index of an sub-problem optimization defined over a revolution of the spacecraft
m	mass of the spacecraft, kg
N_r	total number of revolutions/planning steps
Q	measure of the expected reward, sidereal days
R_k	reward/transfer time corresponding to k-th revolution, sidereal days
W_1, W_2	weights of inputs for the neural network
w_e	relative weight for eccentricity vector of the osculating orbit
w_{hxy}	relative weight for the projection of the angular momentum vector to the inertial X-Y plane
w_h	relative weight for angular momentum component for the osculating orbit and the GEO

ACRONYMS

GEO	Geosynchronous Equatorial Orbit
GTO	Geosynchronous Transfer Orbits
MDP	Markov Decision Process
ML	Machine Learning
MSE	Mean Squared Error
NN	Neural Network
RL	Reinforcement Learning
Sub-GTO	Sub Geosynchronous Transfer Orbits
Super-GTO	Super Geosynchronous Transfer Orbits
TT_{adapt}	Transfer time with adaptive weights
TT_{const}	Transfer time with constant weights
$TT_{general}$	Transfer time using generalized NN

CHAPTER 1

INTRODUCTION

1.1 Motivation

For many years now, onboard low-thrust electric propulsion has been proving more efficient than chemical propulsion, owing to the reduced fuel consumption[1]. Electric Propulsion (EP) is a class of space propulsion that has the largest market potential for commercial GEO telecommunication satellites. In 2017, the EUTELSAT 172B satellite, an "all-electric" built by Airbus DS, reached geostationary orbit by using the all-electric architecture [2]. In the last two decades, commercial GEO telecommunication satellites have become more competitive among satellite operators with the adoption of EP for North-South Station Keeping (NSSK) and Electric Orbit Raising (EOR) [3]. These telecommunication satellites are launched into initial orbits like Low Earth Orbit (LEO), Geostationary Transfer Orbits (GTO) and then with the help of onboard propulsion, orbit-raising maneuvers are performed to reach GEO. Table 1.1 provides the data for the initial orbit specifications taken from Falcon 9 user manual[4].

Table 1.1: Specifications of initial orbits

Initial Orbit	Orbit Altitude (km)	Inclination angle (deg)
LEO	200-2000	28.5, 38, 51.6
GTO	10000-100000	15, 17, 19, 21, 23, 25, 28.5

Telecommunication satellites using EP for orbit-raising have greater appeal since the propellant mass saved can be used to accommodate larger and more complex payloads. Besides, in the last decade, the trend in GEO telecommunication satellites has consolidated into a considerable increase in electrical power to satisfy the payload needs [5]. Specific

mission requirements, in terms of power availability, satellite mass, and mission profile, dictate the choice of the particular EP technology which is used for a mission.

Although limited to being used for station-keeping for a number of years, recent advancements in all-electric propulsion architecture have seen satellites perform the orbit-raising maneuver using EP. The major advantage of all-electric architecture over traditional chemical (and even hybrid) systems is that it enables the design of smaller and lighter satellites that can be potentially launched at the same time thereby reducing launch costs[6].

However, the biggest trade-off is the transfer time to GEO; while chemical propulsion gets the job done in days, EP takes months because of the low-thrust generated by the electric thrusters [7]. The longer transfer time leads to degradation of solar arrays of the satellite due to continuous exposure to Van Allen radiation[8] which are the doughnut-shaped zones of highly energetic charged particles trapped within the zones in the magnetic field of Earth. In this thesis, the problem of electric orbit-raising has been revisited as detailed in a recent work that computes low-thrust orbit-raising trajectories in a fast and robust manner [9]. The considered low-thrust orbit-raising problem is posed as a sequence of multi-revolution optimal control subproblems. This framework of multiple subproblems aims to minimize the distance of the spacecraft from GEO for every subproblem without the need of initial guess, thereby generating electric orbit-raising trajectories in a fast automated way. However, these solutions obtained comes at the cost of suboptimality of the generated solutions.

Motivated by the aim of reducing this optimality gap of the computed solutions, this thesis introduces an approach that builds upon the branch of machine learning (ML). It is believed that recent advancements in the field of machine learning may hold the answer to our problem. By definition, ML allows systems to improve their behaviors as they gain experience. To this end, the application of machine learning techniques to spacecraft orbit raising problems are considered to improve previously computed solutions by this method.

1.2 Literature Review

1.2.1 Low-thrust Orbit Raising Optimization

Over the last few decades, researchers have been focusing on how to solve optimal control problem associated with electric low-thrust maneuvers concerning different mission scenarios [10, 11, 12]. For low thrust mission scenarios, as mentioned earlier, the primary concern of using all-electric spacecraft is the long transfer time taken by the spacecraft to reach GEO. Moreover, spacecraft is encountered by multiple eclipses during its route which further prolongs the transfer time unless stored energy is provided to electric thrusters to operate during eclipses[13]. That is why electric orbit-raising problem mission scenarios should be investigated closely with a variety of required crucial factors by the mission designers. These include the initial orbit of the spacecraft, propellant mass, delivered dry mass, the capacity of the solar array panels to generate thrust during transfer, collision avoidance with external objects, the type of thrusters to be used including but not limited to Hall, Arcjet, Magneto Plasma Dynamic (MPD) or Ion thrusters and the type of power source during eclipses. Therefore, a mission designer has to investigate variety of scenarios in order to obtain trajectory with least time.

The problem of minimum-time trajectory determination is an optimal control problem which has been studied by numerous researchers within the field of low thrust orbit raising[14, 15]. In general, an optimal control problem is solved by applying numerical methods from the calculus of variations. To solve low-thrust trajectory optimization. traditional approaches uses this calculus of variation technique [16]. Indirect optimization methods transform a problem into a nonlinear two-point boundary value problem (TPBVP) with necessary conditions of optimality. These necessary conditions defined by a set of ordinary differential equations which are solved along the equations of motion given constraints on initial or final states. Solutions to TPBVP are then obtained by various methods, including a well-known shooting method [17]. This solution is at least locally optimal since the first-order necessary conditions for optimality are satisfied. Thorne and Hall [18] found the method for

minimum transfer time for circular to circular orbit-raising by indirect optimization method in 1996. However, TPBVP is sensitive to the initial guess with a small convergence domain that implies a small change in the values can lead to divergence. This leads to difficulty in converging to an optimal solution.

In order to overcome this issue, another approach called direct optimization methods are used. One can discretize the state and control variables with respect to time and apply quadrature rules across the discretized segments in order to set up a parameter optimization problem which is a nonlinear programming problem[19, 20], solved by software such as Sparse Nonlinear OPTimizer (SNOPT), Interior Point OPTimizer (IPOPT) [21, 22, 23]. Even though convergence rate of direct methods for low-thrust optimization is better than indirect methods, direct methods still need some kind of user-input initial guesses. Due to the lack of exact knowledge about these initial guesses, a number of studies such as shape-based methods [24, 25, 26, 27, 28, 29] and guidance-based schemes [30, 31, 32], have been investigated in order to generate approximate solutions to low-thrust optimization problem. These solutions can be used as a good initial guesses for direct or indirect optimization methods.

The problem of electric orbit-raising has been studied using a variety of different dynamic models, and Ref.[33] provides a comparison of a variety of dynamic models, specifically for their influence on the convergence of numerical optimization schemes for computing the low-thrust trajectory. By providing comparisons with various state variable models, a set of newly introduced regularized elements (Ref.[9]) was proved to be efficient for the low-thrust orbit-raising problem. The key advantages of using this dynamic model are:

- The singularity vanishes for both equatorial or circular orbits.
- Since among the regularized elements, five are constant for Keplerian motion that change slowly under the action of perturbations. This makes them suitable for use in trajectory optimization schemes that compute long-time-scale transfers.

1.2.2 Machine Learning

Artificial Intelligence (AI) is an interdisciplinary science with multiple approaches that includes the systems that think like humans, systems that act rationally, systems that act like humans and systems that act rationally[34]. However, advancements in machine learning are creating a paradigm shift in virtually every sector of the tech industry. Machine learning (ML) is the method in the field of artificial intelligence which involves different learning techniques for systems to automatically learn and improve from experience without being explicitly programmed. In ML literature, there are three types of learning techniques commonly used based on the type of problem in consideration. These types are:

i) Supervised learning

ii) Unsupervised learning

iii) Reinforcement learning

Supervised learning

Supervised learning is a type of machine learning in which learning is done offline from a labeled training dataset provided by a knowledgeable *teacher*. The goal is to approximate the function from the labelled training dataset. This function is then used to predict the output variables for the new untrained input data. A detailed comparison of supervised learning algorithms is given by Ref.[35] This is an important kind of learning, but alone it is not adequate for learning from interaction with the environment.

Unsupervised learning

Unsupervised learning is another type of machine learning in which learning is typically about finding structure hidden in collections of unlabeled data. Unsupervised learning tries to make sense of the structure in the given data and responds accordingly. Unsupervised learning has many applications in the real world. Clustering is an important type of learning when it comes to unsupervised learning. In the medical field, it has been proven to

be a powerful tool [36]. Other typical applications of unsupervised learning are included in the fields of genome informatics[37] and telecommunications fraud detection[38].

Reinforcement learning

Reinforcement learning (RL) is one of the learning techniques used in machine learning for solving sequential decision problems [39]. It was originally developed as a way of describing observations in animal behavior and has been successfully used in applications in various domains such as medical imaging [40], finance and management [41], cryptocurrency [42], robotics [43], cell-phone network routing [44] and control theory [45, 46, 47, 48]. These RL problems can be formalized by Markov decision process (MDP). By describing a problem in terms of a MDP, dynamic programming(DP) techniques may be applied to find an optimal solution in which a complex problem is broken into simple similar sub-problems, so that their results can be re-used. Dynamic programming however requires that a complete and accurate model of the environment is available, and this may not be the case. RL allows an agent to learn in an uncertain environment by building up an internal model of its environment through sample interactions. As its understanding of the environment increases with experience the learning agent may predict with more confidence which actions will tend to lead to preferred results. The working of RL is depicted in Figure 1.1.

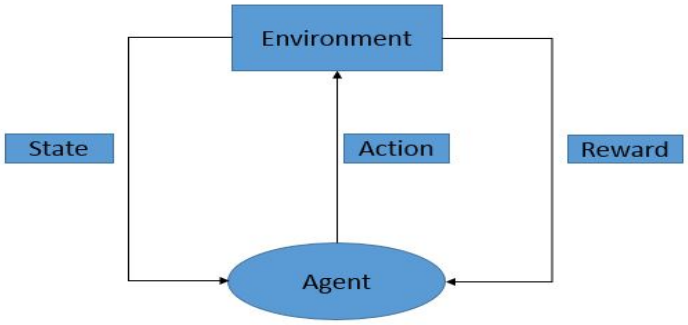

Figure 1.1: Representation of Reinforcement Learning

In recent years, machine learning has been applied to several aerospace problems, a survey of which can be found in Ref. [49]. Geometric Reinforcement Learning (GRL) where

reward matrix is adaptively updated based on the geometric distance and risk information shared by other UAVs is proposed for Unmanned Aerial Vehicles (UAVs) path planning[50], Proximal Policy Optimization (PPO) method which is a complex policy that is developed by training a neural network controller to solve a low-trust optimal control problem[51] and Q-Learning is used for path planning for a spacecraft to maneuver through a satellite cluster[52]. The Q-Learning algorithm allows the satellite controller to make sequential decisions in order to safely navigate through the environment. This algorithm is later augmented by Deep Neural Network (DNN) which results in Deep Q-Network (DQN).

1.2.3 Neural Networks

Artificial Neural Network (ANN) is a collection of simple high-speed computational devices that are connected and can learn to do tasks aided by parallel computing. ANNs are inspired by biological neural networks in the human brain. Learning of a neural network (NN) is necessary when the information about inputs/outputs is unknown or incomplete a priori, so that no design of a network can be performed in advance. Almost all the neural networks require learning in a supervised mode, unsupervised mode or reinforcement mode. A neural network is an ANN with artificial neurons that mimic the operations of a human brain to recognize relationships between vast amounts of data. While neural networks can learn to solve enormously diverse tasks, the inner workings of these models are often difficult to comprehend.

Moreover, neural networks have got the attention of various researchers in the field of spaceflight mechanics, especially as an application to spacecraft trajectory optimization. For instance, in Ref. [53] trained NNs are used for mass optimal control for Earth-Mars transfers, Ref. [54] depicts the performance of the trained NNs for time-optimal low-thrust orbit raising problems.

1.3 Thesis Contributions

The contribution of this thesis are as follows:

1. This thesis presents a parametric study in order to investigate the impact of the relative weights of the objective function on the orbit-raising transfer time. This is done to reduce the optimality gap of the computed solutions generated by the considered orbit-raising trajectory optimization method. The parametric study provides insight into selection of objective function weights which remain constant during the orbit-raising maneuver. The work has been published in the proceeding of $29th$ AAS/AIAA Space Flight Mechanics Meeting [21].

2. This thesis develops a new formulation that allows the application of reinforcement learning to the orbit-raising problem. The proposed reinforcement learning framework, aided by neural networks, provide a mechanism to modify the weights during the maneuver. This work has been recently published in AIAA Scitech 2020 Forum [55].

1.3.1 Organization of Thesis

The thesis is organized as follows. Chapter 2 describes the spacecraft translational dynamics and dynamic model used for the low-level optimization for orbit-raising problem. It also briefly describes the optimization scheme used for our low-level optimization problem for orbit-raising. Chapter 3 introduces the formulation of high-level planning problem for optimizing the relative weights and how it is integrated with the low-level optimization scheme. Furthermore, it describes the machine learning technique in consideration, that is, reinforcement learning which is used to optimize the relative weights of the objective function for low-level optimization problem for orbit-raising. In Chapter 4, the implementation of the proposed reinforcement learning algorithm to the given planning problem is demonstrated and discussed with the help of various numerical examples. Chapter 5 summarizes the work and recommends future directions.

CHAPTER 2

MATHEMATICAL FORMULATION OF ORBIT-RAISING PROBLEM

In this chapter, the mathematical formulation of the low-thrust orbit-raising problem is developed allowing the application of a reinforcement learning technique. Specifically, this is formulated as a two-step decision-making process: a low-level trajectory optimization problem and a high-level planning problem. The low-level optimization problem determines the spacecraft trajectory over a planning horizon by optimizing a convex combination of objective functions, while the high-level planning problem decides on the objective function weights. To facilitate this formulation, a set of planning states is introduced apart from the spacecraft state variables, and also outline the dynamics associated with the planning states apart from the spacecraft dynamic equations. This chapter provides the spacecraft dynamics and low level optimization scheme associated with it.

2.1 Minimum-time Trajectory Optimization for Orbit-raising

Numerous researchers have looked at the problem of optimal low-thrust trajectories in general and various studies have investigated the orbit-raising problem in particular over the last few decades. Popular research topic includes use of optimal-control trajectories to minimize time or fuel usage of constant-thrust transfers. The problem of minimum-time control is solved for constant-thrust coplanar transfers to generate simple graphical and analytical tools to relate vehicle design parameters to orbit design parameters for orbit raising[56]. In Ref.[12], a direct optimization framework for near-optimal minimum-time low Earth orbit (LEO) to geostationary orbit (GEO) and geosynchronous transfer orbit (GTO) to GEO transfers is developed and examined showing robust results. Moreover, in Ref.[57] parameterized control law is employed together with orbital averaging in order to solve three common near-optimal minimum-time Earth-orbit transfers. In addition, a direct optimization method for the electric orbit-raising problem is employed using equinoctial

elements [22]. Later, minimum-time transfers using direct collocation for a range of initial thrust accelerations and constant specific impulse values are addressed by Ref.[14].

Within the literature of electric orbit-raising, a number of studies have been investigated which to compute initial guesses to generate the solution for low thrust trajectory. An intelligent guess is constructed for the multiple-phase optimal orbit-raising control problem with eclipsing constraint by considering a series of single-phase optimal control problems that are solved by the adaptive collocation method [23]. The receding horizon technique is used in Ref.[58] to construct an initial guess for low thrust orbit transfers. These studies require initial guess in order to generate the solution for low thrust trajectory. However, Ref.[9] explores an unconstrained optimization scheme that allows the computation of the electric orbit-raising trajectory in the order of tens of seconds without the need for any user-inputted initial guess. In addition, the formulation has the benefit to analyze numerous electric orbit-raising mission scenarios without the need for intervention by the user. The low-level optimization problem is solved by considering the same methodology in this work.

2.2 Translational Dynamics

Low-thrust trajectory optimization involves the translational dynamics of the spacecraft. This dynamics is based on Newtonian mechanics, in which Newton's law of gravitation states that an object attracts the other with a force equal to the product of their masses and inversely proportional to the square of the distance between them. A combination of this law with Newton's equations of motion forms the basis to study the orbital motion of the objects. For the electric orbit-raising mission, only two-body body problem is considered for low-level problem in this work. In the two-body problem, two assumptions are considered. First, only two bodies which are spherical and have uniform density are considered. Another assumption is that these bodies are point masses. However, these assumptions do not hold in reality. Presence of three bodies and perturbations somehow change the orbit significantly. Two-body problem is described as follows:

Consider vectors $\mathbf{r_1}$ and $\mathbf{r_2}$ which represent the position of the two bodies with masses m_1 (Earth) and m_2 (spacecraft) with respect to the origin of the frame $\mathcal{I}$ in Figure 2.1.

