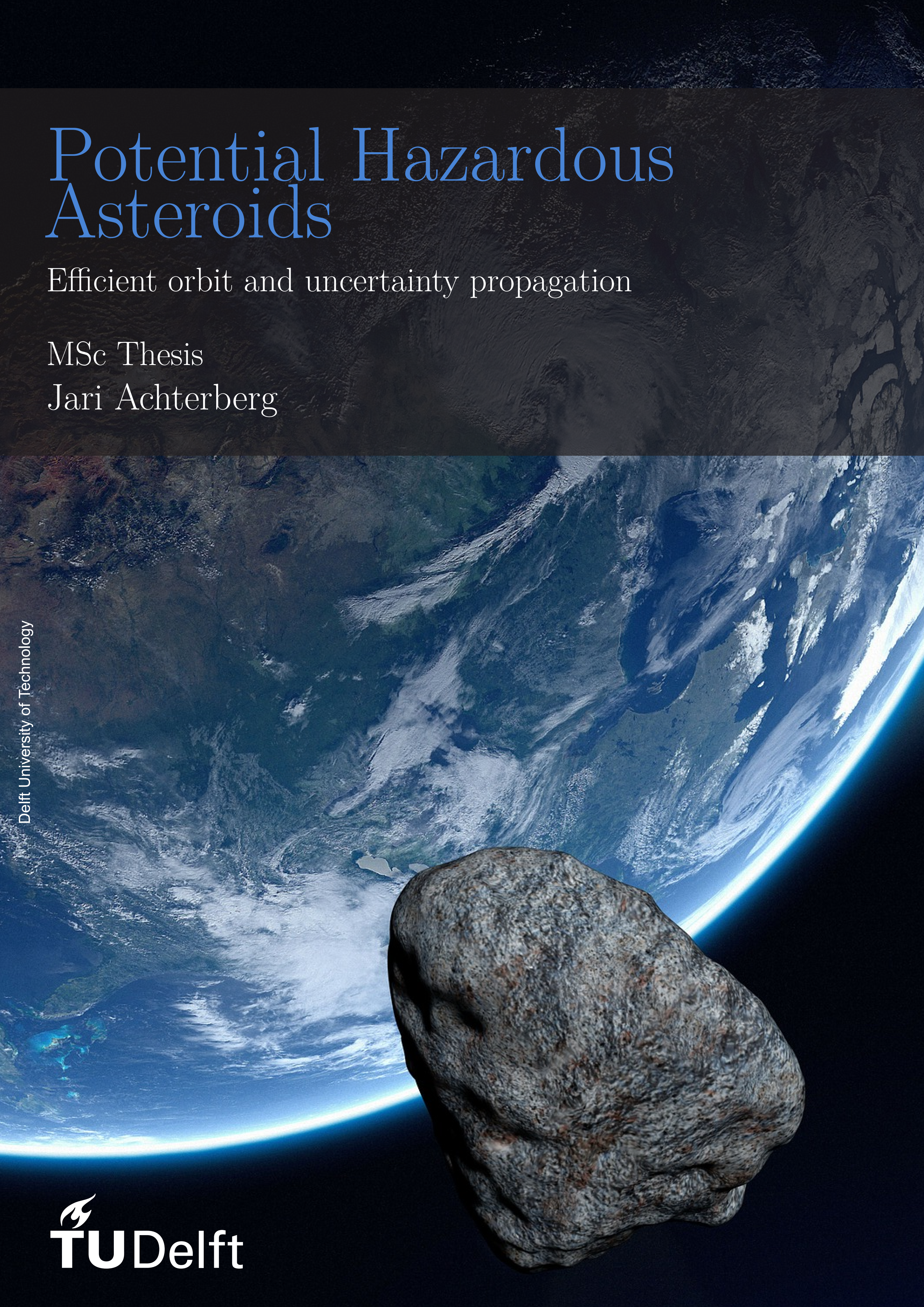
Potential Hazardous Asteroids

Efficient orbit and uncertainty propagation

MSc Thesis

Jari Achterberg



Delft University of Technology

TUDelft

Potential Hazardous Asteroids

Efficient orbit and uncertainty propagation

MSc Thesis

by

Jari Achterberg

In partial fulfilment for the requirements for the degree of

Master of Science

in Aerospace Engineering

at Delft University of Technology,
to be defended publicly at Wednesday, November 23, 2022 at 2:00 PM.

Student number:	4685652	
Project duration:	28th of January 2022 - 30th of September 2022	
Supervisor:	Ir. R. Noomen	Astrodynamics and Space Missions
Thesis committee:	Dr. Ir. E. Mooij	Astrodynamics and Space Missions
	Dr. A. Menicucci	Space Systems Engineering

Cover image obtained from Pixabay [40]

TUDelft

Preface

I have spent the last 2 years studying the Master Aerospace Engineering and the last year I worked on this thesis within the track 'Space Flight'. I have researched the topic orbit propagation in more detail, which is in my opinion a very fascinating subject of physics. I enjoyed working on this project and have broaden my scientific view on subjects like these with a certain significance.

During this research and my Master I was greatly affected by the pandemic, as I was spending many days studying at home. It has been difficult at times for me, and this also holds for other students, to be able to cope with that situation. Luckily, the past 6 months most of the restrictions set by the government were removed. There is no guarantee that the upcoming years are free from restrictions, but the prospects are promising. In the end I was able to focus quite well on my study throughout the Master, as I needed almost no extra time compared to the nominal study period.

I would like to thank my parents for their encouragements and support throughout my studies. I would like to thank the professors for the education and for sharing their thoughts about space-related topics. I would like to thank Ir. K.J. Cowan in particular, for helping me by sharing his expertise on machine learning and neural networks. Last, but not least, I would like to thank my supervisor Ir. R. Noomen for guiding me through this study.

Jari Achterberg
Nieuw-Beijerland, November 2022

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
ABM	Adam-Bashforth-Moulton
ADAM	ADAptive Moment estimation
AU	Astronomical Unit
BS	Bulirsch-Stoer
CNEOS	Center for Near-Earth Object Studies
DEC	Declination
DOPRI	Dormand–Prince
ECLIP	Ecliptic
EM	Exponential Map
ESA	European Space Agency
IAU	International Astronomical Union
JD	Julian Date
JPL	Jet Propulsion Laboratory
LBFGS	Limited-memory Broyden-Fletcher-Goldfarb-Shanno
MAE	Mean Absolute Error
MC	Monte Carlo
MEE	Modified Equinoctical Elements
MMS	MinMax Scaler
MPC	Minor Planet Center
MRP	Modified Rodrigues Parameters
MSE	Mean Squared Error
NASA	National Aeronautics and Space Administration
NN	Neural Network
PHA	Potential Hazardous Asteroid
PM	Point-Mass gravity model
QT	Quantile Transformer
RA	Right Ascension
ReLU	Rectified Linear Unit
RKDP	Runge-Kutta Dormand-Prince
RKF	Runge-Kutta Fehlberg
SBDL	Small-Body Database Lookup
SBDQ	Small-Body Database Query
SDM	Standard Dynamical Model
SGD	Stochastic Gradient Descent
SS	Standard Scaler
SSA	Space Situational Awareness
SSB	Solar System Barycenter
TUDAT	TU Delft Astrodynamics Toolbox
USM	Unified State Model
V&V	Verification and Validation

Symbols

Symbol	Definition	Unit
A	Effective surface area	[m^2]
a	Semi-major axis	[m]
b	Biases for machine learning	[-]
c	Speed of light	[m/s]
e	Eccentricity	[-]
d_E	Distance between asteroid and Earth	[m]
d_r	Radial distance	[m]
f	Frequency	[Hz]
f_0	Base frequency	[Hz]
i	Inclination	[rad]
L	Function of weights and biases in SGD	[-]
M	Mean anomaly	[rad]
m	Mass	[kg]
m	Number of hidden layers	[-]
n	Number of data points	[-]
n_i	Number of inputs	[-]
n_o	Number of outputs	[-]
n_L	Number of neurons per layer	[-]
n_w	Number of weights	[-]
p	Number of inputs	[-]
q	Number of outputs	[-]
R	Asteroid radius	[m]
R‘	Updated asteroid radius	[m]
R_E	Radius of Earth	[m]
r	Position vector, 3D	[m]
r	Distance	[m]
T	Orbital period	[s]
TOF	Time Of Flight	[s]
t	Time	[s]
t_d	Time delay	[s]
V	Velocity vector, 3D	[m/s]
V_r	Radial velocity	[m/s]
w	Weights for machine learning	[-]
x	Input variable for machine learning	[-]
x	x-coordinate position	[m]
y	True values	[-]
$\hat{y}$	Predicted values	[-]
$\bar{y}$	Mean of true values	[-]
y	y-coordinate position	[m]
z	z-coordinate position	[m]
α	Strength of regularization parameter	[-]
α	Learning rate	[-]
β	Exponential decay rate for momentum in ADAM	[-]
Δ	Difference between	[-]
Δt	Step size	[s]
ϵ	Position error due to uncertainties	[m]
ϵ	Obliquity of the ecliptic	[rad]
ϵ	Numerical stability in ADAM	[-]
θ	True anomaly	[rad]
θ	Weights and biases vector in SGD	[-]
μ	Mean	[-]

Symbol	Definition	Unit
μ	Gravitational parameter	[m^3/s^2]
ρ	Asteroid density	[kg/m^3]
σ	Activation function	[-]
σ	Standard deviation	[-]
ω	Argument of pericenter	[rad]
Ω	Right ascension of the ascending node	[rad]

Abstract

One of the main concerns of Space Situational Awareness currently is about Earth impacts. Potential Hazardous Asteroids (PHAs) are a potential hazard to Earth and therefore it is required to predict possible impacts in advance. This is done by means of orbit and uncertainty propagation. The orbit of a PHA is typically propagated numerically and also the uncertainties are propagated to estimate the magnitude of such uncertainties. Orbits can be propagated using an analytical, numerical or hybrid propagation technique. A numerical model is accurate, but not very fast, while an analytical model is usually fast with limited accuracy. A hybrid model combines an analytical or numerical model with an error modelling technique, focused on predicting the error made by the numerical or analytical model. Error modelling techniques can also be used as a stand-alone model, to save even more time. One of these error modelling techniques is machine learning. Machine learning will be used for the propagation of the state and the uncertainties.

A machine-learning technique is investigated since it is very fast as the predictions can be made instantly, making it a more efficient way in terms of computational speed and accuracy of orbit and uncertainty propagation. Current techniques require more time, in particular uncertainty propagation, which requires at least 50 times more time, as the state needs to be propagated 50 times to get a reasonable average of the error induced by the uncertainty in the initial state and the dynamics model. Machine learning is implemented as a stand-alone model to be able to aim at a solution that has the lowest computational speed. Data are generated for known PHAs by propagating the state and uncertainties of all PHAs via an integration model. 50 data points evenly distributed in time per PHA (a total of 2241) are stored, which comes in total down to 112,050 data points. These data points are then used to train a neural network, which is aimed at capturing the behaviour of the PHA trajectories by using this data. Two different outputs are predicted; the minimum distance to Earth and the error as a function of time caused by the uncertainties in the initial state and the dynamic model. These outputs contain all the relevant information to draw conclusions on possible Earth impacts. The predictions for these outputs were investigated for three different combinations of input and output variables. The first case uses Kepler elements as the initial state and uncertainties of the asteroid and predicts the position of the asteroid with respect to the Sun and the the error caused by the uncertainties in the initial state and the dynamic model. The second case uses Cartesian state elements instead of Kepler elements. The first two cases require a post-processing step, which computes the distance to Earth from the asteroid position with respect to the Sun using ephemeris data of Earth. The third case provides the position of Earth as an extra input and the neural network predicts the distance to Earth directly instead of predicting the position with respect to the Sun. This means that the third case does not require any post-processing steps, but is more complex in terms of capturing the behaviour of PHAs. The most interesting hyperparameters of the neural networks of these three cases are tuned. The R^2 score is used as the metric to compare different cases and hyperparameters.