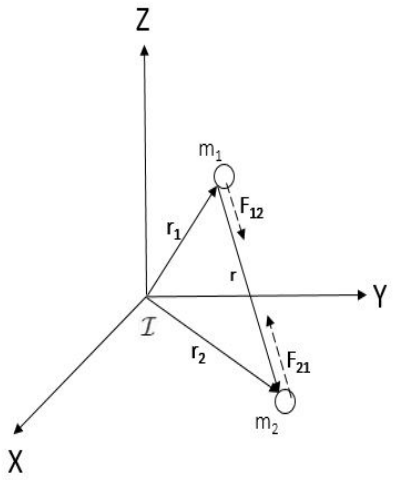

Figure 2.1: Schematic for Two-body problem

The spacecraft position vector with respect to the celestial body by $\mathbf{r}$ and the velocity vector by $\mathbf{v}$. Therefore, spacecraft translational motion can be described by the equations:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\frac{\mu}{r^3}\mathbf{r} + \mathbf{u} \end{aligned} \tag{2.1}$$

where r is the magnitude of vector $\mathbf{r}$, μ is Earth's gravitational parameter due to gravity, $\mathbf{u}$ includes the thrust and perturbations. In this work, the effect of perturbations is not considered, which therefore imply that only thrust forces are considered.

2.3 Optimization of Orbit-raising Subproblem

A multi-revolution large-scale optimal control problem is broken into a sequence of optimization sub-problems. To define the trajectory optimization sub-problem, the spacecraft states $\boldsymbol{x}$ are represented by a set of so-called *dynamical variables*. These parameters include the specific angular momentum h, components of the specific angular momentum vector (h_X and h_Y) along the X and Y axes of the Earth-centered inertial reference frame $\mathcal{I}$, the components of the eccentricity vector (e_x and e_y) along the x and y axes of the non-

inertial reference frame obtained after a 2-1 rotation sequence from the inertial frame, and a true-anomaly like angle ϕ identifying the location of the spacecraft. The state vector is therefore given by $\mathbf{x} = \begin{bmatrix} h, h_X, h_Y, e_x, e_y, \phi \end{bmatrix}^T$. The spacecraft's equation of motion is given as:

$$\dot{\mathbf{x}} = \mathbf{F}(x) + \mathbb{G}(x)\mathbf{u}, \tag{2.2}$$

where $\boldsymbol{u} \in \mathbb{R}^3$ represents the control thrust vector, $\mathbf{F} \in \mathbb{R}^6$ represents the Keplerian rate of change of state variables and $\mathbb{G} \in \mathbb{R}^{6\times 3}$ represents the rate of the change of the state variable provided by onboard thrusting. The state space description is provided below [9]:

$$\mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{\mu^2 B^2}{h^3} \end{bmatrix}, \quad \boldsymbol{A} = e_x \sin\phi - e_y \cos\phi, \quad \boldsymbol{B} = 1 + e_x \cos\phi + e_y \sin\phi \tag{2.3}$$

$$\mathbb{G} = \begin{bmatrix} 0 & \frac{h^2}{\mu m B} & 0 \\ 0 & \frac{h h_x}{\mu m B} & \frac{h^2\sqrt{h^2-h_x^2-h_y^2}}{\mu m B\sqrt{h^2-h_y^2}}\sin\phi + \frac{h h_x h_y}{\mu m B\sqrt{h^2-h_y^2}}\cos\phi \\ 0 & \frac{h h_y}{\mu m B} & -\frac{h\sqrt{h^2-h_y^2}}{\mu m B}\cos\phi \\ \frac{h\sin\phi}{\mu B} & \frac{2h\cos\phi}{\mu m} + \frac{hA\sin\phi}{\mu m B} & \frac{h e_y h_y \sin\phi}{\mu m B\sqrt{h^2-h_y^2}} \\ -\frac{h\cos\phi}{\mu B} & \frac{2h\sin\phi}{\mu m} - \frac{hA\cos\phi}{\mu m B} & -\frac{h e_x h_y \sin\phi}{\mu m B\sqrt{h^2-h_y^2}} \\ 0 & 0 & -\frac{h h_y \sin\phi}{\mu m B\sqrt{h^2-h_y^2}} \end{bmatrix}, \tag{2.4}$$

Trajectory over each segment is discretized into $p+1$ nodes and is denoted by ϕ_j the angles demarcating the discretized nodes, where $j = 0, ..., p$. For every revolution, the dynamical states h, h_X, h_Y, e_x and e_y are assumed to be approximately constant. Consequently, the trajectory of the spacecraft is represented in the shape of a conic section. The

discretization of the trajectory over one revolution is depicted by Figure 2.2. The details of this optimization algorithm can be found in Ref. [9].

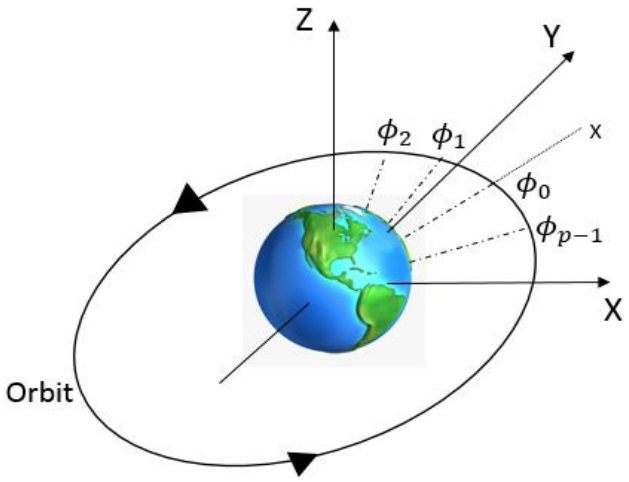

Figure 2.2: Subdivision of trajectory over one whole revolution

This optimization sub-problem methodology allows the computation of the electric orbit-raising trajectory without the need for any user-inputted initial guess in the matter of a few seconds.

GEO is described as a circular orbit which has altitude 35786 km with 0 degree inclination, as per Inter-Agency Space Debris Coordination (IADC)[59],. Its orbital period is one sidereal day which matches with Earth rotation period. The objective of each optimization sub-problem is to move the spacecraft as close to GEO as possible at the end of each revolution. The proximity to GEO is determined by the following quantities:

i) The difference of the magnitude of the specific angular momentum h for the osculating orbit and the GEO,

ii) The difference in eccentricity $e = \sqrt{e_x^2 + e_y^2}$ of the osculating orbit and that of GEO,

iii) The difference of the magnitude of the projection of the angular momentum vector to the inertial X-Y plane ($h_{xy} = \sqrt{h_X^2 + h_Y^2}$) .

The objective function to be minimized is written in the form of a convex combination of three factors mentioned above:

$$J = \min[w_h(h_{GEO} - h_{end})^2 + w_e(e_{x,end}^2 + e_{y,end}^2) + w_{hxy}(h_{x,end}^2 + h_{y,end}^2)], \tag{2.5}$$

where the subscript *end* indicates the dynamical variables at the end of a revolution.

In addition, w_h, w_e and w_{hxy} represent the relative weights for the three individual components of the objective function as described above. Note that, the sum of the weights equals 1, that is

$$w_h + w_e + w_{hxy} = 1 \tag{2.6}$$

For the special case of planar transfer in which the orientation of the orbit is not considered. Therefore, in planar transfer $h_X^2 + h_Y^2 = 0$ by definition, hence, there is no need to consider the third weight factor. In such a scenario, the only state variables that are relevant for the transfer are h, e_x and e_y and the objective function gets reduced to:

$$J_p = \min\left[w_h(h_{GEO} - h_{end})^2 + w_e(e_{x,end}^2 + e_{y,end}^2)\right]. \tag{2.7}$$

The overall optimization algorithm is depicted in the form of a flowchart in Figure 2.3.

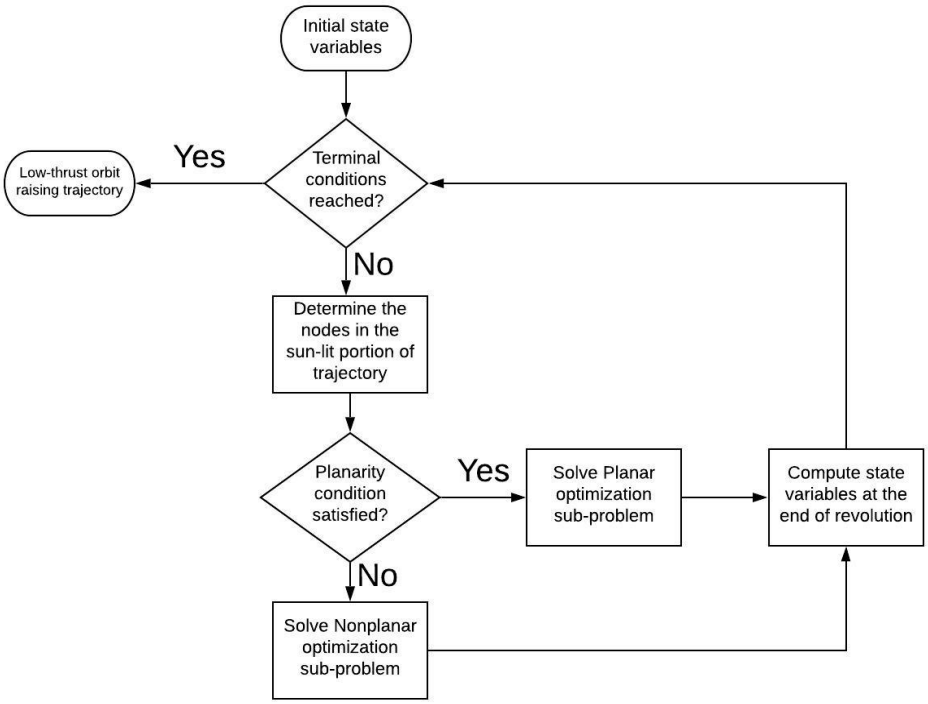

Figure 2.3: Low-level optimization algorithm flowchart

2.3.1 Terminal conditions

These stopping conditions determine that if the spacecraft has reached its final destination, that is, GEO. The targetted GEO parameters are the magnitude of the specific angular momentum, eccentricity and the magnitude of the projection of the angular momentum on the inertial X-Z plane. By the definition of GEO, tolerances on the eccentricity, semi-major axis, and inclination angle are set accordingly. These criteria is given by:

$$0 \leq \sqrt{e_{x,end}^2 + e_{y,end}^2} \leq e_{tol}, \tag{2.8}$$

$$a_{GEO} - a_{tol} \leq \frac{h_{end}^2}{\mu(1 - e_{x,end}^2 - e_{y,end}^2)} \leq a_{GEO} + a_{tol} \tag{2.9}$$

$$0 \leq \sqrt{\frac{h_{X,end}^2 + h_{Y,end}^2}{h}} \leq \sin i_{tol} \tag{2.10}$$

where e_{tol}, a_{tol} and i_{tol} are the tolerances defined for eccentricity, semi-major axis and inclination angle of the GEO, respectively.

2.4 Parametric Study

The solutions obtained by the considered low-level optimization problem are suboptimal, and it is demonstrated that the optimality gap is larger for starting elliptic orbits such as GTO, sub-GTO, and super-GTO, compared to starting circular orbits in low-Earth orbits (LEO)[9]. With the goal of reducing the optimality gap, low-thrust sequential optimization sub-problems are considered, whose objective function has three components (two in the case of planar transfers) given by Eq.(3.4) and Eq.(3.6) combined using a convex combination of weights. Considering the local objective function for an optimization sub-problem, a parametric study is conducted in order to investigate the effect of these relative on the orbit-raising transfer time using the above-mentioned dynamic model. A number of different orbit-raising scenarios are used to illustrate this impact with numerical examples[21].

2.4.1 Planar Transfers

For planar transfer maneuvers, the solution of 100 orbit-raising problems for each initial orbits by varying w_h linearly between 0.1 and 0.99. This is because the selection of w_h completely describes the objective function given by Eq.(2.7) with $w_{hxy} = 0$ and $w_e = 1 - w_h$.

Example 1: GTO to GEO Planar Transfer

In this transfer type, GTO is taken as the initial orbit of the spacecraft with a perigee altitude of 250 km. Naturally, the apogee altitude is 35786 km. Moreover, the initial mass of the spacecraft is taken as 5000 kg. For this scenario, a number of orbit-raising transfer problems are generated for linearly space choices of $w_{h,k}$ between 0.1 and 0.99. Figure 2.4 shows the variation of the transfer time with the considered values of w_h. Among all transfers considered, the trajectory with $w_h = 0.8282$ was found to yield the minimum transfer time of 107.94 sidereal days, a final mass of 4413.38 kg and a total of 168 revolutions. Compared to the scenario when weights are equal (that is, $w_h = w_e = 0.50$), the transfer time is 124.96 sidereal days, final mass is 4328.72 kg, and the transfer is completed over 228 revolutions.

By significantly increasing the relative weighting for the angular momentum difference (high w_h), there is a substantial reduction in the transfer time of 13.65%.

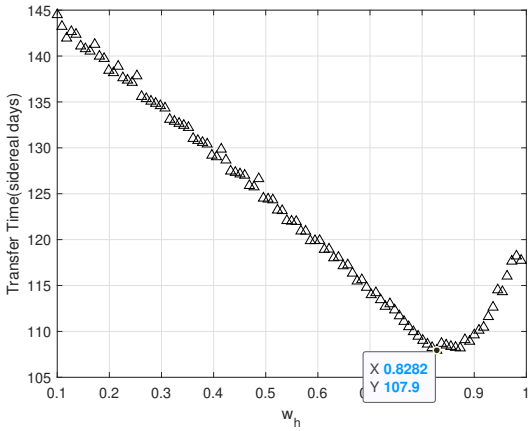

Figure 2.4: Variation of w_h for GTO to GEO planar Transfer

Example 2: Sub-GTO to GEO Planar Transfer

In this transfer type, the initial orbit of the spacecraft is considered to be Sub-GTO. The perigee altitude and apogee altitude for this particular orbit are taken as 250 km and 30000 km, respectively. Moreover, the spacecraft with initial mass of 5000 kg is taken for this trasnfer. For this scenario as well, multiple orbit-raising problems are solved for linearly-spaced vector w_h varying between 0.1 and 0.99. Figure 2.5 depicts the variation of the corresponding transfer time with respect to chosen values of w_h for these transfers. Of those computed trajectories, the one with $w_h = 0.8731$ yields the minimum transfer time of 113.1 sidereal days, a final mass of 4387.24 kg after completing a total of 192 revolutions. On comparing with the least transfer time result, an equal weighting of the individual objectives ($w_h = w_e = 0.50$) result in transfer time of 133.46 sidereal days, a final mass of 4288.98 kg, and a total of 282 revolutions. By a heavier weighting of w_h, there is a considerable reduction in transfer time by 15.25%.

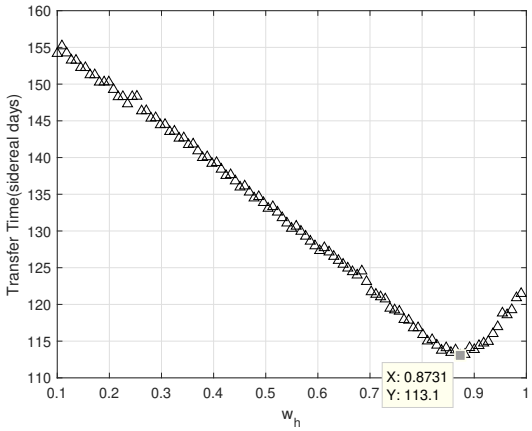

Figure 2.5: Variation of w_h for Sub-GTO to GEO planar Transfer

Example 3: Super-GTO to GEO Planar Transfer

In this transfer type, the initial orbit of the spacecraft is considered to be Super-GTO. The perigee altitude and apogee altitude for this particular orbit are taken as 250 km and 45000 km, respectively. Initial value of the mass of the spacecraft is taken to be 5000 kg, same as in the previous transfers. For this scenario, a number of orbit-raising transfer trajectories are computed for linearly spaced vector w_h varying between 0.1 and 0.99. Figure 2.6 depicts the variation of the corresponding transfer time with respect to chosen values of w_h for these transfers. Out of these computed low-thrust orbit-raising trajectories, the one corresponding to $w_h = 0.8012$ gives the least transfer time of 103.34 sidereal days, a final mass of 4434.96 kg and a total of 134 revolutions. Comparing to the solution for equal weighting of the two components of the objective function ($w_h = w_e = 0.50$) with the transfer time of 113.07 sidereal days, a final mass of 4386.57 kg and a total of 168 revolutions), it is found that there is a reduction in transfer time for Super-GTO scenario by 8.6% by following a non-equal weighting of the objective function components.