The first case is found to be the best case with a certain set of hyperparameters and with an R^2 score of 0.96. This is fine-tuned by looking at a selection of different aspects of the model. The performance turns out to be worse for the case where less data points per asteroid are used. This is expected, as less information on the behaviour of the asteroids is used. The performance improves slightly for the case where the outputs are predicted separately, but only marginally. Another test is the use of less data, and focuses on the more interesting data points (points closer to Earth). This also results in a worse performance of these interesting points, as the effect of using less data points is larger than the effect of predicting the most interesting points in a more focused way. One last test is performed on the error caused by the uncertainties in the initial state and the dynamic model specifically, and uses a different scaling method; a log scale on the output. The performance of the model using this different scaling method is significantly better compared to the previous best result. This final result is tuned again for all the hyperparameters to see if a better result can be obtained with this new scaling method.

The final best neural network performance is found by predicting the error caused by the uncertainties in the initial state and the dynamic model separately and by using a log scale on the output. The model contains 112,050 data points, of which 20% is test data. This result is found by using a neural network with an adaptive learning rate of 0.001, a batch size of 150, 3 hidden layers and 128 neurons per layer, ADAM as the solver for weight optimization and ReLu as the activation function. This neural network model results in an average error of 27.7 % compared to the actual error caused by the uncertainties in the initial state and the dynamic model. The standard deviation for this result is 51.7 %. The standard deviation is sensitive to outliers due to the large variety in percentages. The standard deviation decreases to 41.7% when 3 out of the total 448 PHAs that were tested are removed. A normal distribution was also fitted based on the right hand side of the data (percentages larger than the mean). This method ignores outliers due to the fitting procedure and this resulted in a standard deviation of 33.7%. The performance of the model is thus better for a large fraction of the more interesting points of the data compared to all data. These values show that the error caused by the uncertainties in the initial state and the dynamic model can be estimated better than the same order of magnitude, which shows that the model performs well in terms of accuracy. The model can be applied to newly detected PHAs. The passing distance to Earth for that same time instant can be computed using an integration model, and combining these methods results in a computation time that is at least 50 times faster than that of current propagation techniques. The computation time is thus much better in terms of efficiency and the accuracy is similar compared to current techniques, showing that this model is more efficient.

1 Introduction

1.1. Background

The amount of space debris is increasing over time [13]. Space debris is mainly located in orbits around Earth, making it a challenge to prevent collisions between satellites and space debris [20]. From the perspective of collisions, a similar type of objects are natural objects, such as asteroids, comets and meteoroids. In this particular case the impact with Earth is of most interest, as impact events can be a potential hazard to human population [41] [36]. The influence of space debris and Earth impacts are both concerns of Space Situational Awareness (SSA) [14].

In order to predict a possible collision or impact it is necessary to predict the position of a natural object or space debris at a later moment in time, known as orbit propagation. It also comes with an uncertainty, as the initial position and velocity of an object and the perturbation forces in the dynamics model contain uncertainties (e.g. the uncertainty in the ephemeris of Earth, effecting the third-body perturbation exerted by Earth) and this should also be propagated. These predictions need to be accurate enough, to be able to draw reasonable conclusions. Due to the increasing number of objects and the fact that the uncertainties also require propagation, this also requires a fast computation scheme. Thus it is desired to propagate orbits and their uncertainties with a high accuracy and a low computational speed.

Several propagation techniques exist, and the two main categories are numerical and analytical propagation. Numerical propagation techniques are usually accurate, but not very fast, while analytical propagation techniques are usually fast, but not very accurate [43]. To be able to find a more efficient technique, hybrid propagation techniques have been introduced. Hybrid propagation techniques combine either numerical or analytical propagation techniques with a forecasting technique, such as statistical time-series models or machine-learning techniques [51] [42]. This forecasting technique estimates the difference between the less-accurate analytical or numerical technique with a very accurate orbit solution and the two models combined can result in a model that is both accurate and fast.

1.2. Method

This thesis project will focus on the propagation of the orbits of asteroids, in particular Potential Hazardous Asteroids (PHAs), to predict Earth impacts. The prediction of Earth impacts requires the propagation of the nominal trajectory and uncertainty propagation. Uncertainty propagation depends on many factors (e.g. the magnitude of the initial uncertainty and the propagation time), and is therefore necessary to make more realistic statements about close approaches or possible Earth impacts. This is also done by the Center for Near-Earth Object Studies (CNEOS), which uses an Earth Impact Monitoring system called Sentry [10]. Sentry uses either a Monte-Carlo analysis for uncertainty propagation or the uncertainty region is derived from the limits of the observational data. The uncertainty propagation described here is computationally expensive, and this shows the need of a fast propagation technique.

A hybrid propagation technique will be used, to be able to decrease the computational speed, making it a more efficient way of orbit propagation. Machine learning will be used, as it is a new technique for these kind of purposes and it is capable of reducing the computational speed significantly [43]. Machine learning is able to capture the relations between input and output data, which makes it a special type of a regression problem. In this thesis project the inputs are the initial position and velocity of the PHA, the propagation time and all the uncertainties, and the outputs are the position of the PHA and the error caused by the uncertainties in the initial state and the dynamic model. This information is generated by propagating PHAs that are already known. Machine learning then uses a neural network to train a model based on these data. It results in a model that can be seen as a sort of formula, where new inputs can be provided and the outputs are almost instantly computed. The model can be used as a forecasting technique, focusing completely on the error of a simplified numerical or analytical propagation technique, but can also be used as a stand-alone model, meaning that the full trajectory (orbit and uncertainty) is estimated by the neural network.

The research question that follows from the previous paragraphs is as follows:

- **To what extent is machine learning able to improve orbit and uncertainty propagation of asteroids, in terms of both accuracy and computational speed?**

Sub-questions of this research question are:

- *How does analytical propagation compare to numerical propagation?*
- *What is the most convenient way to represent the state and its uncertainties and propagate them?*
- *Is the technique of machine learning suitable for orbit and uncertainty propagation of asteroids?*
- *How well does machine learning perform compared to current techniques?*
- *To what extent can optimization techniques contribute to the machine-learning process?*
- *How does the performance of machine learning used as a forecasting technique compare to as it is used as a stand-alone model?*

1.3. Structure

The structure of the report is described below and visualized in Figure 1.1. Chapter 2 starts with an investigation of the availability of asteroid data and describes which data can be used for what purpose. After that Chapter 3 describes the astrodynamics regarding asteroid orbits, including the requirements, the initial settings of the propagation, the integrator and propagator selection and the influence of perturbations on the orbit. The data acquired from Chapter 2 is used to propagate PHAs in Chapter 3. The effect of uncertainties in the initial position and velocity of an asteroid, as well as uncertainties in the dynamics model is described in Chapter 4. Chapter 5 then describes the method of machine learning, including the selection of the input and output variables to be used in the neural network, a description of the hyperparameters of a neural network and the tuning process. The network is trained using data coming from the propagation of PHAs as described in Chapter 3 and Chapter 4. The main results of this research are coming from the trained neural networks and is addressed in Chapter 6. Chapter 7 describes the verification and validation cases of the databases, the integration model and the neural network. The conclusions and recommendations are addressed in Chapter 8.

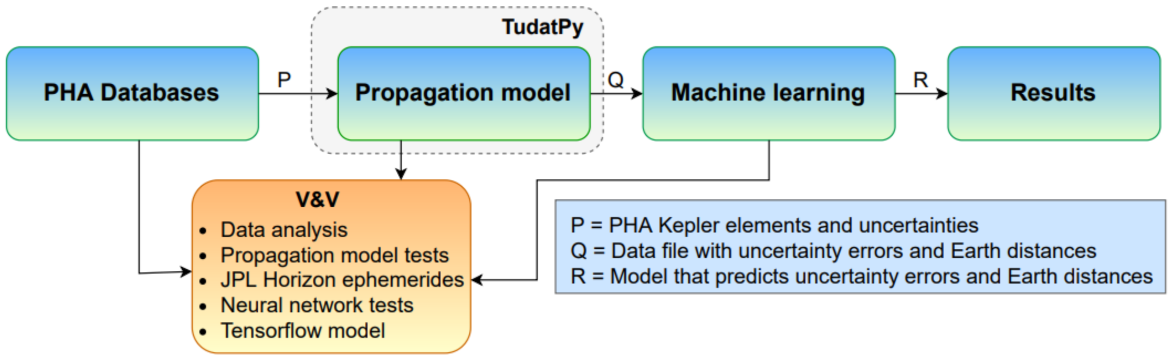

TudatPy
PHA Databases
P
Propagation model
Q
Machine learning
R
Results
V&V
• Data analysis
• Propagation model tests
• JPL Horizon ephemerides
• Neural network tests
• Tensorflow model
P = PHA Kepler elements and uncertainties
Q = Data file with uncertainty errors and Earth distances
R = Model that predicts uncertainty errors and Earth distances

Figure 1.1: Thesis project flow diagram.

2 Data availability

Data needs to be generated to be able to work with machine-learning techniques. This data will be generated by means of orbit propagation of PHAs, which will be explained in the upcoming chapters. Information of PHAs that is required for this propagation needs to be extracted from databases and the availability of this data will be discussed in this chapter. Section 2.1 describes the investigation of available databases containing information of asteroids and Section 2.2 explains in more detail how much observation data are available and how useful this data is.

2.1. Databases investigation

Two databases containing relevant asteroid data were identified and investigated, as they contain enough and relevant information regarding asteroids. These are the database from the Jet Propulsion Laboratory (JPL) and the database from the Minor Planet Center (MPC). These two will be discussed and compared in the following subsections.

2.1.1. Jet Propulsion Laboratory

JPL is a research and development lab founded by NASA. It contains several tools dedicated to 'small bodies', where small bodies are referred to as natural bodies, except for natural satellites and planets. It is possible to acquire Kepler elements and their uncorrelated uncertainties of all small bodies using the Small-Body Database Lookup (SBDL) tool (for finding one specific object) and the Small-Body Database Query (SBDQ) tool (for finding Kepler elements and uncorrelated uncertainties of PHAs) [33]. Besides these tools, radar astrometry data are available for all objects. The radar astrometry data consists of two different types: one is a Doppler frequency, which is used to determine the radial velocity of the asteroid, and the other is a time delay to determine the distance between the asteroid and Earth [34]. Observations of PHAs can be extracted from the total set of observations by looking at the designation of the object (which can be extracted from the SBDQ tool).

2.1.2. Minor Planet Center

MPC provides a database including information regarding minor planets, and is part of the International Astronomical Union (IAU) [30]. It also includes data from PHAs. Kepler elements and uncertainties can be extracted from this database, but is not different from the contents of the JPL database. This database however also includes different kinds of observation data. These are optical measurements, which are observations that provide the J2000 right ascension and declination angles. Some of them are measured from a different object (e.g. a satellite). In this case the vector from Earth to the satellite is provided as well. Radar astrometry data are also available in the MPC database, but these measurements are not complete.

2.2. Observation data

The Kepler elements and the uncertainties of PHAs in both databases can be used to propagate the nominal orbit of a PHA and its uncertainties. The observation data can be used to fit an orbit through points. It can also be used as part of a neural network. Even if the observation data are not used for the thesis project directly, it will still serve as a proper validation for the Kepler elements and uncertainties provided by the databases. This is described in more detail in Chapter 7.

The observation data will be investigated for both databases described in the previous section. The most valuable information is extracted from these databases, for example the designation of a PHA, which is a code that specifies the asteroid. Some asteroids in the MPC database have multiple designations, since it can happen that a newly observed object turns out to be equal to an already registered object a few months later. This means that some observations were assigned to a different object than other observations for the same PHA. If an object has for example two designations, only one of them is defined as the main designation, and therefore stored in the list of characterized PHAs. The other designation is not officially recognized as a PHA, making it difficult to filter all the PHA observations from the total number of observations. However there are not many objects that have multiple designations, it appears to be the case for approximately 50 out of 2200 (=2.3%) bodies and only 3000 out of 600,000 (=0.5%) observations. Therefore it is concluded that observations of objects that have multiple designations can be deleted from the data set without consequences. The observation data of JPL and MPC are given in Table 2.1.

Table 2.1: Observation data facts of PHAs.