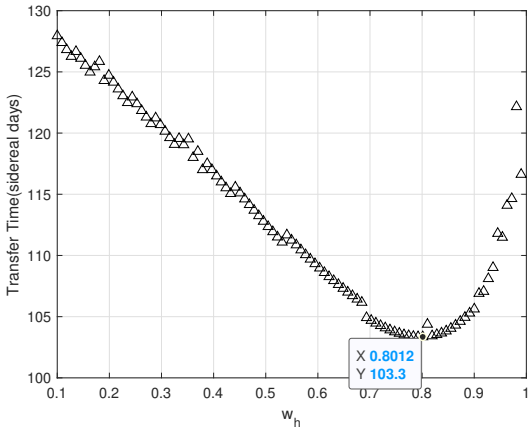

Figure 2.6: Variation of w_h for Super-GTO to GEO planar Transfer

2.4.2 Nonplanar Transfers

For nonplanar orbit-raising maneuvers, solution of $24^2 = 576$ orbit-raising problems for each initial orbits by varying w_h and w_{hxy} linearly between 0.1 and 0.99. Choosing at least two relative weights out of three are mandatory which describes the objective function given by Eq.(2.5). Note that selection of w_h and w_{hxy} automatically determines the third relative weight $w_e = 1 - w_h - w_{hxy}$.

Example 4: GTO to GEO Nonplanar Transfer

In this case, the spacecraft initially to be in a perigee altitude of 250 km and an apogee altitude of 35786 km with an inclination of 28.5 deg is considered. The initial mass of the spacecraft is considered as 5000 kg, same as in the planar transfers. For the considered nonplanar transfer scenario, 576 orbit-raising instances are solved, considering linearly spaced values of w_h and w_{hxy} varying between 0.1 and 0.99. Figure 2.7 illustrates the variation of the corresponding transfer time with respect to chosen values of w_h and w_{hxy} for these transfers. Out of those trajectories, the one with $w_h = 0.4400$ and $w_{hxy} = 0.5257$ gives the least transfer time of 155.91 sidereal days, a final mass of 4134.28 kg and a total of 225 revolutions. Compared to the case of equal weighting ($w_h = 0.3333, w_{hxy} = 0.3333$) that yields a transfer time of 190.46 sidereal days, a final mass of 3933 kg and a total of 335

revolutions, the unequal weighting of the objective function components yields a 18.14 % reduction in transfer time.

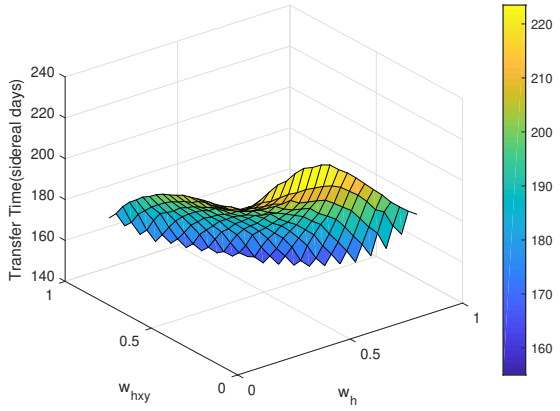

Figure 2.7: Variation of w_h for GTO to GEO nonplanar Transfer

Example 5: Sub-GTO to GEO Nonplanar Transfer

In this case, the spacecraft initially to be in a perigee altitude of 250 km and an apogee altitude of 30000 km with an inclination of 28.5 deg is considered. Similar to the planar transfers, initial mass of the spacecraft to be 5000 kg is taken and 576 orbit-raising problems are solved based on equally spaced values of w_h and w_{hxy} varying between 0.1 and 0.99. Figure 2.8 depicts the variation of the corresponding transfer time with respect to chosen values of w_h and w_{hxy} for these transfers. Out of those trajectories, trajectory with $w_h = 0.4400$ and $w_{hxy} = 0.5250$ gives the least transfer time of 161.60 sidereal days, a final mass of 4105.74 kg and a total of 265 revolutions. When compared to the solution yielded by an equal weighting of the components of the objective function ($w_h = 0.3333, w_{hxy} = 0.3333$, transfer time of 197.24 sidereal days, final mass of 3897.84 kg and total of 394 revolutions), it is found that unequal weighting scheme can lead to a reduction in transfer time by 18.07%.

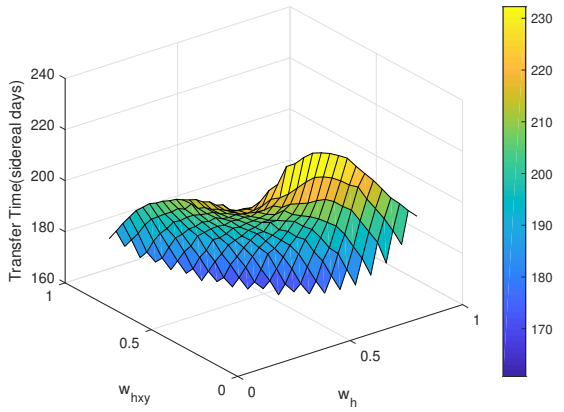

Figure 2.8: Variation of w_h for Sub-GTO to GEO nonplanar Transfer

Example 6: Super-GTO to GEO Nonplanar Transfer

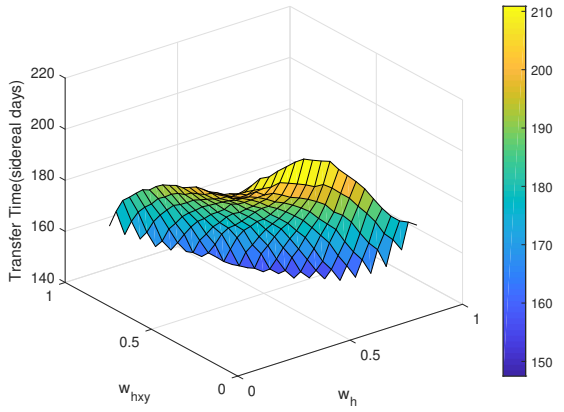

Figure 2.9: Variation of w_h for Super-GTO to GEO nonplanar Transfer

In this case,the spacecraft initially to be in a perigee altitude of 250 km and an apogee altitude of 45000 km with an inclination of 28.5 deg, with the initial mass of the spacecraft to be 5000 kg is considered. A number of orbit-raising problems are solved generated for linearly spaced values of w_h and w_{hxy} varying between 0.1 and 0.99, and Figure 2.9 depicts the variation of the corresponding transfer time with respect to chosen values of w_h and w_{hxy} for these transfers. Out of those trajectories, trajectory with $w_h = 0.40$ and $w_{hxy} = 0.55$ gives the minimum transfer time of 150.09 sidereal days, a final mass of 4166 kg and a total of 193 revolutions as compared to $w_h = 0.3333$ and $w_{hxy} = 0.3333$ with the transfer time of

181.86 sidereal days, a final mass of 3978.87 kg and a total of 269 revolutions. By adjusting w_h and w_{hxy}, there is a reduction in transfer time for by 17.47%.

For each orbital transfer type, the computational time is observed in the order of hours to generate minimum transfer time trajectory. Table 2.1 summarizes the numerical results for all mission scenarios for planar and nonplanar low-thrust maneuvers.

Table 2.1: Summary of Parametric study results

Initial Orbit	Least Transfer Time (sidereal days)	w_h	w_e	w_{hxy}	Final Mass (kg)
Planar Transfers					
GTO (250 km x 35786 km altitude)	107.94	0.8282	0.1718	-	4413.38
Sub-GTO (250 km x 30000 km)	113.1	0.8731	0.1269	-	4387.24
Super-GTO (250 km x 45000 km)	103.34	0.8012	0.1988	-	4434.96
Nonplanar Transfers (i = 28.5 deg)					
GTO (250 km x 35786 km)	155.91	0.4400	0.0343	0.5257	4134.28
Sub-GTO (250 km x 30000 km)	161.60	0.4400	0.035	0.5250	3897.84
Super-GTO (250 km x 45000 km)	150.09	0.40	0.05	0.55	4166

CHAPTER 3

PROPOSED METHODOLOGY

In this chapter, the optimization scheme described above is formulated as the low-level trajectory optimization for our orbit-raising problem and a methodology for incorporating a machine learning algorithm is provided to facilitate the high-level decision-making of adaptive choosing of objective function weights. To facilitate this formulation, a set of planning states is introduced apart from the spacecraft state variables, and also outlines the dynamics associated with the planning states apart from the spacecraft dynamic equations as described in the previous chapter.

3.1 High-Level Planning Problem

To this end, an integer $k \in \mathbb{I}+$ is introduced that represents the index of the revolution that the spacecraft is undergoing at a given time. Note here that although the number of revolutions is finite for a given orbit-raising problem, the total number of revolutions is not known a priori and is obtained as a part of the solution.

At the high-level, the decision-making involves the selection of the weights $w_{h,k}$, $w_{e,k}$ and $w_{hxy,k}$ associated with the objective function. This decision-making problem is considered to be a discrete optimization process, where selections are made at each planning horizon. Each selection is referred to as an action that determines the objective function for the low-level trajectory optimization problem. To this end, a set of planning states is defined and then an action is defined that transfers the planning problem from one state to another. The set $\boldsymbol{s}_k$ planning states is defined by the spacecraft orbit (defined by h_k, e_k and inclination i_k) at the beginning of the k-th revolution, along with the weights $w_{h,k}$, $w_{e,k}$ and $w_{hxy,k}$. Without loss of generality, we will assume that we will make modifications to $w_{h,k}$ and $w_{hxy,k}$. Therefore, for each planning step (k-th revolution), a set of feasible selections of weights is defined for:

$$\mathcal{W}_k = \{(w_{h,k}, w_{hxy,k}),\ (w_{h,k} + \Delta w_h, w_{hxy,k}),\ (w_{h,k} - \Delta w_h, w_{hxy,k}),\\ (w_{h,k}, w_{hxy,k} + \Delta w_{hxy}),\ (w_{h,k}, w_{hxy,k} - \Delta w_{hxy})\}, \tag{3.1}$$

where Δw_h and Δw_{hxy} are user-defined changes in weights that can be considered in each planning period.

Note here, for the special case of a planar transfer, the planning problem simplifies to the consideration of a smaller number of variables. Specifically, the variables i_k and $w_{hxy,k}$ are zero. Furthermore, the set of feasible selections is reduced to the following:

$$\mathcal{W}_{p,k} = \{w_{h,k},\ w_{h,k} + \Delta w_h,\ w_{h,k} - \Delta w_h\}. \tag{3.2}$$

Now, action a_k is defined as a mapping from the current weights $w_{h,k}$ and $w_{hxy,k}$ to the new weights $w_{h,k+1}$ and $w_{hxy,k+1}$ to be used for low-level optimization problem. For planar and nonplanar transfers, a_k is described as

$$a_k : \begin{cases} (w_{h,k}, w_{hxy,k}) \mapsto (w_{h,k+1}, w_{hxy,k+1}), & \text{Nonplanar transfer} \\ w_{h,k} \mapsto w_{h,k+1}, & \text{Planar transfer} \end{cases} \tag{3.3}$$

To complete the description of the decision-making problem associated with the high-level planning of the orbit-raising trajectory, each action is considered to have a corresponding cost (in the case of minimization problem) or reward (in the case of a maximization problem). In the context of the current work, the reward function is considered to be the amount of transfer time associated with the orbit-raising maneuver obtained due to the change of the objective function weights. This reward function at the beginning of the k-th planning step is denoted by $Q(\boldsymbol{s}_k, a_k)$. The objective of the high-level planning problem is to choose the action that leads to the largest reward, or equivalently the smallest transfer time for the orbit-raising maneuver. This strategy of determining the next planning state of the spacecraft ($\boldsymbol{s_{k+1}}$) by employing the best/optimal action based on the known current planning state ($\boldsymbol{s_k}$) is known

as the optimal policy that is denoted by $\pi^*(\boldsymbol{s_k}, \boldsymbol{a_k})$. The optimal action for a particular planning state is defined as an action which gives the minimum transfer to reach the target orbit GEO.

3.2 Low-level Trajectory Optimization

The objective function for low-level optimization problem is same as Eq.(2.5) which is to be minimized during the k^{th} revolution is written as:

$$J_k = \min[w_{h,k}(h_{GEO} - h_{k+1})^2 + w_{e,k}(e_{x,k+1}^2 + e_{y,k+1}^2) + w_{hxy,k}(h_{x,k+1}^2 + h_{y,k+1}^2)], \tag{3.4}$$

where the subscripts k and $k+1$ indicate the variables at the beginning and end of k-th revolution respectively. Note that the end of the k-th revolution marks the beginning of the $(k+1)$-th revolution. In addition, $w_{h,k}$, $w_{e,k}$ and $w_{hxy,k}$ represent the relative weights for the three individual components of the objective function as described above. Note that, the sum of the weights equals 1, that is,

$$w_{h,k} + w_{e,k} + w_{hxy,k} = 1. \tag{3.5}$$

Similarly the objective function for planar case in Eq.(2.7) is written as:

$$J_{pk} = \min\left[w_{h,k}(h_{GEO} - h_{k+1})^2 + w_{e,k}(e_{x,k+1}^2 + e_{y,k+1}^2)\right]. \tag{3.6}$$

Furthermore, low-level optimization is executed with the algorithm represented by Figure 2.3. The optimization methodology approximates the trajectory over a revolution by a conic section; hence, the transfer time over each planning horizon is given by:

$$R_k = \frac{h_{k+1}^3}{\mu^2}\sqrt{\frac{1}{1 - e_{x,k+1}^2 - e_{y,k+1}^2}}, \tag{3.7}$$

which will be treated as the reward corresponding to the k^{th} revolution, in the context of applying reinforcement learning to the orbit-raising problem. Note that the constraint Eq. 3.5 implies that only two of the three weights need to be selected.

The terminal conditions for the low-level optimization problem in the Eq.(2.8), Eq.(2.9) and Eq.(2.10) are then updated as

$$0 \leq \sqrt{e_{x,k+1}^2 + e_{y,k+1}^2} \leq e_{tol}, \tag{3.8}$$

$$a_{GEO} - a_{tol} \leq \frac{h_{k+1}^2}{\mu(1 - e_{x,k+1}^2 - e_{y,k+1}^2)} \leq a_{GEO} + a_{tol} \tag{3.9}$$

$$0 \leq \sqrt{\frac{h_{X,k+1}^2 + h_{Y,k+1}^2}{h}} \leq \sin i_{tol} \tag{3.10}$$

3.3 Q-Learning for Low-thrust Orbit Raising Problem

In Reinforcement learning (RL), the agent (spacecraft) first takes an action a_k from an initial state $\boldsymbol{s}_k$ to reach $\boldsymbol{s}_{k+1}$ and obtain reward R_k or the feedback to the agent's actions from the environment that eventually leads it to the terminal state. This one series of states, actions, rewards that ends at the terminal state is called an episode. As its experience with the environment increases, the learning agent may predict with more confidence which actions will tend to lead to preferred results. In the literature, the future rewards are discounted by a factor γ per time-step giving more preference to immediate rewards rather than long-term rewards. The sum of the discounted rewards from state $\boldsymbol{s}_k$ is defined as in Ref. [39].

$$G(\boldsymbol{s}_k) = R_{\mathrm{k}} + \gamma R_{\mathrm{k+1}} + ... = \sum_{k=1}^{N_r} \gamma^{k-1} R_k, \tag{3.11}$$

where γ is the discount factor considered to be 1 for our planning problem since the agent will evaluate each of its actions based on the sum total of all of its future rewards, R_k is the reward value for the k^{th} planning step given by Eq.(3.7) and N_r is the number of planning steps/revolutions determined by the low-level planning problem. In the Q-Learning

algorithm, $Q(\boldsymbol{s}_k, a_k)$ is the measure of the overall expected reward when the agent is at planning state s_k and performs a_k. This is used to approximate the Q-value based on a state. This Q-value is written as:

$$Q(\boldsymbol{s}_k, a_k) = E[G|\boldsymbol{s}_k, \boldsymbol{a}_k, \gamma = 1] = [-R(\boldsymbol{s}_k, \boldsymbol{a}_k) + \gamma \underset{\boldsymbol{a}_{k+1}}{max}(-Q(\boldsymbol{s}_{k+1}, \boldsymbol{a}_{k+1}))|\boldsymbol{s}_k, \boldsymbol{a}_k], \quad (3.12)$$

where $R(\boldsymbol{s}_k, \boldsymbol{a}_k)$ is the immediate reward when agent moves from state $\boldsymbol{s}_k$ to $\boldsymbol{s}_{k+1}$ and $Q(\boldsymbol{s}_k, \boldsymbol{a}_k)$ is the highest Q-value given state $\boldsymbol{s}_k$. In order to learn how to move and make decisions within the environment, the agent has to be trained. The agent explores every possible state-action pairs for all the states. A list of Q-values for all possible state-action pairs is constructed and these estimates are stored in a table which is known as the Q-table. These values keep on updating after every episode until the agent learns to act optimally in that environment, that is when the agent makes decisions such that it maximizes the expected future rewards (minimized transfer time). Trajectory of low-thrust orbit raising for the Q-learning process is illustrated by Figure 3.1.