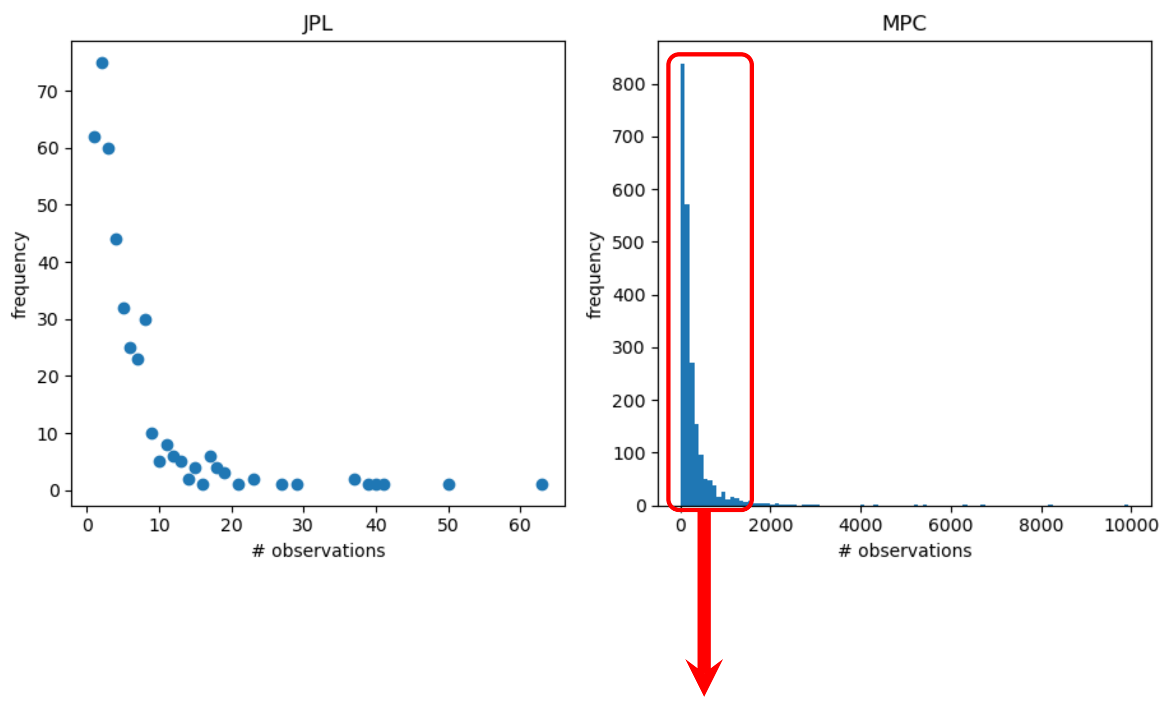
Database	JPL	MPC
Observation type	Radar astrometry	Optical and radar astrometry
# Observations	2403	611,525
# Observed objects	416	2170
Average # observations per PHA	5.78	281.8
PHA with largest number of observations	4179 Toutatis (1989 AC)	99942 Apophis (2004 MN4)
# Observations of PHA with largest number of observations	63	8168

The total number of known PHAs is 2241, as of February 2022 [33]. As can be seen, the data from the JPL database is only available for a very limited number of objects. Radar astrometry data are thus only available for a fraction of objects, and it should therefore be taken very carefully if and how radar astrometry data will be used in this thesis project.

The data obtained from the MPC database shows that observation data are available for almost all known PHAs, and that for every object on average roughly 282 observations are available. This can also be seen from Figure 2.1. This figure also roughly shows that for over 800 objects more than 200 observations are available, and that for almost 1400 objects more than 100 observations are available. This is made more clear in Figure 2.2.

It can be concluded that observation data from both databases can be used in this thesis project, although the radar astrometry data are more sparse and the MPC database is therefore preferred. If and how this observation data will be used will be explained in Chapter 5. This however only considers observation data, orbital elements and uncertainties are available in both databases.

Figure 2.1: Comparison of observation data availability between the JPL and MPC databases.



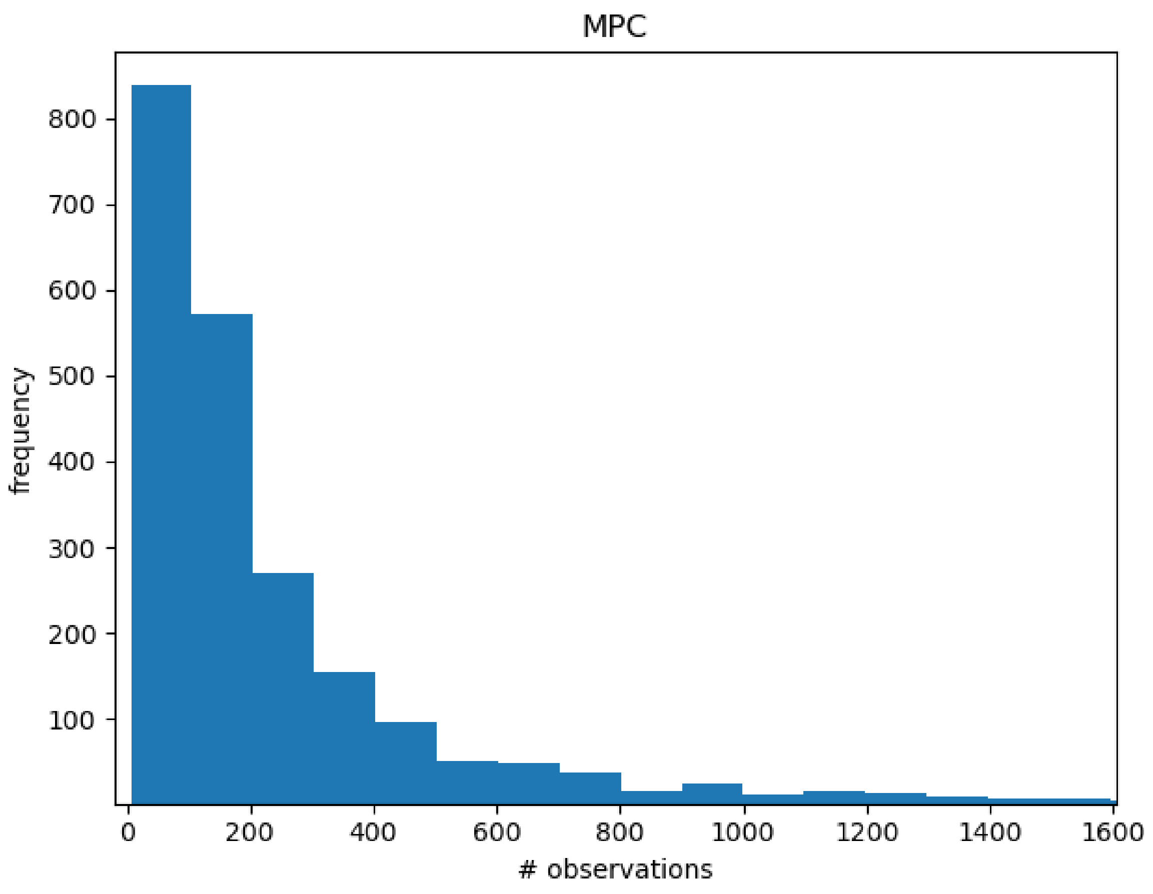


Figure 2.2: Observation data availability of the MPC database (see also Figure 2.1).

3 Astrodynamics

This chapter describes the aspects related to the astrodynamics of asteroid state and uncertainty propagation. This is necessary to be able to generate data for the machine-learning process. This chapter uses the Kepler elements and uncertainties coming from the JPL database [33] directly (see also Chapter 2). The need of observation data is discussed in Chapter 5. If this is the case, the MPC database will be used to provide information regarding optical and radar astrometry data. This is however not required for this investigation. First of all the requirements will be explained in Section 3.1. After that the numerical approach will be discussed in Section 3.2, including the definition of the initial settings and input and defining the benchmark. From that the integrator and propagator will be selected, and also the perturbations will be selected that are required for the state propagation of PHAs. The results are summarized in Section 3.3.

3.1. Requirements

One of the key elements of orbit propagation is accuracy. For that reason an accuracy requirement needs to be established. This accuracy requirement is bounded by the radius of the Earth (6378.136 km [49]). If the accuracy is taken as 10^4 km, the asteroid could either completely miss or hit Earth, with the prediction being the opposite. To make sure that the propagation outcome of hitting Earth or not is reliable, a value of **10^3 km** is taken as the accuracy requirement.

Besides the accuracy requirement there is not really a limiting factor for the propagation of the asteroid states. Computation time is preferably as small as possible, but there is no fundamental requirement. However it is necessary to define the total propagation interval. This will be defined by looking at the orbital periods of the asteroids. Taking the orbital period as the propagation interval is a good measure, since it makes it possible to investigate the error after one revolution. The smallest orbital period of a PHA is currently 183 days and the largest is 75 years. Table 3.1 also shows that the orbital period of 75 years is an outlier. It is however arbitrary to rely on such huge differences in propagation intervals, especially when accuracy is taken into consideration. To cover more potential close approaches of PHAs, it is desired to end up with a model that is able to predict states and uncertainties of newly discovered PHAs far enough in the future, which means that an orbital period of 183 days is relatively small when that same time is used for the propagation interval.

Table 3.1: PHAs with the largest orbital periods.

Asteroid	T [y]
1999 XS35	75.0
2015 HX176	7.67
2001 XQ	6.93

In order to avoid an arbitrary propagation interval, to cover more potential close approaches and to cover most of the orbital periods of the asteroids a propagation interval of 5 years is selected. 5 years is also selected to speed up the process of finding the propagation settings. The propagation interval will be increased to 10 or 20 years once the optimal settings are found, and then it only requires one extra run to see if the requirements are still met.

3.2. Numerical approach

A numerical integration model will now be created in order to check if and how many perturbations are required for the propagation of the state of PHAs. A tool is selected to set up the model and a benchmark is generated, which is required for the selection and tuning of the integrator and propagator. The selected integrator and propagator will then be used to see what effect perturbations will have on the orbit.

3.2.1. Tool selection

A numerical approach requires a tool that is capable of propagating orbits to a future instance in time. The tool chosen here is the TU Delft Astrodynamics Toolbox (Tudat). Tudat is a set of libraries, used for astrodynamics and space research, originally created in C++ [12]. TudatPy is the Python equivalent of Tudat and this variant will be used, since the author of this thesis report has the most experience with Python. Tudat is capable of extracting relevant information, such as the ephemerides of all Solar-System bodies. It is also capable of setting up environment and acceleration models, which is necessary to be able to define the settings of the propagation. On top of that it is able to provide independent variables at every available time instance (determined by the time step). This can be for example the distance to Earth, measured from the asteroid. TudatPy thus has the relevant capabilities for the purpose of this study and will therefore be used.

3.2.2. Initial settings and input

Three asteroids are selected as a representation of all asteroids; one representing the average asteroid (99942 Apophis), one asteroid with a relatively large eccentricity (1566 Icarus), and one asteroid with a relatively large orbital period (2001 XQ), see also Table 3.2. An orbit with a large eccentricity has a relatively larger velocity difference throughout the orbit compared to an orbit with a small eccentricity. Icarus is therefore expected to be the most critical PHA in terms of accuracy. An orbit with a large orbital period might not be as interesting for integration purposes, but might serve as a representative orbit for specific perturbations that need to be taken into account. It is assumed that using these three PHAs that have different Kepler elements provides enough information to make critical decisions on the choice of integrators, propagators and perturbations.

Table 3.2: Initial states of asteroids considered for propagation [33].

Asteroid	a [AU]	e [-]	i [°]	Ω [°]	ω [°]	M_0 [°]	T [y]
99942 Apophis	0.9227	0.1914	3.339	204.0	126.6	195.7	0.886
1566 Icarus	1.078	0.8269	22.80	87.96	31.44	0.3303	1.12
(2001 XQ)	3.623	0.7211	28.59	250.7	189.8	325.7	6.93

Note that the information of the start time is already included in the mean anomaly M. This is equal for all PHAs and is the 21st of January in 2022 at 0:00. The initial states are defined in the J2000 ecliptic reference frame, which will therefore also be used as the frame orientation of the propagation. The frame origin will be set as the center of the propagation, which is the Solar System Barycenter (SSB).

The asteroid mass and radius must also be inputs to the numerical integration in order to for example properly define the solar radiation pressure, which is dependent on the area-to-mass ratio. These relevant physical quantities can be found in Table 3.3. The mass of (2001 XQ) was not provided in any source, therefore an estimate was made using an estimate of the density of asteroids [7], which shows an average density (ρ) of 2.0 g/cm^3. The volume is calculated using the assumption of a perfect sphere, so the mass is calculated from the radius as follows: [28]

$$m = \frac{4}{3}\pi R^3 * \rho \tag{3.1}$$

Table 3.3: Relevant physical quantities of asteroids considered for propagation [7] [15] [31] [33] [50].