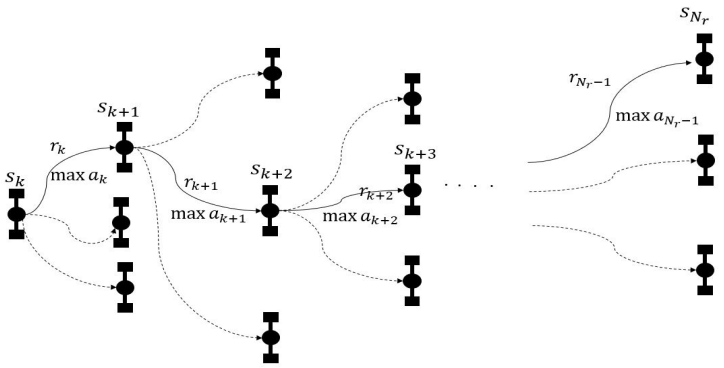

Figure 3.1: Graphic representation of Q-learning for orbit-raising trajectory

An important point to note here is the lack of knowledge of this reward function; hence, in the next section, a methodology is presented using neural network-based estimation of the reward function in order to facilitate the application of a reinforcement learning technique.

3.4 Deep Q-Learning

The environment for low-thrust planning problem is not defined explicitly in the reinforcement learning literature and it is impossible to access all the planning states and possible and create Q-table. It becomes challenging to get optimal state-action values for this type of complex problem with high dimensional states. To tackle this high dimensional problem, Q-table is replaced by a neural network which is known as a function approximator which is denoted by $\hat{Q}(\boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta})$. Here, $\boldsymbol{\theta}$ is defined as the network parameter which comprises of trainable weights and biases of the neural network, thus eliminating the need of learning rate. The performance of the neural network is calculated by the loss function given by

$$\mathrm{LF}\left(\boldsymbol{s}_k, \boldsymbol{a}_k\right) = \left[\hat{Q}\left(\boldsymbol{s}_k, \boldsymbol{a}_k; \boldsymbol{\theta}\right) - Q(\boldsymbol{s}_k, \boldsymbol{a}_k)\right]^2, \tag{3.13}$$

where $\hat{Q}\left(\boldsymbol{s}_k, \boldsymbol{a}_k; \boldsymbol{\theta}\right)$ is the predicted value of transfer time required to reach GEO from planning state $\boldsymbol{s}_k$ by the neural network and $Q(\boldsymbol{s}_k, \boldsymbol{a}_k)$ is the actual transfer time required to reach GEO from planning state $\boldsymbol{s}_k$.

The implementation of deep neural networks with Q-learning is known as Deep Q-Learning. Only the planning states are taken as the input to the network and the target are the Q-values or the transfer time. Then, the neural network is trained with those input and target values to create the model of the dynamic model used for low-level planning problem of minimizing the objective function which is a convex combination of components. Commonly, neural networks are used for non-linear function approximation in regression or classification tasks. A typical neural network consists of multiple layers. These consist of an input layer and an output layer with one or more hidden layers between them. Neurons present in the hidden layer process incoming signals from the input neurons and then transmits a resulting signal to its subsequent neurons to the next layer. Artificial neural network with three inputs and a hidden layer and one output can be illustrated by Figure 3.1.

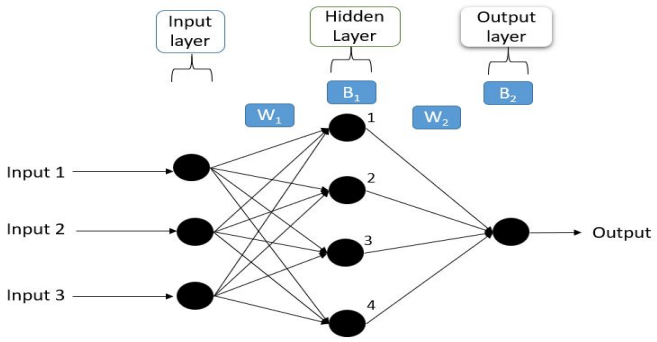

Figure 3.2: An artificial neural network

The optimized output of the neural network is:

$$\hat{Q}^{\star}(\boldsymbol{s}_k, \boldsymbol{a}_k) = f_2(W_2(f_1(W_1\boldsymbol{I} + B_1)) + B_2), \tag{3.14}$$

where $W_1 \in \mathbb{R}^{3\times4}$ which represents the weight matrix W_1 having row vector as the number of input and column vector as the number of hidden neuron whereas $W_2 \in \mathbb{R}^{4\times1}$ which represents the weight matrix W_2 having row vector as the number of hidden neurons and column vector as number of output neurons, $\boldsymbol{I} \in \mathbb{R}^3$ where $\boldsymbol{I}$ represents the input vector, $B_1 \in \mathbb{R}^{1\times4}$ and $B_2 \in \mathbb{R}^{1\times1}$ which represents the bias matrices for hidden layer and output layer respectively which are used to adjust the output along with the weighted sum of the inputs to the neuron. The dimensions of the weight matrix of a particular layer are such that it can take the input of that layer and transform it into the size of the next layer. f_1 and f_2 are defined as the activation functions for the hidden layer and output layer respectively, which are used to add nonlinearities to the neural network in an element-wise manner. The most popular activation function options include sigmoid, tanh, rectified linear unit (ReLu). A neural network can learn from data so it can be trained to recognize patterns, classify data, and forecast future events. The behavior of the neural network is defined by the *weights* or by the strength of individual neuron is connected which each other in each layer. These weights are called the synaptic weights. These weights are adapted by the algorithm used for learning in a way to minimize the performance function that maps the behaviour

between the input and output. This process is known as the training of the neural network. Before analyzing any typical data, it is normally divided into three data sets. These include Training data set, Validation data set and Testing data set. The training data set is used to train the neural network, whereas the validation data set is mainly used to avoid overfitting. If the training data set performance is more than the validation set, then the situation is known as overfitting. This data is used to refine the neural network. The testing data set used only for testing the final solution in order to confirm the actual predictive power of the network. The training is stopped when either of the following conditions is satisfied:

i) Validation data set error (performance) reaches minimum

ii) Convergence of the performance function, or

iii) Number of epochs (iterations) reaches maximum.

For the network to learn, various training algorithms are used to model these weights and biases into generating more accurate output. The aim of these algorithms are to minimize the objective function with respect to the weights and biases within the network. In this study, Levenberg-Marquardt optimizer is used for training the network due to its faster convergence rate and its robustness[60, 61], which is available in the Neural Network Toolbox in MATLAB. Training of these neural networks is executed on a desktop computer that runs the Windows 7 Enterprise 64-bit operating system on an Intel Core i5-4570 CPU 3.2 GHz with 8 GB of RAM.

3.5 Data Used for Neural Network

Low-thrust orbit-raising trajectories are optimized by the dynamic model in the reference 10 for various transfers from different initial orbits to geostationary equatorial orbit (GEO). These initial orbits include geostationary transfer orbit (GTO), Sub-GTO, Super-GTO. For data generation, 100 orbit-raising transfer problems are generated for linearly spaced choices of $w_{h,k}$ between 0.1 and 0.99 for all the orbit-raising scenarios. To train the neural network for planar transfer, input vector $\boldsymbol{I}_P$ whose components are magnitude of

angular momentum (h), magnitude of eccentricity vector (e), w_h and w_e, and therefore, is represented as

$$\boldsymbol{I}_P = \left[h_{k+1}, e_{k+1}, w_{h,k+1}, w_{e,k+1} \right] \tag{3.15}$$

whereas the output is taken as the time required by the spacecraft to transfer from the current planning state $\boldsymbol{s}_k$ to reach GEO after every revolution which is represented by Eq.(3.14).

For nonplanar transfer, 24 orbit-raising transfer problems are generated for linearly spaced choices of $w_{h,k}$ and $w_{hxy,k}$ between 0.1 and 0.99 for all the orbit-raising scenarios. Here, input vector $\boldsymbol{I}_{NP}$ whose components are angular momentum magnitude (h), inclination angle ($i = \arcsin(\frac{\sqrt{h_x^2+h_y^2}}{h})$), magnitude of eccentricity vector (e), w_h, w_{hxy} and w_e and therefore represented as

$$\boldsymbol{I}_{NP} = \left[h_{k+1}, i_{k+1}, e_{k+1}, w_{h,k+1}, w_{hxy,k+1}, w_{e,k+1} \right] \tag{3.16}$$

whereas output, similar to planar case, is the time required by the spacecraft to transfer from the current planning state to reach GEO after every revolution which is represented by Eq.(3.14).

For both planar and nonplanar transfers, the number of trainable data points generated are given by Table 3.1.

Table 3.1: Number of Trainable Data points

Initial Orbit	Data points
Planar Transfers	
GTO to GEO	21708
Sub-GTO to GEO	26804
Super-GTO to GEO	16183
Nonplanar Transfers (i =28.5 deg)	
GTO to GEO	75459
Sub-GTO to GEO	88653
Super-GTO to GEO	59947

3.6 NN selection

The main parameter considered to train the neural network in this work is the neurons count present in the only hidden layer. Input data to the neural network is divided randomly

into the training set, validation set and testing set in the percentages 70, 15 and 15. That means 70 percent of the input data is considered for training, rest 30 percent of the data is divided equally for validation and testing. Furthermore, the training is done by LM algorithm with random initialized synaptic weights of the input neurons. Activation function for the hidden layer is taken as tanh (f_1) and for output layer as linear (f_2) which are the default settings for the activation function in the LM training in the MATLAB toolbox. The performance of the trained network is evaluated by the mean squared error (MSE) which is given by:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{z}_i - z_i)^2 \tag{3.17}$$

where n is the number of data points or samples. This MSE is analogous to the loss function for the deep neural network which is given by Eq.(3.13) where ($\hat{z}_i$) is the predicted value at ith data point which corresponds to $Q(\boldsymbol{s}_k, \boldsymbol{a}_k; \theta)$ and (z_i) is the Target or observed values at the ith data point corresponds to $Q(\boldsymbol{s}_k, \boldsymbol{a}_k)$. Furthermore, accuracy is then calculated which evaluates the percentage of correct predictions by the trained neural network that can be easily derived from: $Accuracy = 1 - MSE$.

The neural network which gives least MSE with less number of hidden neurons is selected for that particular scenario.

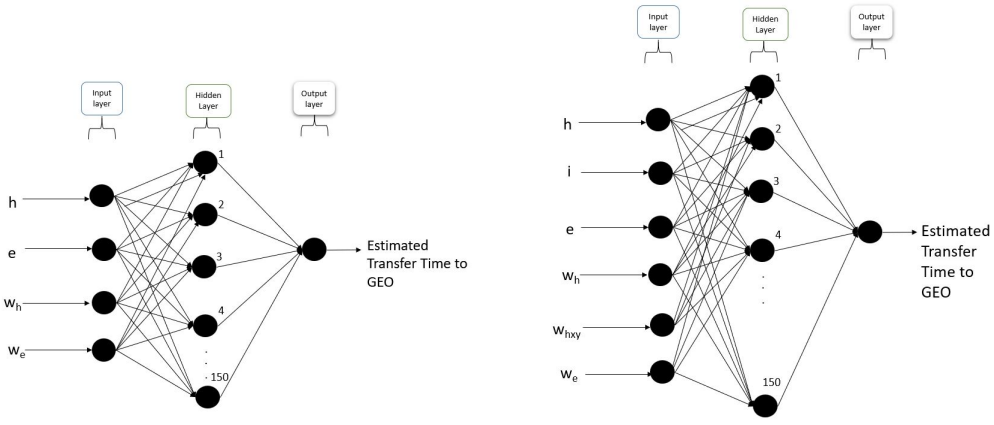

Figure 3.3: Neural Network Architecture for Planar and Nonplanar GTO transfer

3.6.1 GTO transfer

Data for GTO transfer is generated considering perigee altitude of 250 km and the apogee altitude of 35786 km. Initial value of the mass of the spacecraft is considered to be 5000 kg. For planar transfer, 4 inputs and 1 output are taken for training, as explained in the earlier section. Different neural networks are trained with different number of hidden layer neurons on a trial and error basis which includes 10, 50, 100, 120, 130, 150, 200 and 380. The number of training iterations are taken as 5000. Moreover, the training is stopped by the LM algorithm when one of the conditions is satisfied, which in this case, is network performance on the validation vectors fails to improve or remains the same for max_fail (default value = 6) epochs in a row. It is found that the network with 150 neurons in the hidden layer gives the least MSE of 0.0025 with an accuracy of 99.75%.

For nonplanar transfer, data is generated with similar perigee and apogee altitude as in the planar case but with an inclination angle of 28.5 degrees. Different NNs are trained with multiple values of hidden neurons. Nonplanar NN selection is based on the training with the least number of hidden neurons and is stopped when the performance of the NN has reached the same as the planar neural network. It is found that NN is trained with 150 hidden neurons is selected, 6 inputs and 1 output with the stopping condition set as minimum MSE same as the planar case which is 0.25%. For both GTO planar transfer and nonplanar transfer, it took approximately 1 hr and 3 hrs, respectively to train the neural networks. Once trained, neural networks for both planar transfer and nonplanar transfer are evaluated with the simulations and whose results are compared with Reference [9].

3.6.2 Sub-GTO Transfer

Similarly for Sub-GTO transfer, data for is generated considering perigee altitude and apogee altitude as 250 km and 30000 km, respectively. Initial value of the mass of the spacecraft is taken out to be 5000 kg. For planar transfer, 4 inputs and 1 output are taken for training. Different neural networks are trained with different number of hidden layer neurons on a trial and error basis. The number of training iterations are taken as 5000.

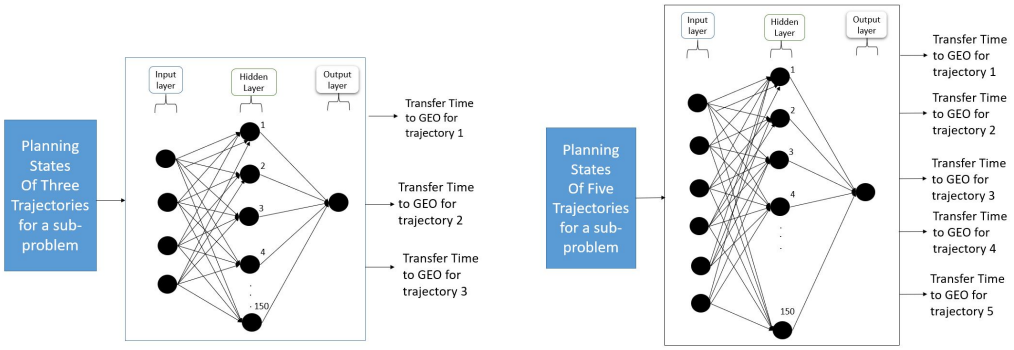

Figure 3.4: Deep Q-Learning Architecture for Planar and Nonplanar GTO transfer

Training is stopped by the LM algorithm when one of the conditions is satisfied, which in this case, is network performance on the validation vectors fails to improve or remains the same for max_fail (default value = 6) epochs in a row. It is found that the network with 150 neurons in the hidden layer gives the least MSE of 0.0047 with an accuracy of 99.53%.

For nonplanar transfer, data is generated with similar perigee and apogee altitude as in the planar case but with an inclination angle of 28.5 degrees. Different NNs are trained with multiple values of hidden neurons. Nonplanar NN selection is based on the training with the least number of hidden neurons and is stopped when the performance of the NN has reached same as the planar neural network. It is found that the neural network with 120 hidden neurons is selected, 6 inputs and 1 output with the stopping condition set as minimum MSE same as the planar case which is 0.47%. For both Sub-GTO planar transfer and nonplanar transfer, it took approximately 1.5 hrs and 6 hrs, respectively to train the neural networks. After training the networks for both planar and nonplanar transfer, they are evaluated by integrating it with the dynamic model.

3.6.3 Super-GTO Transfer

For Super-GTO transfer, data for is generated considering perigee altitude of 250 km and the apogee altitude of 45000 km. Initial value of the mass of the spacecraft is considered to be 5000 kg. Likewise for planar transfer, 4 inputs and 1 output are taken for training. Different neural networks are trained with different number of hidden layer neurons on a trial and error basis. The number of training iterations are taken as 5000 which is same for

GTO case and Sub-GTO case. Training is stopped by the LM algorithm when one of the conditions is satisfied. It is found that the network with 130 neurons in the hidden layer gives the least MSE of 0.0091 with an accuracy of 99.09%.

For nonplanar transfer, data is generated with similar perigee and apogee altitude as in the planar case but with an inclination angle of 28.5 degrees. Different NNs are trained with multiple values of hidden neurons. Nonplanar NN selection is based on the training with the least number of hidden neurons and is stopped when the performance of the NN has reached same as the planar neural network. It is found that the neural network with 100 hidden neurons is selected, 6 inputs and 1 output with the stopping condition set as minimum MSE same as the planar case which is 0.91%. For both Super-GTO planar transfer and nonplanar transfer, it took approximately 0.5 hrs and 3 hrs, respectively to train the neural networks. After training the networks for both planar and nonplanar transfer, they are evaluated by integrating it with the dynamic model.