Asteroid	R [m]	m [kg]
99942 Apophis	170	$6.2 * 10^{10}$
1566 Icarus	500	$1.0 * 10^{12}$
(2001 XQ)	365	$4.1 * 10^{11}$

A numerical propagation will now be set up using a start time equal to the 21st of January of 2022 at 0:00 and a total propagation interval of 5 years. Perturbations are neglected, so only the point mass gravity of the Sun is taken into account. This holds for all the subsections below, unless specified otherwise.

3.2.3. Benchmark generation

Before the integrators and propagators can be compared, a benchmark is required to define the accuracy level of integrators and propagators. This can be an analytical solution (e.g. a Kepler orbit) or a numerically generated benchmark that is known to be exact [11]. Using a Kepler orbit as the benchmark is only valid if no perturbations are included in the propagation. Other analytical solutions that take into account some perturbations also exist, but it is very difficult to find a solution that includes the perturbations required in this specific case. The perturbations are also not set at this point. Therefore it is decided to generate the benchmark numerically. The Kepler orbit will be used as a validation test case, see Chapter 7 for more details.

It is expected that the orbit of the asteroid with the largest range of velocities results in the worst-case scenario for the propagation. This is the orbit with the largest eccentricity, which is Icarus. It is decided to only test two asteroids, namely Icarus and Apophis. Apophis will be used to show the contrast between Icarus and Apophis more clearly. The unnumbered asteroid (2001 XQ) is not used for the choice of integrators and propagators.

The accuracy of the benchmark is determined as follows; the solution found for a step size of 700,000 seconds is compared to the solution with a step size of 350,000 seconds, the maximum difference is then considered as a quantification of the error of the solution with a step size of 700,000 seconds. This is done for a number of step sizes in the range between 20,000 and 700,000 seconds and the result is shown in Figures 3.1 and 3.2.

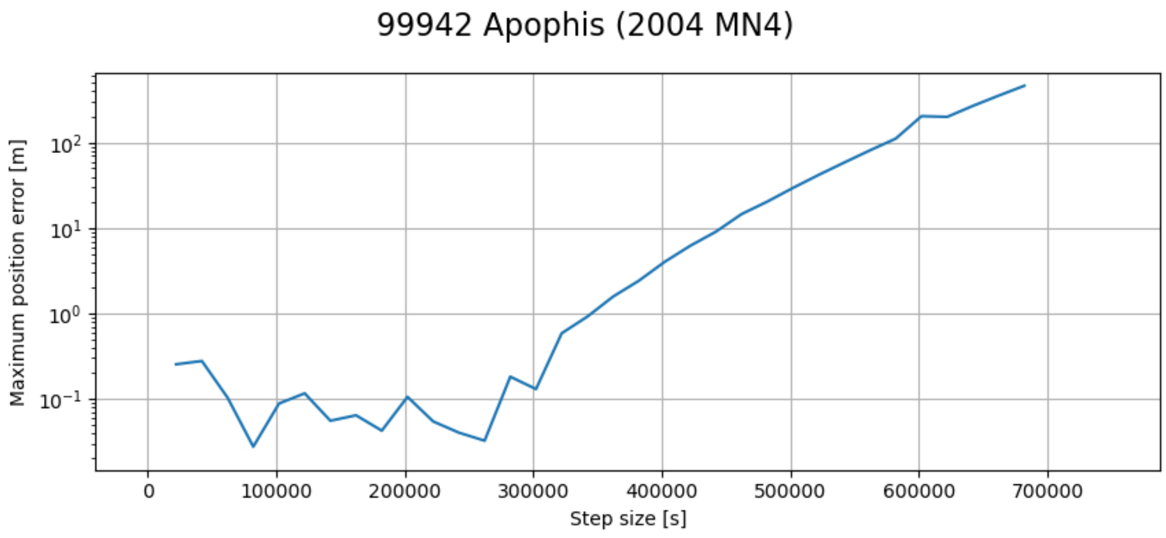


Figure 3.1: Maximum position error as a function of the fixed integration step size for Apophis. The propagation interval is 5 years.

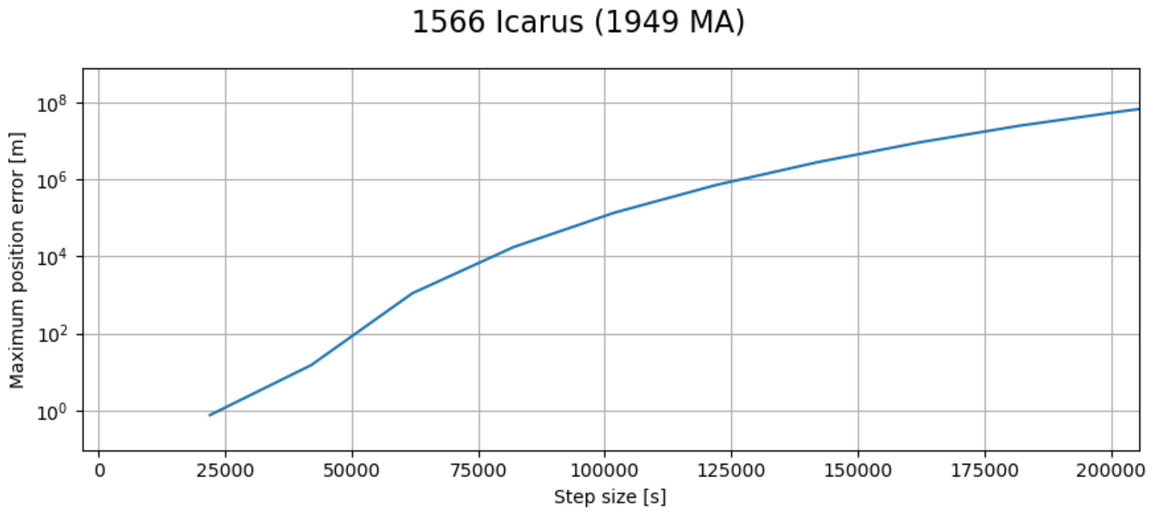


Figure 3.2: Maximum position error as a function of the fixed integration step size for Icarus. The propagation interval is 5 years.