Table 3.2: Neural networks Training Results Summary

Initial Orbit	Input-Hidden-Output	Activation func of hidden layer	Activation func of output layer	Training time (hours)	MSE
Planar Transfers					
GTO	4-150-1	tanh	linear	1	0.0025
Sub-GTO	4-150-1	tanh	linear	1.5	0.0047
Super-GTO	4-130-1	tanh	linear	0.5	0.0091
Nonplanar Transfers					
GTO	6-150-1	tanh	linear	3	0.0025
Sub-GTO	6-120-1	tanh	linear	6	0.0047
Super-GTO	6-100-1	tanh	linear	3	0.0091

State variables and Planning states (both planar and nonplanar) are initialized for low-level trajectory optimization scheme and high-level planning problem, respectively. Given the current planning state, Q-values/transfer time to reach GEO are predicted by the trained planar and nonplanar neural networks for all the considered actions/trajectories from that planning state. The action corresponding to the maximum Q-value (minimum transfer time

to reach GEO) is selected and executed. Furthermore, the reward value or the transfer time by the spacecraft to reach the state variables for the next subproblem is calculated for that action which is given as a function of state variables computed at the end of a revolution by low-level optimization scheme as given by Eq.(3.7). These computed state variables become the initial state variables and form the planning state for the next optimization subproblem. This sequence of selecting an action by the NN that gives the minimum transfer time to reach GEO from a planning state is continued until the spacecraft reaches the GEO. The working of the planning algorithm is depicted by flowchart in Figure 3.5.

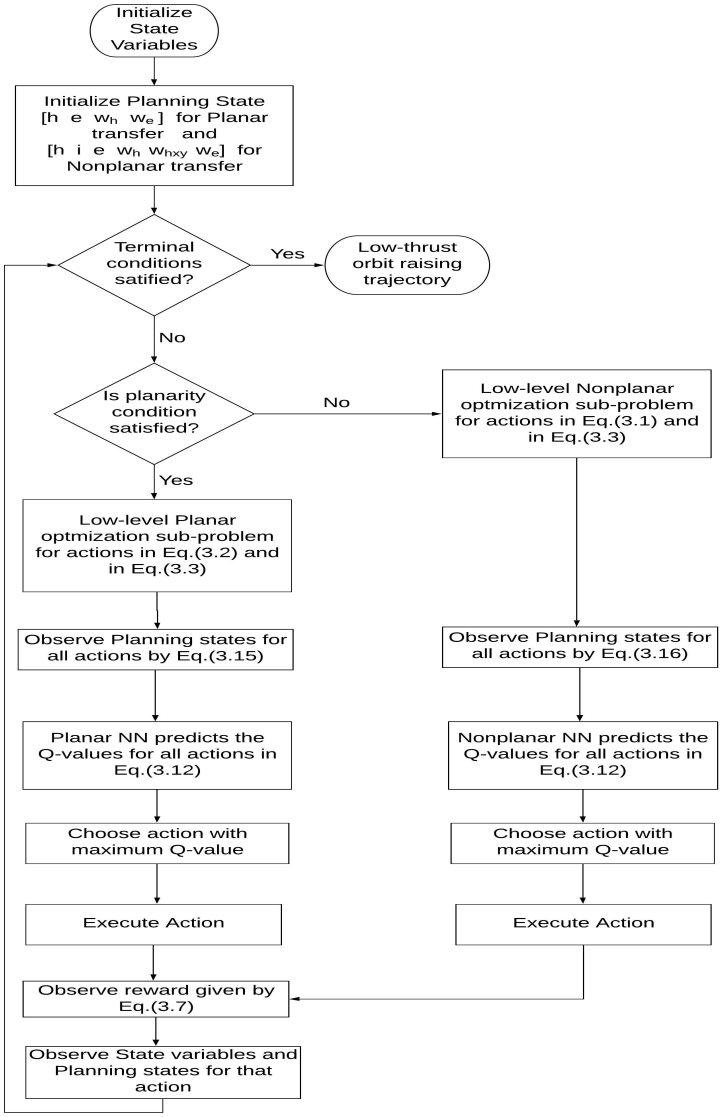

Figure 3.5: Planning Algorithm flow chart

3.7 Generalized Neural Network

Instead of using different NNs for different mission scenarios, a single generalized neural network is introduced for all the considered scenarios. This generalized neural network is trained on the data for all the equatorial orbits used earlier.

3.7.1 Data for Single NN

As mentioned, data for this single NN is generated by combining the data for all the equatorial orbits namely GTO, Sub-GTO and Super-GTO, for both planar and nonplanar transfers. For both planar and nonplanar transfers, six planning variables are taken in the input vector $\boldsymbol{I} \mapsto \boldsymbol{s}_k$ which is represented as

$$\boldsymbol{I} = \left[h_{k+1}, i_{k+1}, e_{k+1}, w_{h,k+1}, w_{hxy,k+1}, w_{e,k+1} \right] \tag{3.18}$$

where values for i_{k+1} and $w_{hxy,k+1}$ are taken as zero for planar transfers. The output for the generalized NN is considered same as before, that is, transfer time required to reach GEO from the current planning state $\boldsymbol{s}_k$. The total number of trainable data points is taken as 288,754.

3.7.2 Generalized NN selection

Like before, the only parameter considered for training the neural network is the hidden neurons count. NN is trained in the similar fashion, that is, by changing the number of hidden neurons of the network until the network which provides least MSE given by Eq.(3.17) with less number of hidden neurons is found.

Different neural networks are trained with different number of hidden layer neurons on a trial and error basis. The number of training iterations are taken as 5000. Training is stopped by the LM algorithm when one of the conditions is satisfied, which in this case, is network performance on the validation vectors fails to improve or remains the same for max_fail (default value = 6) epochs in a row. It is found that the network with 170 neurons in the hidden layer gives the least MSE of 0.1561 with an accuracy of 84.39%. It took

approximately 20 hrs to train the generalized neural network. Once trained, this single neural network is used for optimizing the the relative weights of the objective function in Eq.(3.4) and Eq.(3.6) like the networks used in the previous section. This generalized NN is integrated with low-level optimization scheme whose working is shown by the flowchart in Figure 3.6.

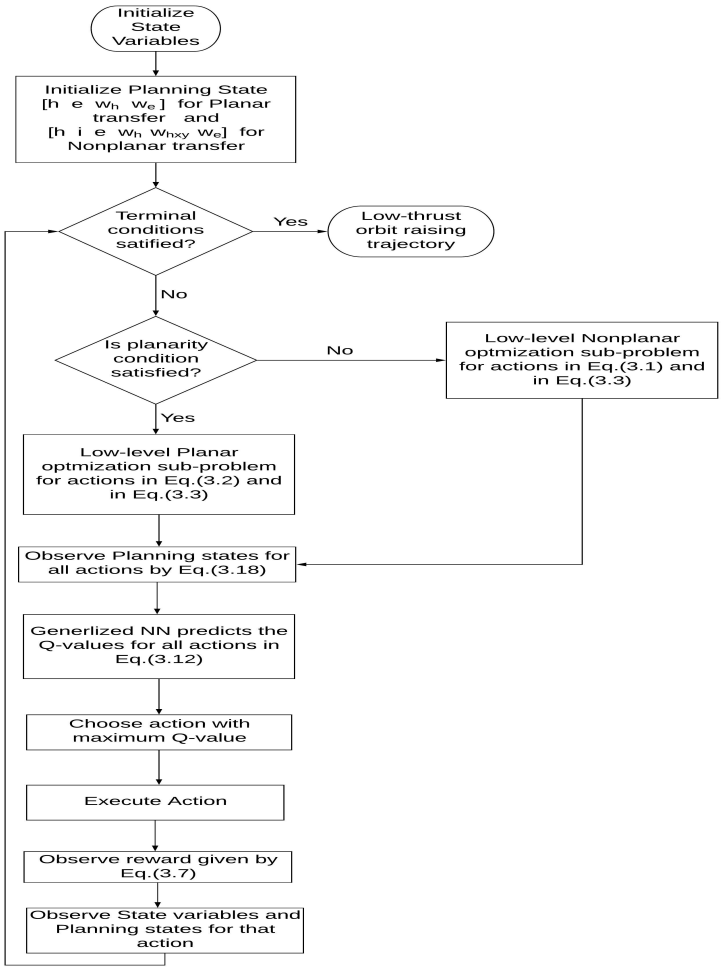

Figure 3.6: Planning Algorithm flow chart for Generalized NN

CHAPTER 4

ORBIT-RAISING PROBLEM SIMULATION RESULTS

In this chapter, numerical results for orbit-raising maneuvers starting from equatorial orbits (GTO, Sub-GTO, and Super-GTO) are demonstrated. For low-level planning problem, an engine with four BPT-4000 Hall thrusters is considered for low-thrust orbit-raising trajectory generation for all mission scenarios. The choice of selecting BPT-4000 thruster is due to the fact that it can be used for both small and large class of telecommunication satellites.

4.1 Machine Learning Results

For all examples in high-level planning problem, neural networks for each scenario are created and trained with the data generated using the dynamic model described by low-level trajectory optimization problem. As explained in the previous chapters, the data includes the planning variables and the transfer time required to reach GEO corresponding to those planning variables. Neural networks are trained using Levenberg-Marquardt optimizer which is available in the Neural Network Toolbox in MATLAB.

4.1.1 Planar Transfers

First, planar transfer scenarios having equatorial orbits as initial orbits with no inclination are considered. The objective function for the planar transfer is given by Eq.(3.6). Initially, $w_{h,k}$ and $w_{e,k}$ values are taken as 0.75 and 0.25, respectively. All four thrusters together generates a thrust value equal to 1.16 N. Consequently, they provide a specific impulse of 1788 s. For all cases, the initial guess for control input α is taken as zero. Stopping criteria for high-level planning algorithm is defined when the terminal conditions for low-level optimization problem given by Eq.(3.8), Eq.(3.9) and Eq.(3.10) are met where $e_{tol} = 0.01$ and $a_{tol} = 0.01$.

Example 7: GTO to GEO Planar Transfer

First, for GTO planar transfer $w_{h,k}$ is initialized as 0.75. Then, three actions/trajectories given by Eq.(3.2) are considered. Q-values or the time to reach GEO from all the planning states are estimated by the trained neural network for all the three trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. Then the transfer time for that sub-problem is calculated by Eq.(3.7) with the obtained planning state.

Likewise, the network selects the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state. This goes until the terminal conditions defined by low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-size (Δw_h) which remains constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.75$ and $\Delta w_h = 0.001$ gives the transfer time of 109.98 sidereal days with 174 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.75$ for every sub-problem with a transfer time of 111.74 sidereal days with 185 revolutions given in [9]. Simulation results for this planar case are illustrated by Figure 4.1.

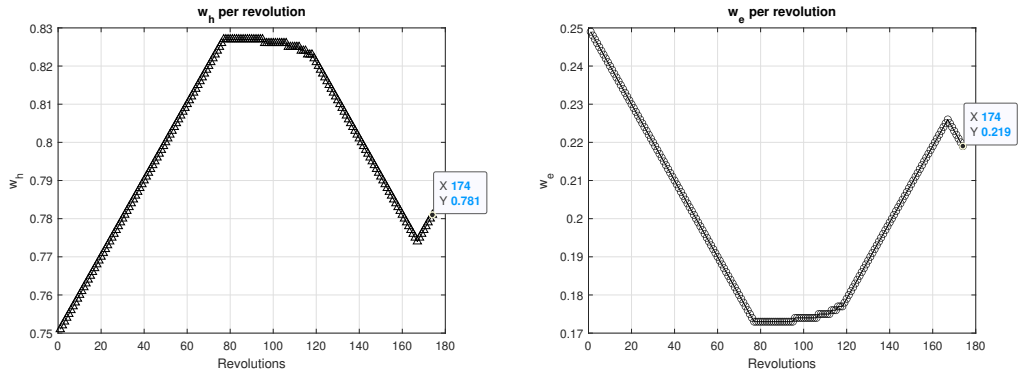

Figure 4.1: Variation of w_h and w_e for every revolution for GTO Planar case

Example 8: Sub-GTO to GEO Planar Transfer

Similarly, for Sub-GTO planar transfer $w_{h,k}$ is initialized as 0.75. Then, three actions/trajectories are considered that are given by Eq.(3.2). Q-values or the transfer time to reach GEO from all the planning states are estimated by the trained neural network for all the three trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. Then the transfer time for that sub-problem is calculated by Eq.(3.7) with the obtained planning state.

Likewise, the network selects the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state. This goes until the terminal conditions defined by low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-size (Δw_h) which remains constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.75$ and $\Delta w_h = 0.008$ gives the transfer time of 115.10 sidereal days with 204 revolutions. When compared with initial and constant value of $w_{h,k} = 0.75$ for every sub-problem, transfer time came out to be 118.79 sidereal days with 227 revolutions as shown in [9]. Simulation results for this planar case are illustrated by Figure 4.2.

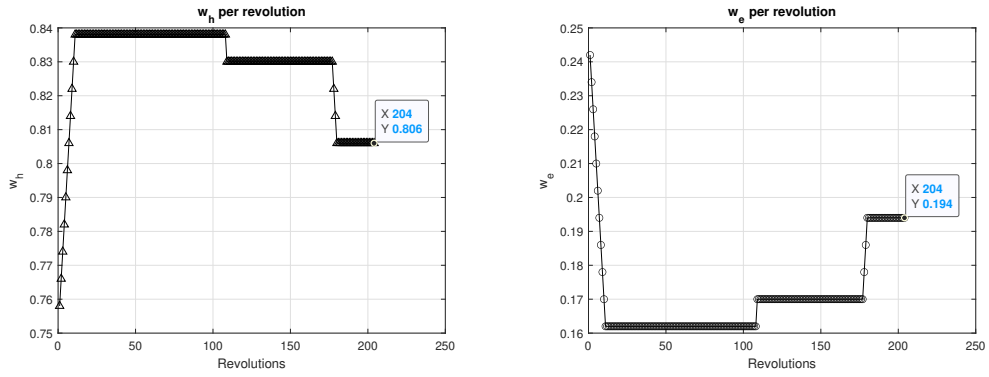

Figure 4.2: Variation of w_h and w_e for every revolution for Sub-GTO Planar case

Example 9: Super-GTO to GEO Planar Transfer

Similarly, for Super-GTO planar transfer $w_{h,k}$ is initialized as 0.75. Then, three actions/trajectories given by Eq.(3.2) are considered. Q-values or the transfer time to reach GEO from all the planning states are estimated by the trained neural network for all the three trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. Then the transfer time for that sub-problem is calculated by Eq.(3.7) with the obtained planning state.

Likewise, the network selects the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state. This goes until the terminal conditions defined by low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-size (Δw_h) which remains constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.75$ and $\Delta w_h = 0.001$ gives the transfer time of 103.70 sidereal days with 136 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.75$ for every sub-problem with transfer time of 104.02 sidereal days with 140 revolutions given in [9]. Figure 4.2 represents results for the Super-GTO planar case.

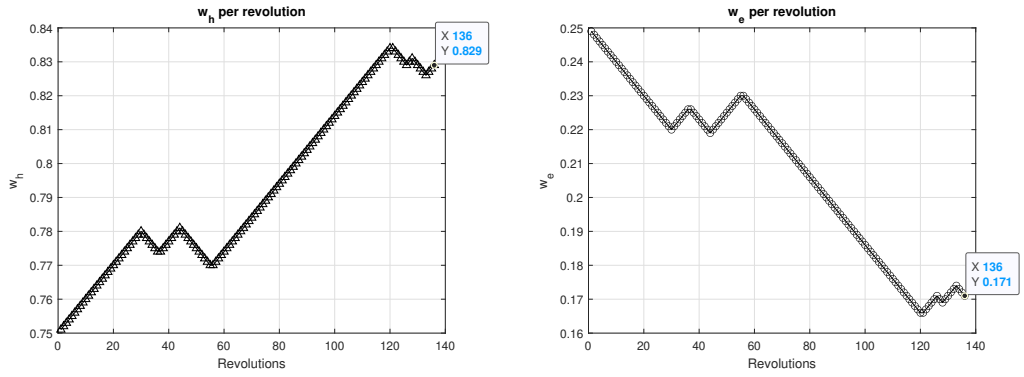

Figure 4.3: Variation of w_h and w_e for every revolution for Super-GTO Planar case

4.1.2 Nonplanar Transfers

For nonplanar transfers, 28.5 degrees is taken as the initial value for inclination angle. The objective function for the nonplanar cases is given by Eq.(3.4) where initial values of $w_{h,k}$, $w_{e,k}$ and $w_{hxy,k}$ are set as 0.5, 0.1 and 0.4, respectively. The engine is composed with four BPT-4000 thrusters. The generated thrust has a value of 1.16 N, similar to the planar case. Consequently, the specific impulse 1786 s is provided to the spacecraft. Stopping conditions for high-level planning algorithm are defined when the terminal conditions for low-level optimization problem are met which are $e_{tol} = 0.01$, $i_{tol} = 0.01$ and $a_{tol} = 0.01$.

Example 10: GTO to GEO Nonplanar Transfer

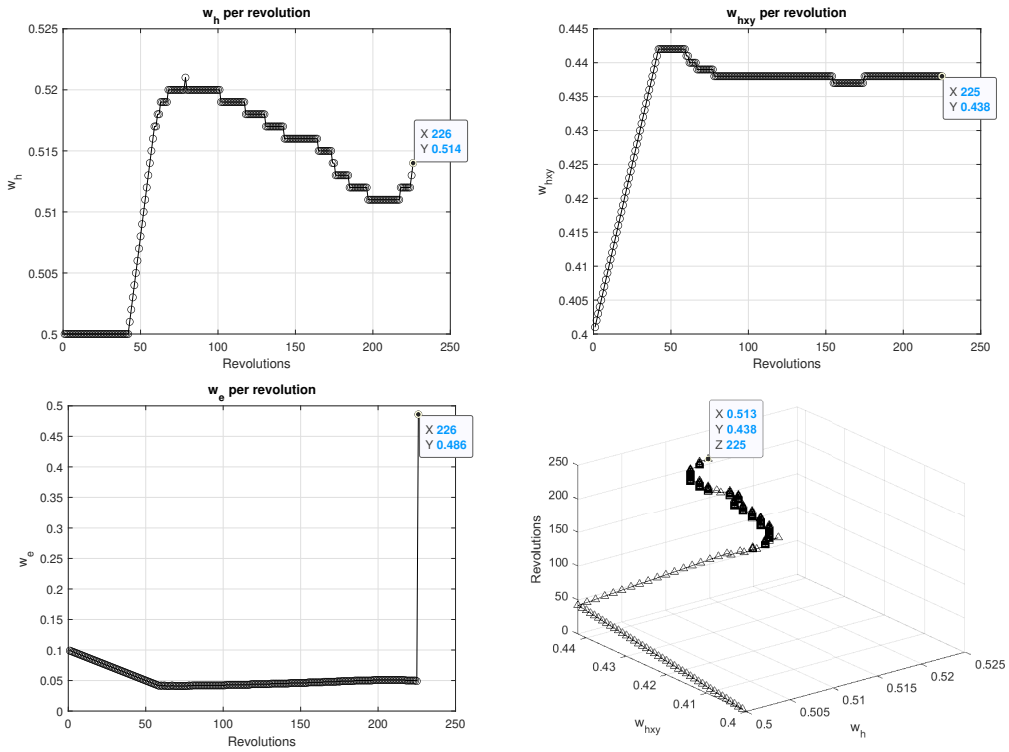

Figure 4.4: Variation of relative weight for every revolution for GTO Nonplanar case

For GTO nonplanar transfer, initial values of $w_{h,k}$ and $w_{hxy,k}$ are taken as 0.5 and 0.4 respectively. Then, five actions/trajectories given by Eq.(3.1) are considered. Q-values or the transfer time to reach GEO from states are estimated by NN for all the five trajectories and the one trajectory with minimum transfer time $[\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))]$ is selected for that sub-

problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. The transfer time for that sub-problem is then calculated by Eq.(3.7).

The network keeps on selecting the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state until the terminal conditions defined by the low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-sizes (Δw_h and Δw_{hxy}) which remain constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ with $\Delta w_h = 0.001$ and $\Delta w_{hxy} = 0.001$ gives the transfer time of 158.95 sidereal days with 226 revolutions is less as compared with initial and constant value of $w_h = 0.5$ and $w_{hxy,k} = 0.4$ for every sub-problem with transfer time of 167.72 sidereal days with 255 revolutions. Simulation results for this nonplanar case is illustrated by Figure 4.4. There is a jump in the value of $w_{e,k}$ at the end for the nonplanar case. This is because the maneuver becomes planar for that sub-problem and value of $w_{hxy,k}$ becomes zero. This means there is a switching of NNs from nonplanar NN to planar NN for which the values of $w_{h,k}$ is selected by the trained planar transfer neural network giving corresponding relative weight $w_{e,k}$.

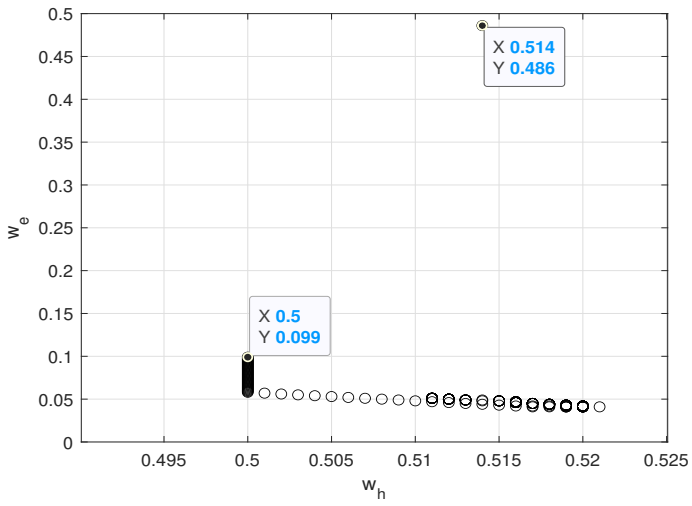

Figure 4.5: Values of w_h vs w_e for Nonplanar GTO Transfer

Example 11: Sub-GTO to GEO Nonplanar Transfer

For Sub-GTO nonplanar transfer, initial values of $w_{h,k}$ and $w_{hxy,k}$ are taken as 0.5 and 0.4 respectively. Then, five actions/trajectories given by Eq.(3.1) are considered. Q-values or the transfer time to reach GEO from states are estimated by NN for all the five trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. The transfer time for that sub-problem is then calculated by Eq.(3.7).

The network keeps on selecting the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state until the terminal conditions defined by the low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-sizes (Δw_h and Δw_{hxy}) which remain constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ with $\Delta w_h = 0.004$ and $\Delta w_{hxy} = 0.004$ gives the transfer time of 170.09 sidereal days with 284 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ for every sub-problem with transfer time of 173.28 sidereal days with 297 revolutions. Simulation results for this nonplanar case are illustrated by Figure 4.6.

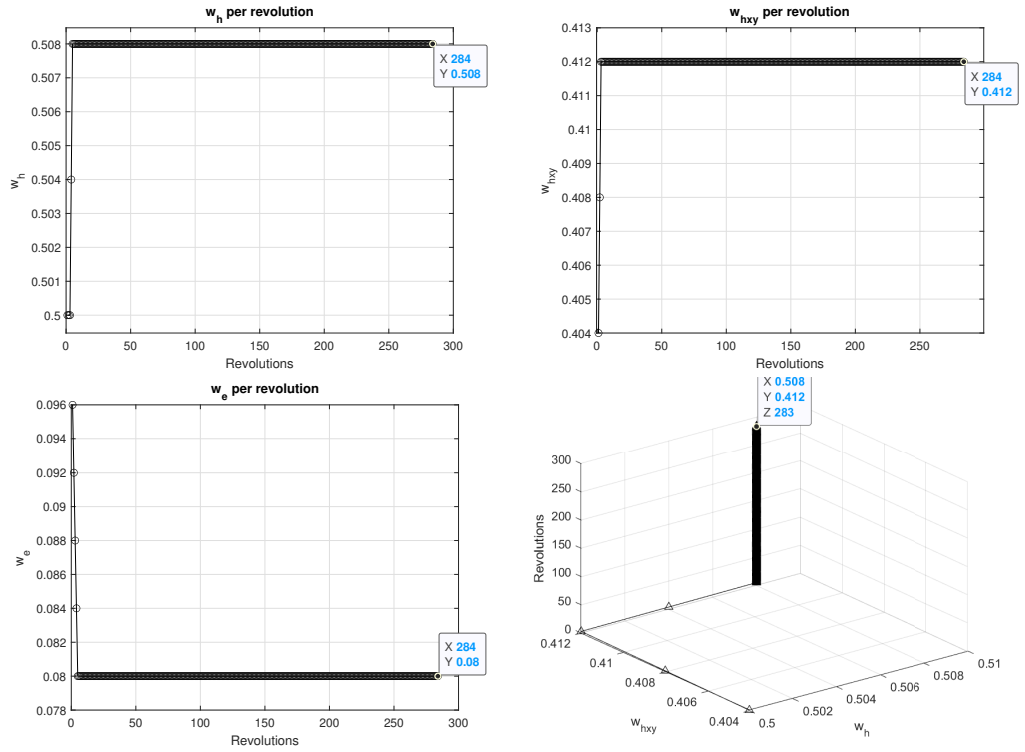

Figure 4.6: Variation of relative weights for every revolution for Sub-GTO Nonplanar case

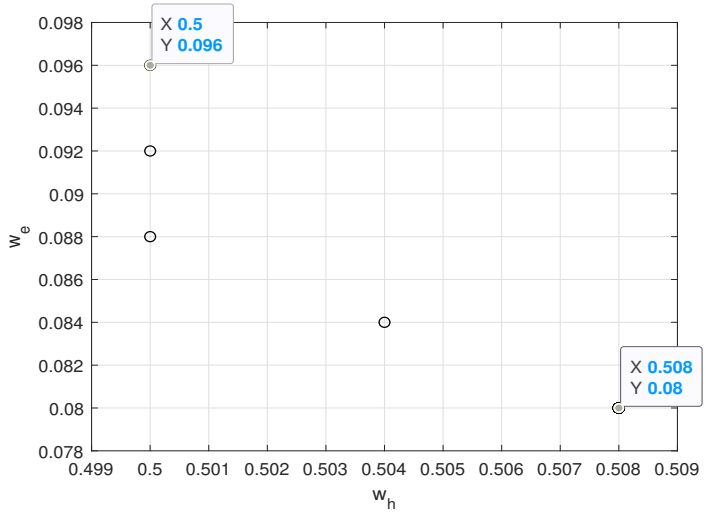

Figure 4.7: Values of w_h vs w_e for Nonplanar Sub-GTO Transfer

Example 12: Super-GTO to GEO Nonplanar Transfer

For Super-GTO nonplanar transfer, initial values of w_h and w_{hxy} are taken as 0.5 and 0.4 respectively. Then, five actions/trajectories given by Eq.(3.1) are considered. Q-values or

the transfer time to reach GEO from states are estimated by NN for all the five trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. The transfer time for that sub-problem is then calculated by Eq.(3.7).

The network keeps on selecting the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state until the terminal conditions defined by the low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-sizes (Δw_h and Δw_{hxy}) which remain constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ with $\Delta w_h = 0.009$ and $\Delta w_{hxy} = 0.009$ gives the transfer time of 153.92 sidereal days with 194 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ for every sub-problem with transfer time of 158.16 sidereal days with 205 revolutions. Figure 4.9 represents the simulation results for Super-GTO nonplanar transfer.

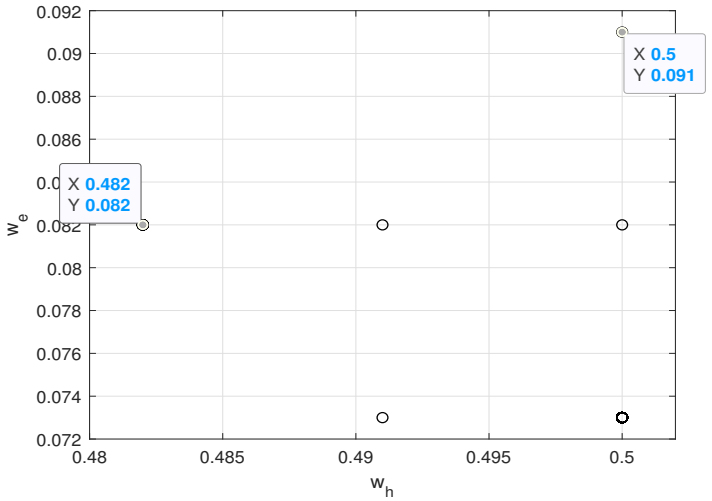

Figure 4.8: Values of w_h vs w_e for Nonplanar Super-GTO Transfer

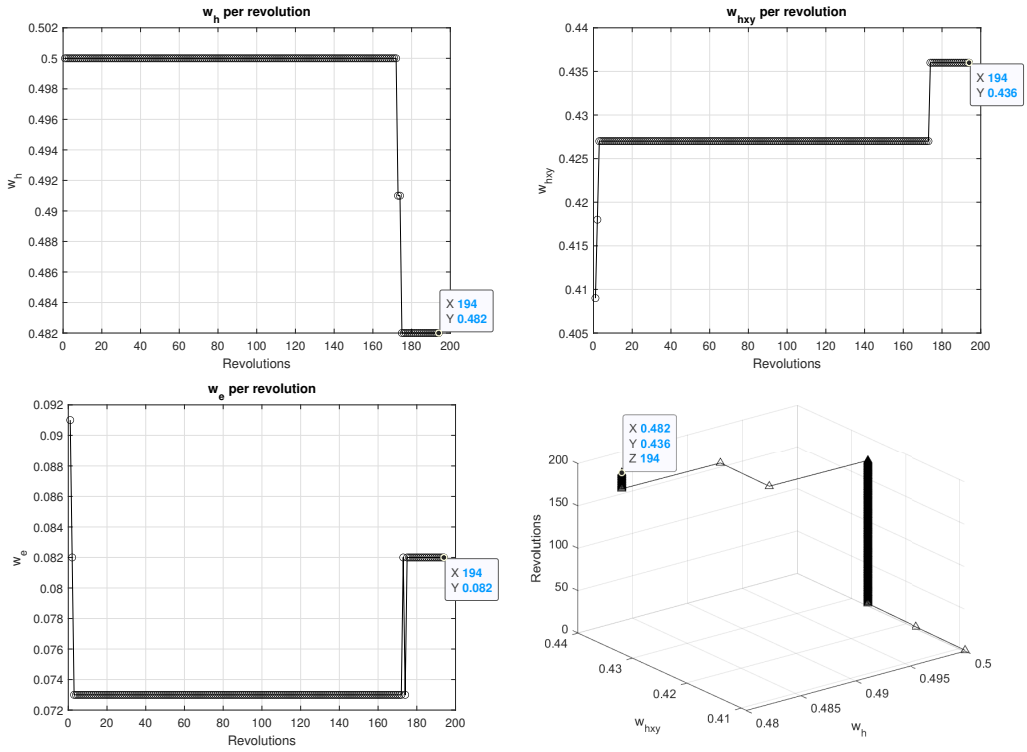

Figure 4.9: Variation of relative weights for every revolution for Super-GTO Nonplanar case

4.2 Single generalized NN Results

In this section, the results for high-level planning problem using a single neural network is discussed. For all low-thrust maneuvers, initial values of $w_{h,k}$, $w_{e,k}$ and $w_{hxy,k}$ in the objective function Eq.(3.4) and Eq.(3.6) are taken same as before, that is, 0.75, 0.25 and 0 respectively for planar transfer and 0.5, 0.1 and 0.4 for nonplanar transfers. Stopping criteria for high-level planning algorithm with single neural network remains the same which is defined when the terminal conditions for low-level optimization problem given by Eq.(3.8), Eq.(3.9) and Eq.(3.10) are met where $e_{tol} = 0.01$, $i_{tol} = 0.01$ and $a_{tol} = 0.01$.

4.2.1 Planar Transfers

Example 13: GTO to GEO Planar Transfer

The working of this single NN is same as the neural network trained specifically for GTO to GEO planar maneuver. First, for GTO planar transfer $w_{h,k}$ is initialized as 0.75. Then, three actions/trajectories given by Eq.(3.2) are considered. Q-values or the transfer

time to reach GEO from all the planning states are estimated by the trained single neural network for all the three trajectories and the one trajectory with minimum transfer time $[\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))]$ is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. Then the transfer time for that sub-problem is calculated by Eq.(3.7) with the obtained planning state.

Likewise, the network selects the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state. This goes until the terminal conditions defined by low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-size (Δw_h) which remains constant for each planning period. Trajectory with an initial value of $w_{h,k} = 0.75$ and $\Delta w_h = 0.004$ gives the transfer time of 108.79 sidereal days with 170 revolutions is less as compared with the initial and constant value of $w_{h,k} = 0.75$ for every sub-problem with a transfer time of 111.74 sidereal days with 185 revolutions given in [9]. Simulation results for this planar case example are illustrated by Figure 4.10.

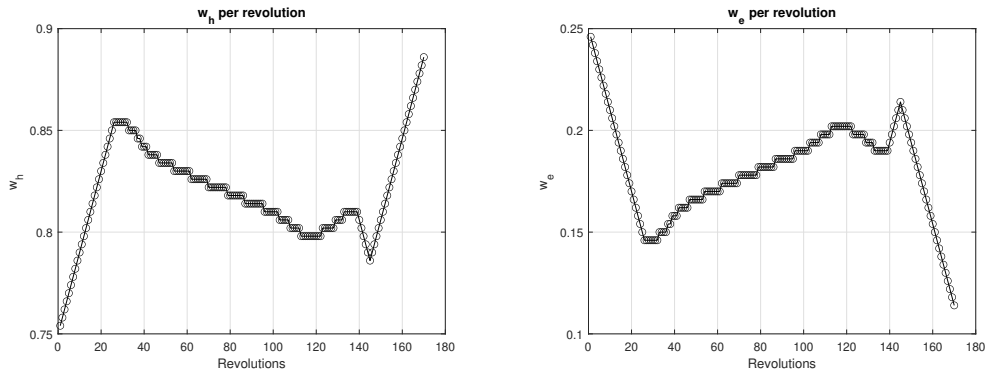

Figure 4.10: Generalized neural network results for GTO Planar case

Example 14: Sub-GTO to GEO Planar Transfer

Similarly, for Sub-GTO planar transfer $w_{h,k}$ is initialized as 0.75. Then, three actions/trajectories given by Eq.(3.2) are considered by the generalized network. Q-values or the transfer time to reach GEO from all the planning states are estimated by the trained single

neural network for all the three trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. Then the transfer time for that sub-problem is calculated by Eq.(3.7) with the obtained planning state.

Likewise, the network selects the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state. This goes until the terminal conditions defined by low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-size (Δw_h) which remains constant for each planning period. Trajectory with an initial value of $w_{h,k} = 0.75$ and $\Delta w_h = 0.003$ gives the transfer time of 114.13 sidereal days with 202 revolutions. When compared with the initial and constant value of $w_{h,k} = 0.75$ for every sub-problem, transfer time came out to be 118.79 sidereal days with 227 revolutions as shown in [9]. Simulation results for this planar case are illustrated by Figure 4.11.

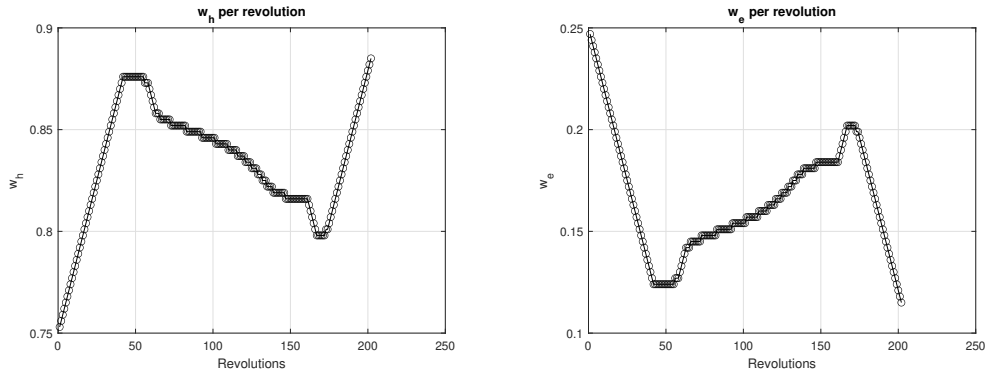

Figure 4.11: Generalized neural network results for Sub-GTO Planar case

Example 15: Super-GTO to GEO Planar Transfer

Similarly, for Super-GTO planar transfer $w_{h,k}$ is initialized as 0.75. Then, three actions/trajectories given by Eq.(3.2) are considered. Q-values or the transfer time to reach GEO from all the planning states are estimated by the trained single neural network for all the three trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$]

is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. Then the transfer time for that sub-problem is calculated by Eq.(3.7) with the obtained planning state.

Likewise, the network selects the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state. This goes until the terminal conditions defined by low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-size (Δw_h) which remains constant for each planning period. Trajectory with an initial value of $w_{h,k} = 0.75$ and $\Delta w_h = 0.001$ gives the transfer time of 103.34 sidereal days with 137 revolutions is less as compared with the initial and constant value of $w_{h,k} = 0.75$ for every sub-problem with transfer time of 104.02 sidereal days with 140 revolutions given in [9]. Figure 4.12 represents results for the Super-GTO planar case.

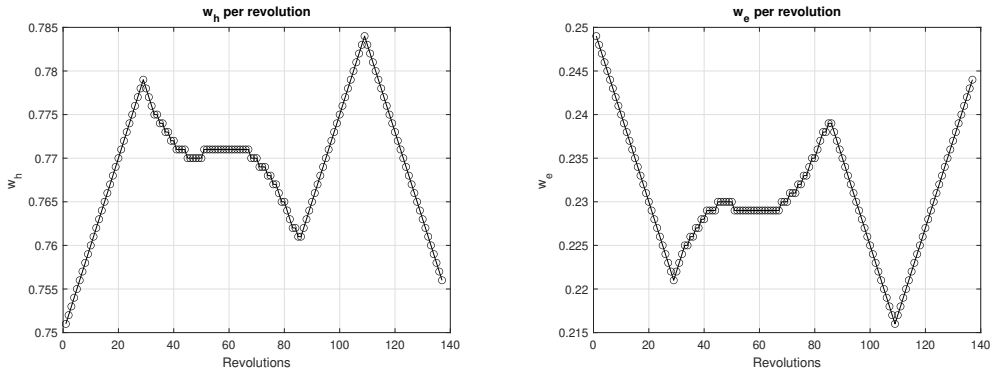

Figure 4.12: Generalized neural network results for Super-GTO Planar case

4.2.2 Nonplanar Transfers

Example 16: GTO to GEO Nonplanar Transfer

For GTO nonplanar transfer, initial values of $w_{h,k}$ and $w_{hxy,k}$ are taken as 0.5 and 0.4 respectively. Then, five actions/trajectories given by Eq.(3.1) are considered. Q-values or the transfer time to reach GEO from states are estimated by the generalized NN for all the five trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from

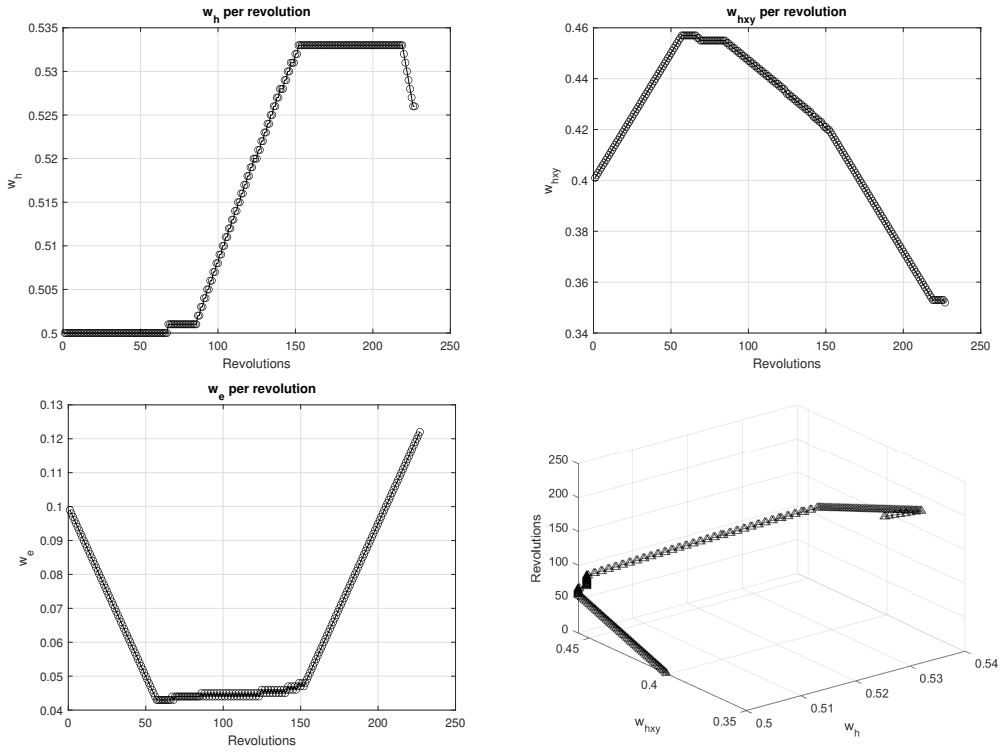

Figure 4.13: Generalized neural network results for GTO Nonplanar case

the dynamic model becomes the initial planning state for the next sub-problem. The transfer time for that sub-problem is then calculated by Eq.(3.7).

The network keeps on selecting the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state until the terminal conditions defined by the low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-sizes (Δw_h and Δw_{hxy}) which remain constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ with $\Delta w_h = 0.001$ and $\Delta w_{hxy} = 0.001$ gives the transfer time of 158.66 sidereal days with 227 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ for every sub-problem with transfer time of 167.72 sidereal days with 255 revolutions. Simulation results for this nonplanar case are illustrated by Figure 4.13. Note that by using a single neural network, there is no switching from NN used for

GTO nonplanar transfer to NN used for GTO planar transfer which is observed in Figure 4.4.

Example 17: Sub-GTO to GEO Nonplanar Transfer

For Sub-GTO nonplanar transfer, initial values of $w_{h,k}$ and $w_{hxy,k}$ are taken as 0.5 and 0.4 respectively. Then, five actions/trajectories given by Eq.(3.1) are considered. Q-values or the transfer time to reach GEO from states are estimated by generalized NN for all the five trajectories and the one trajectory with minimum transfer time [$\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))$] is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. The transfer time for that sub-problem is then calculated by Eq.(3.7).

The network keeps on selecting the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state until the terminal conditions defined by the low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-sizes (Δw_h and Δw_{hxy}) which remain constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ with $\Delta w_h = 0.0006$ and $\Delta w_{hxy} = 0.0006$ gives the transfer time of 164.95 sidereal days with 267 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ for every sub-problem with transfer time of 173.28 sidereal days with 297 revolutions. Simulation results for this nonplanar case are illustrated by Figure 4.14.

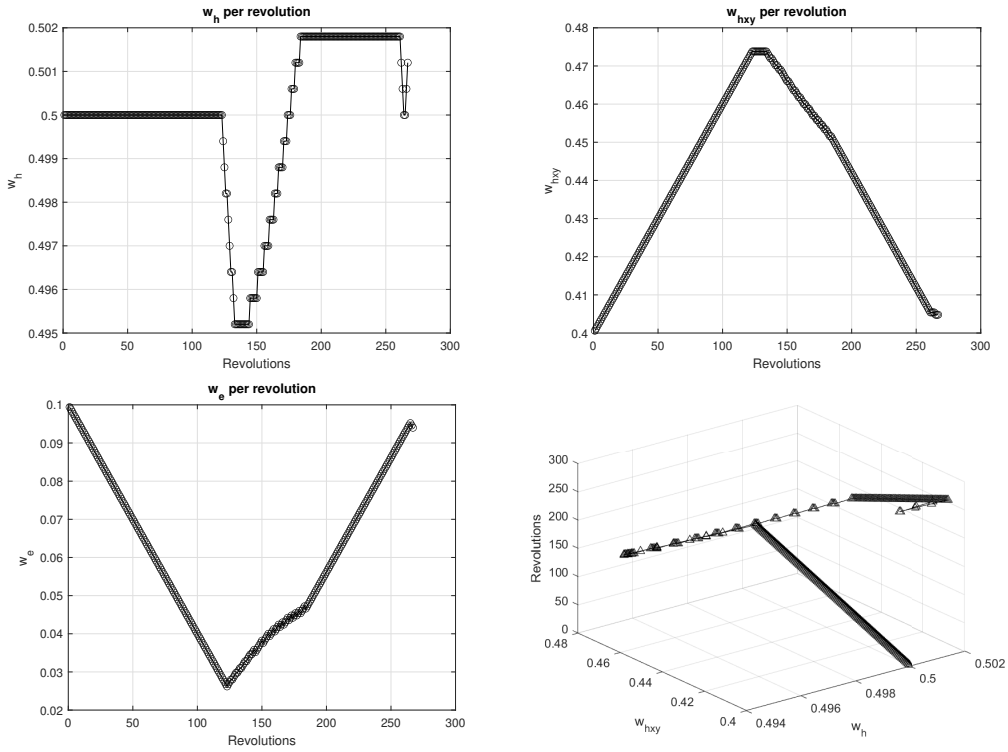

Figure 4.14: Generalized neural network results for Sub-GTO Nonplanar case

Example 18: Super-GTO to GEO Nonplanar Transfer

For Super-GTO nonplanar transfer, initial values of w_h and w_{hxy} are taken as 0.5 and 0.4 respectively. Then, five actions/trajectories given by Eq.(3.1) are considered. Q-values or the transfer time to reach GEO from states are estimated by NN for all the five trajectories and the one trajectory with minimum transfer time $[\max(-Q(\boldsymbol{s}_k, \boldsymbol{a}_k))]$ is selected for that sub-problem and the corresponding planning state which is obtained from the dynamic model becomes the initial planning state for the next sub-problem. The transfer time for that sub-problem is then calculated by Eq.(3.7).

The network keeps on selecting the planning state for every revolution or sub-problem which gives minimum transfer time to GEO from the current planning state until the terminal conditions defined by the low-level planning problem are satisfied. The proposed machine learning algorithm is tested for different values of user-defined step-sizes (Δw_h and Δw_{hxy}) which remain constant for each planning period. Trajectory with initial value of $w_{h,k} = 0.5$

and $w_{hxy,k} = 0.4$ with $\Delta w_h = 0.001$ and $\Delta w_{hxy} = 0.001$ gives the transfer time of 151.05 sidereal days with 186 revolutions is less as compared with initial and constant value of $w_{h,k} = 0.5$ and $w_{hxy,k} = 0.4$ for every sub-problem with transfer time of 158.16 sidereal days with 205 revolutions. Simulation results for Super-GTO nonplanar transfer are represented by Figure 4.15.

Table 4.1 represents the comparison of results for final transfer time required to reach GEO from all the considered initial orbit obtained by the neural networks with the results for final transfer time found in Ref.[9].

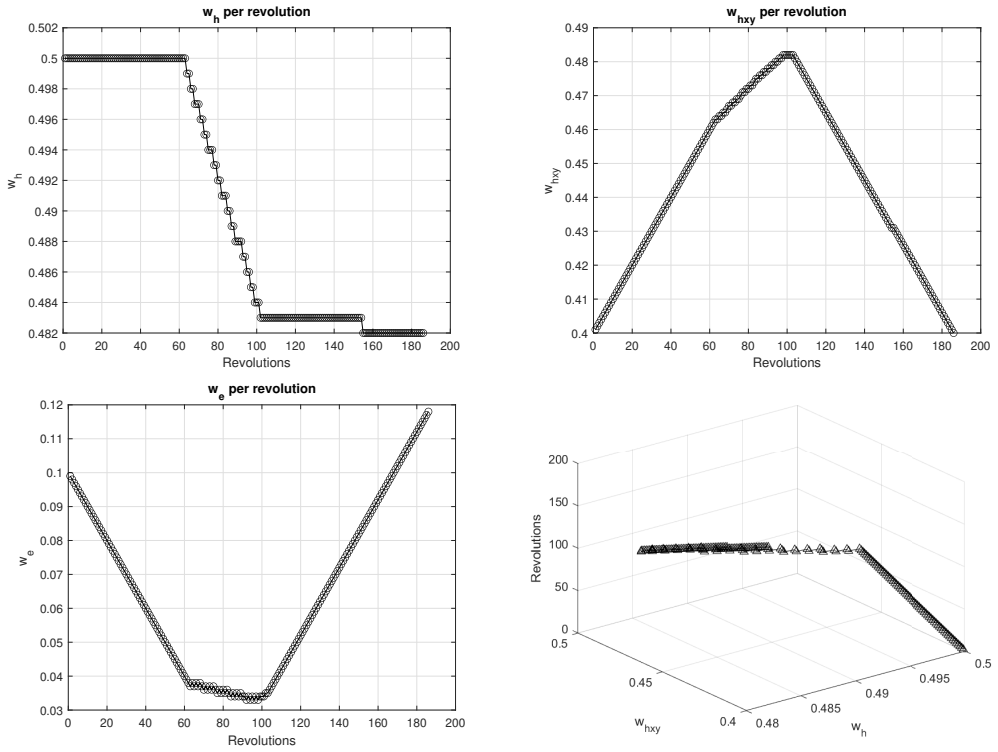

Figure 4.15: Generalized neural network results for Super-GTO Nonplanar case

Table 4.1: Summary of Results

Initial Orbit	Initial w_h	Initial w_e	Initial w_{hxy}	TT_{const} (sidereal days)	TT_{adapt} (sidereal days)	$\mathrm{TT}_{general}$ (sidereal days)
Planar Transfers						
GTO (250 km x 35786 km)	0.75	0.25	-	111.74 Ref.[9]	109.98	108.79
Sub-GTO (250 km x 30000 km)	0.75	0.25	-	118.79 Ref.[9]	115.10	114.13
Super-GTO (250 km x 45000km)	0.75	0.25	-	104.02 Ref.[9]	103.70	103.34
Nonplanar Transfers (i =28.5 deg)						
GTO (250 km x 35786 km)	0.5	0.1	0.4	167.72 Ref.[9]	158.95	158.66
Sub-GTO (250 km x 30000 km)	0.5	0.1	0.4	173.28 Ref.[9]	170.09	164.95
Super-GTO (250 km x 45000km)	0.5	0.1	0.4	158.16 Ref.[9]	153.92	151.05

CHAPTER 5

CONCLUSION

5.1 Thesis Summary

During recent years, electric propulsion has increasingly become the solution of choice for propulsion systems for commercial telecommunication satellites. However, when the electric propulsion is used, it can take transfer time up to several months to reach the target orbit due to lower thrust as compared with the conventional chemical propulsion. The longer transfer time results in satellites spending a longer time in the Van Allen belts, where they are exposed to the harmful effects of space radiation. Therefore, this work approached the challenge of long transfer time taken by all-electric spacecraft to reach GEO.

The electric orbit-raising problem is formulated as a two-step optimization problem, with the low-level optimization problem generating the trajectory over a planning horizon. The high-level planning problem seeks to adapt the weights associated with the objective function of the low-level optimization problem so as to reduce the projected transfer time. By performing a parametric study, it is observed that fine-tuning the weights of the objective function influences the transfer time of the generated low-thrust orbit-raising trajectory and therefore, the challenge of longer transfer time can be resolved. Advances in the field of artificial intelligence and machine learning techniques present one avenue in which this problem can be addressed without any human control and interference. The main aim of this thesis is to explore novel reinforcement learning methodologies for the purpose of sequential selection of relative weights for spacecraft low-thrust orbit raising trajectory alongside state-of-the-art machine learning neural networks, also known as Deep Learning. The reinforcement technique used in this work is Q-learning. Although, due to the high dimensionality of the states of our orbit-raising problem, neural networks are incorporated with Q-learning which is known as Deep Q-learning.

In the proposed methodology, Deep Q-learning algorithm, a technique with characteristics of reinforcement learning and neural networks, allows the satellite controller to make sequential decisions in order to safely navigate through the environment. The formulation for the high-level problem is amenable to the application of reinforcement learning, and artificial neural networks are used to estimate the transfer time for the orbit-raising maneuver corresponding to different orbital states and objective function weight selection.

For each orbit-raising scenario, two neural networks are trained on the data provided by low-level optimization problem for both planar and nonplanar cases. A neural network's architecture is defined as five planning states and six planning states as inputs for planar and nonplanar cases respectively, and transfer time to reach GEO for that planning state as the output with one hidden layer. Training of these networks is done by using the Levenberg-Marquardt algorithm by altering the value of hidden layer neurons. The performance of the neural network is calculated by mean squared error. NN with hidden neurons corresponding to least MSE is selected. Then, this neural network is integrated with low-level optimization problem. For every sub-problem, the algorithm compares three different selections of w_h (three different values for w_h) for planar case and five different actions (five different values for w_h and w_{hxy}) for nonplanar case in order to choose the optimal weight selection for the high-level planning process. These selections are dependent on the constant values of change in weights (Δw_h and Δw_{hxy}) or step-size for each planning period which are user-defined. Different values for these step-sizes are tested for the proposed machine learning algorithm. The action corresponding to the minimum estimated transfer time to GEO is selected for that sub-problem. This sequence of selecting best action is continued until the terminal criteria for low-level optimization problem are satisfied.

Moreover, a single generalized neural network is introduced which is trained with data with the combination of all the mission scenarios considered in this work for both planar and nonplanar transfers. The training of the single NN is done with the same algorithm as in the cases for individual transfers. Then, this generalized NN is integrated with low-level

optimization problem. Consequently, this single generalized neural network gave much better results as compared with results while using individual networks. Numerical results in the previous chapter demonstrated that the adaptive weighting scheme can improve the transfer time for the computed low-thrust orbit-raising trajectory. It is observed that the results obtained for transfer time by the parametric study are better than by using the planning algorithm. Although, that comes at the cost of computational time to generate trajectories. The computational time required to obtain a minimum transfer time trajectory is in the order of hours whereas trajectories obtained by using the planning algorithm are generated in the order of few minutes.

5.2 Future Work

Recommendations for future work are:

1. It is recommended to consider the change in the weights (Δw_h and Δw_{hxy}) or step-size for each planning period as an input while training the neural network. This trained neural network would allow the proposed machine learning algorithm to select the value of step-size as well in an automated manner. Hence, this adaptive step-size scheme has the scope of reducing additional time to reach GEO.

2. In the proposed machine learning algorithm, all neural networks are trained with only one value of thrust and specific impulse implying continuous thrusting with constant values of thrust magnitude and specific impulse throughout the orbit-raising problem. This means a methodology could be developed in which a single neural network is trained for all the possible values of thrust and specific impulse. This would allow the spacecraft to adaptively select the thrust which could lead to reduced fuel costs.

3. The performance of the generalized NN and in fact, the performance of the individual NNs, can be improved by changing the architecture of the considered neural network. For instance, adding one or more hidden layers to the existing neural network could

lead to higher performance. However, that would come at the cost of more training time required for the neural network.

REFERENCES

REFERENCES

[1] Petro, E. M. and Sedwick, R. J., "Survey of moderate-power electric propulsion systems," *Journal of Spacecraft and Rockets*, Vol. 54, No. 3, 2017, pp. 529–541.

[2] Hebden, K., "Europe's first all-electric telecom satellite breaks record," [accessed: 19 December, 2019]. URL: `https://room.eu.com/news/europes-first-all-electric-telecom-satellite-breaks-record/`.

[3] Free, B. and Babuska, V., "Repositioning of geostationary spacecraft-Chemical and electric propulsion options," 16th International Communications Satellite Systems Conference, 1996, pp. 1302–1311.

[4] SpaceX, "Falcon 9 Launch Vehicle Payload User's Guide," [accessed: 20 December, 2019]. URL: `https://www.spaceflightnow.com/falcon9/001/f9guide.pdf`.

[5] Byers, D. C. and Dankanich, J. W., "Geosynchronous-earth-orbit communication satellite deliveries with integrated electric propulsion," *Journal of Propulsion and Power*, Vol. 24, No. 6, 2008, pp. 1369–1375.

[6] Dutta, A., Libraro, P., Kasdin, N., and Choueiri, E., "Satellite power subsystem requirements for time-constrained electric orbit-raising with minimal radiation impact," *Advances in the Astronautical Sciences*, Vol. 148, 1 2013, pp. 765–782.

[7] Dutta, A., Libraro, P., Kasdin, N. J., and Choueiri, E., "A direct optimization based tool to determine orbit-raising trajectories to GEO for all-electric telecommunication satellites," AIAA/AAS Astrodynamics Specialist Conference, 2012, p. 4589.

[8] Fitzgerald, A. M., *The effect of solar array degradation in orbit-raising with electric propulsion*, Ph.D. thesis, Massachusetts Institute of Technology, 1992.

[9] Sreesawet, S. and Dutta, A., "Fast and Robust Computation of Low-Thrust Orbit-Raising Trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 9, 2018, pp. 1888–1905.

[10] Betts, J. T., "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, Vol. 21, No. 2, 1998, pp. 193–207.

[11] Sackett, L. L., Malchow, H. L., and Delbaum, T., "Solar electric geocentric transfer with attitude constraints: analysis," 1975.

[12] Kluever, C. A. and Oleson, S. R., "Direct approach for computing near-optimal low-thrust earth-orbit transfers," *Journal of Spacecraft and Rockets*, Vol. 35, No. 4, 1998, pp. 509–515.

[13] Marasch, M. W. and Hall, C. D., "Application of Energy Storage to Solar Electric Propulsion Orbital Transfer," *Journal of Spacecraft and Rockets*, Vol. 37, No. 5, 2000, pp. 645–652.

[14] Graham, K. F. and Rao, A. V., "Minimum-time trajectory optimization of multiple revolution low-thrust earth-orbit transfers," *Journal of Spacecraft and Rockets*, Vol. 52, No. 3, 2015, pp. 711–727.

[15] Wiesel, W. E. and Alfano, S., "Optimal many-revolution orbit transfer," *Journal of Guidance, Control, and Dynamics*, Vol. 8, No. 1, 1985, pp. 155–157.

[16] Rutherfobd, D. E., "Optimal Trajectories For Space Navigation. By D. F. Lawden. Pp. viii, 126. 21s. net. 1963. (Butterworth and Co.)," *The Mathematical Gazette*, Vol. 48, No. 366, 1964, pp. 478–479. doi:10.2307/3611765.

[17] Russell, R. P., "Primer vector theory applied to global low-thrust trade studies," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 2, 2007, pp. 460–472.

[18] Thorne, J. D. and Hall, C. D., "Approximate initial Lagrange costates for continuous-thrust spacecraft," *Journal of guidance, control, and dynamics*, Vol. 19, No. 2, 1996, pp. 283–288.

[19] Betts, J. T., "Very low-thrust trajectory optimization using a direct SQP method," *Journal of Computational and Applied Mathematics*, Vol. 120, No. 1-2, 2000, pp. 27–40.

[20] Herman, A. L. and Conway, B. A., "Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 3, 1996, pp. 592–599.

[21] Dutta, A. and Arora, L., "Objective function weight selection for sequential low-thrust orbit-raising optimization problem," *Advances in the Astronautical Sciences*, Vol. 168, No. AAS 19-567, 2019, pp. 1023–1038.

[22] Falck, R. and Dankanich, J., "Optimization of low-thrust spiral trajectories by collocation," *AIAA/AAS Astrodynamics Specialist Conference, Guidance, Navigation, and Control and Co-located Conferences*, Minneapolis, MN, August 2012.

[23] Graham, K. F. and Rao, A. V., "Minimum-time trajectory optimization of low-thrust earth-orbit transfers with eclipsing," *Journal of Spacecraft and Rockets*, Vol. 53, No. 2, 2016, pp. 289–303.

[24] Petropoulos, A. E. and Longuski, J. M., "Shape-based algorithm for the automated design of low-thrust, gravity assist trajectories," *Journal of Spacecraft and Rockets*, Vol. 41, No. 5, 2004, pp. 787–796.

[25] Wall, B. J. and Conway, B. A., "Shape-based approach to low-thrust rendezvous trajectory design," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 1, 2009, pp. 95–101.

[26] De Pascale, P. and Vasile, M., "Preliminary design of low-thrust multiple gravity-assist trajectories," *Journal of Spacecraft and Rockets*, Vol. 43, No. 5, 2006, pp. 1065–1076.

[27] Vasile, M., De Pascale, P., and Casotto, S., "On the optimality of a shape-based approach based on pseudo-equinoctial elements," *Acta Astronautica*, Vol. 61, No. 1-6, 2007, pp. 286–297.

[28] Novak, D. M. and Vasile, M., "Improved shaping approach to the preliminary design of low-thrust trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 1, 2011, pp. 128–147.

[29] Taheri, E. and Abdelkhalik, O., "Shape based approximation of constrained low-thrust space trajectories using Fourier series," *Journal of Spacecraft and Rockets*, Vol. 49, No. 3, 2012, pp. 535–546.

[30] Kluever, C. A., "Simple guidance scheme for low-thrust orbit transfers," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 6, 1998, pp. 1015–1017.

[31] Petropoulos, A. E., "Simple control laws for low-thrust orbit transfers," *AAS/AIAA Astrodynamics Specialists Conference*, AIAA Paper 2003-0630, 2003.

[32] Petropoulos, A., "Low-thrust orbit transfers using candidate Lyapunov functions with a mechanism for coasting," *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, AIAA 2004-5089, 2004.

[33] Junkins, J. L. and Taheri, E., "Exploration of Alternative State Vector Choices for Low-Thrust Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 42, No. 1, 2018, pp. 47–64.

[34] Russell, S. J. and Norvig, P., "Artificial intelligence: a modern approach," 2016, pp. 2–5.

[35] Caruana, R. and Niculescu-Mizil, A., "An empirical comparison of supervised learning algorithms," *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 161–168.

[36] Alashwal, H., El Halaby, M., Crouse, J. J., Abdalla, A., and Moustafa, A. A., "The Application of Unsupervised Clustering Methods to Alzheimer's Disease," *Frontiers in Computational Neuroscience*, Vol. 13, 2019, pp. 31. URL: `https://www.frontiersin.org/article/10.3389/fncom.2019.00031`, `doi:10.3389/fncom.2019.00031`.

[37] Wong, K.-C., Li, Y., and Zhang, Z., "Unsupervised learning in genome informatics," Unsupervised learning algorithms, Springer, 2016, pp. 405–448.

[38] Hilas, C. S. and Mastorocostas, P. A., "An application of supervised and unsupervised learning approaches to telecommunications fraud detection," *Knowledge-Based Systems*, Vol. 21, No. 7, 2008, pp. 721–726.

[39] Sutton, R. S. and Barto, A. G., "Reinforcement learning: An introduction," 2018, pp. 1–17.

[40] Mahmud, M., Kaiser, M. S., Hussain, A., and Vassanelli, S., "Applications of deep learning and reinforcement learning to biological data," *IEEE transactions on neural networks and learning systems*, Vol. 29, No. 6, 2018, pp. 2063–2079.

[41] Jiang, Z., Xu, D., and Liang, J., "A deep reinforcement learning framework for the financial portfolio management problem," *arXiv preprint arXiv:1706.10059*, 2017 [accessed: 21 December, 2019].

[42] Jiang, Z. and Liang, J., "Cryptocurrency portfolio management with deep reinforcement learning," 2017 Intelligent Systems Conference (IntelliSys), IEEE, 2017, pp. 905–913.

[43] Polydoros, A. S. and Nalpantidis, L., "Survey of model-based reinforcement learning: Applications on robotics," *Journal of Intelligent & Robotic Systems*, Vol. 86, No. 2, 2017, pp. 153–173.

[44] Ziane, S. and Melouk, A., "A swarm intelligent multi-path routing for multimedia traffic over mobile ad hoc networks," Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks, 2005, pp. 55–62.

[45] Lewis, F. L. and Vrabie, D., "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE circuits and systems magazine*, Vol. 9, No. 3, 2009, pp. 32–50.

[46] Powell, W. B., *Approximate Dynamic Programming: Solving the curses of dimensionality*, Vol. 703, John Wiley & Sons, 2007, pp. 25-38.

[47] Wang, F.-Y., Zhang, H., and Liu, D., "Adaptive dynamic programming: An introduction," *IEEE Computational Intelligence Magazine*, Vol. 4, No. 2, 2009, pp. 39–47.

[48] Xu, X. et al., "Special Section on Reinforcement Learning and Approximate Dynamic Programming." *Journal of Intelligent Learning Systems and Applications*, Vol. 2, No. 2, 2010, pp. 55–56.

[49] Izzo, D., Märtens, M., and Pan, B., "A Survey on Artificial Intelligence Trends in Spacecraft Guidance Dynamics and Control," *Astrodynamics*, 2018, pp. 1–13.

[50] Zhao, S. and Zhang, J., "Minimum-fuel station-change for geostationary satellites using low-thrust considering perturbations," *Acta Astronautica*, Vol. 127, No. C, 2016, pp. 296–307. doi:10.1016/j.actaastro.2016.05.028.

[51] Miller, D. and Linares, R., "Low-Thrust Optimal Control via Reinforcement Learning," *29th AAS/AIAA Space Flight Mechanics Meeting*, 2019, pp. 1–18.

[52] Chu X, Alfriend KT, Z. J. Z. Y., "Q-learning Algorithm for Path planning to maneuver through a satellite cluster," *AAS/AIAA Astrodynamics Specialist Conference*, 08 AAS 18–268, August, 2018.

[53] Izzo, D., Sprague, C. I., and Tailor, D. V., "Machine learning and evolutionary techniques in interplanetary trajectory design," Modeling and Optimization in Space Engineering, Springer, 2019, pp. 191–210.

[54] Li, H., Topputo, F., and Baoyin, H., "Autonomous Time-Optimal Many-Revolution Orbit Raising for Electric Propulsion GEO Satellites via Neural Networks," *arXiv preprint arXiv:1909.08768*, 2019 [accessed: 27 December, 2019].

[55] Arora, L. and Dutta, A., "Reinforcement Learning for Sequential Low-Thrust Orbit Raising Problem," *AIAA Scitech 2020 Forum, 30th AIAA/AAS Space Flight Mechanics Meeting*, 2020, p. (2186). URL: https://doi.org/10.2514/6.2020-2186.

[56] Alfano, S. and Thorne, J. D., "Circle-to-circle constant-thrust orbit raising," *Journal of the Astronautical Sciences*, Vol. 42, No. 1, 1994, pp. 35–45.

[57] Yang, G., "Direct optimization of low-thrust many-revolution earth-orbit transfers," *Chinese Journal of Aeronautics*, Vol. 22, No. 4, 2009, pp. 426–433.

[58] Betts, J. T., "Optimal low–thrust orbit transfers with eclipsing," *Optimal Control Applications and Methods*, Vol. 36, No. 2, 2015, pp. 218–240.

[59] Committee, I.-A. S. D. C. et al., "IADC space debris mitigation guidelines," URL: `http://www.iadc-online.org/Documents/Docu/IADC_Mitigation_Guidelines_Rev1_Sep07.pdf` [cited : 27 December, 2019].

[60] Marquardt, D. W., "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the society for Industrial and Applied Mathematics*, Vol. 11, No. 2, 1963, pp. 431–441.

[61] Ramsin, H. and Wedin, P.-Å., "A comparison of some algorithms for the nonlinear least squares problem," *BIT Numerical Mathematics*, Vol. 17, No. 1, 1977, pp. 72–90.