As can be seen from the figures, the maximum position error for Icarus is much larger than for Apophis. This was expected, as Icarus includes a larger difference in velocity throughout its orbit. A step size of 50,000 seconds will be taken for Icarus to make sure that the truncation error is dominant. Due to this reasoning not the same step size is taken for Apophis. A step size of 300,000 seconds is taken for Apophis. These step sizes result in a maximum position error of meter level, which is required for the benchmark, as the benchmark should have an accuracy better than the requirement of 1000 km. This makes it possible to properly compare different integrators and propagators.

3.2.4. Integrator selection

Integrators that will be tested for the propagation of the states of the selected asteroids have already been defined in the literature study: [4]

- Runge-Kutta: RK4, RKF4(5), RKF5(6), RKF7(8) and RKDP7(8)
- Bulirsch-Stoer: BS4, BS6, BS8 and BS10
- Adam-Bashforth-Moulton: ABM, ABM6, ABM8 and ABM10

All integrators will be tested for a variable and a fixed step size, except for RK4, which has a fixed step size by definition. Other integrators are defined as variable step sizes, but can be used as fixed step size integrators by setting the initial time step, minimum time step and maximum time step to the same value. The BS integrators use the maximum number of steps (BS4 means four steps) and ABM uses a minimum order of 6 and a maximum order of 11 for the nominal case. ABM6 uses 6 as a minimum and 6 as a maximum and the same holds for the other ABM integrators.

The fixed step sizes that will be tested are varying between 7200 and 921,600 seconds and the tolerances for the variable step sizes are varying between 10^{-12} and 10^{-6}. These values have been chosen based on the benchmark results. It should be noted that the number of function evaluations reflects the computational speed of the propagation. Run time itself is not considered, since it is not a very robust measure. In all cases, the results of the previous section are the reference.

The maximum position differences are calculated between the solutions for various integrators and the benchmark for both PHAs. Figure 3.3 shows the comparison between the different integrators for Icarus for a fixed step size and Figure 3.4 shows the same information for Apophis. It can be observed that Icarus generally has a larger maximum position error than Apophis, so the PHA with the larger eccentricity indeed seems to be a worse case compared to Apophis. It can also be seen that BS6, BS8 and RKDP7(8) are performing the best on average when looking at the number of function evaluations with respect to the accuracy.

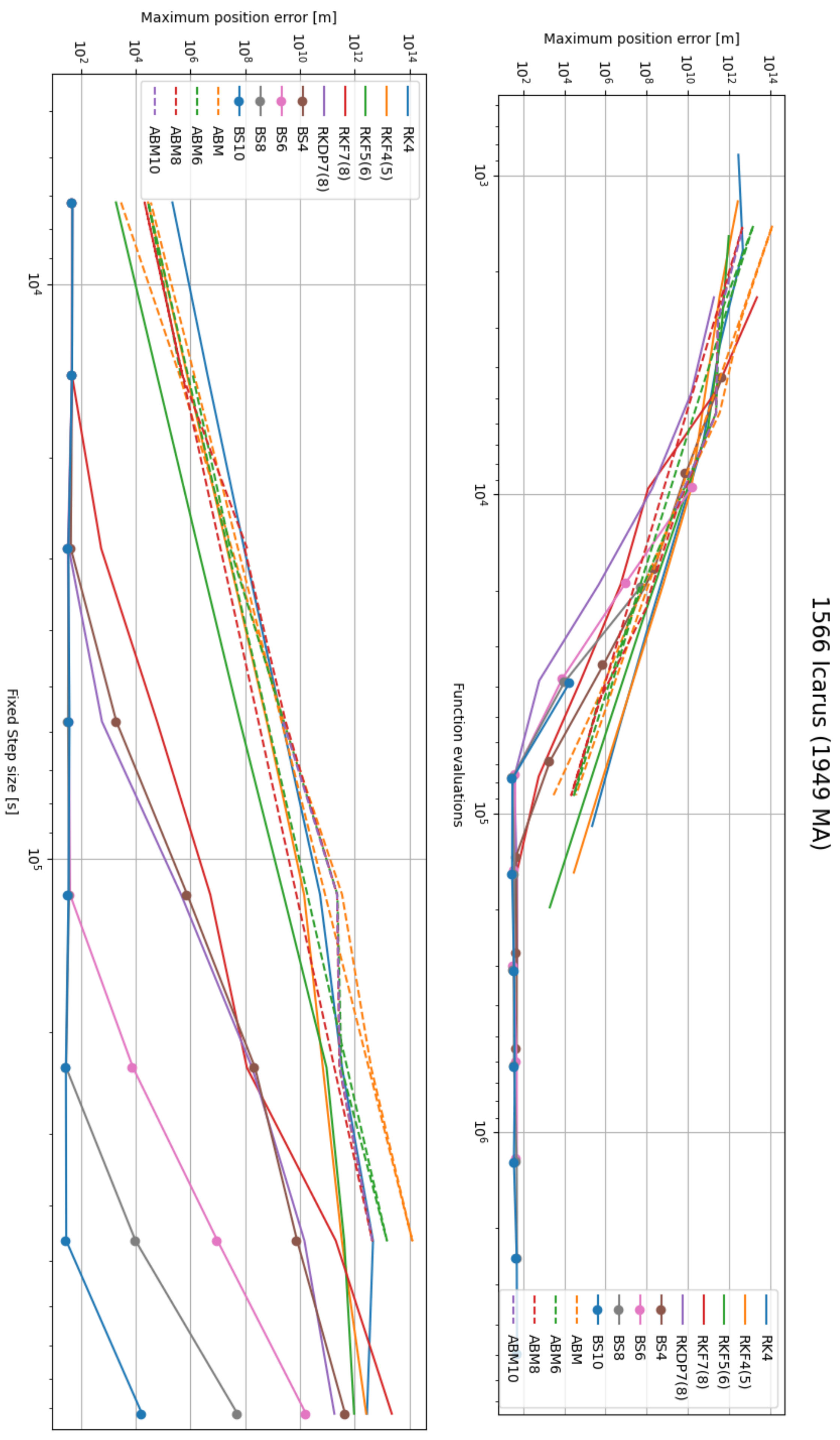


Figure 3.3: Maximum position error as a function of the fixed step size and the number of function evaluations for a number of integrators for the propagation of Icarus.

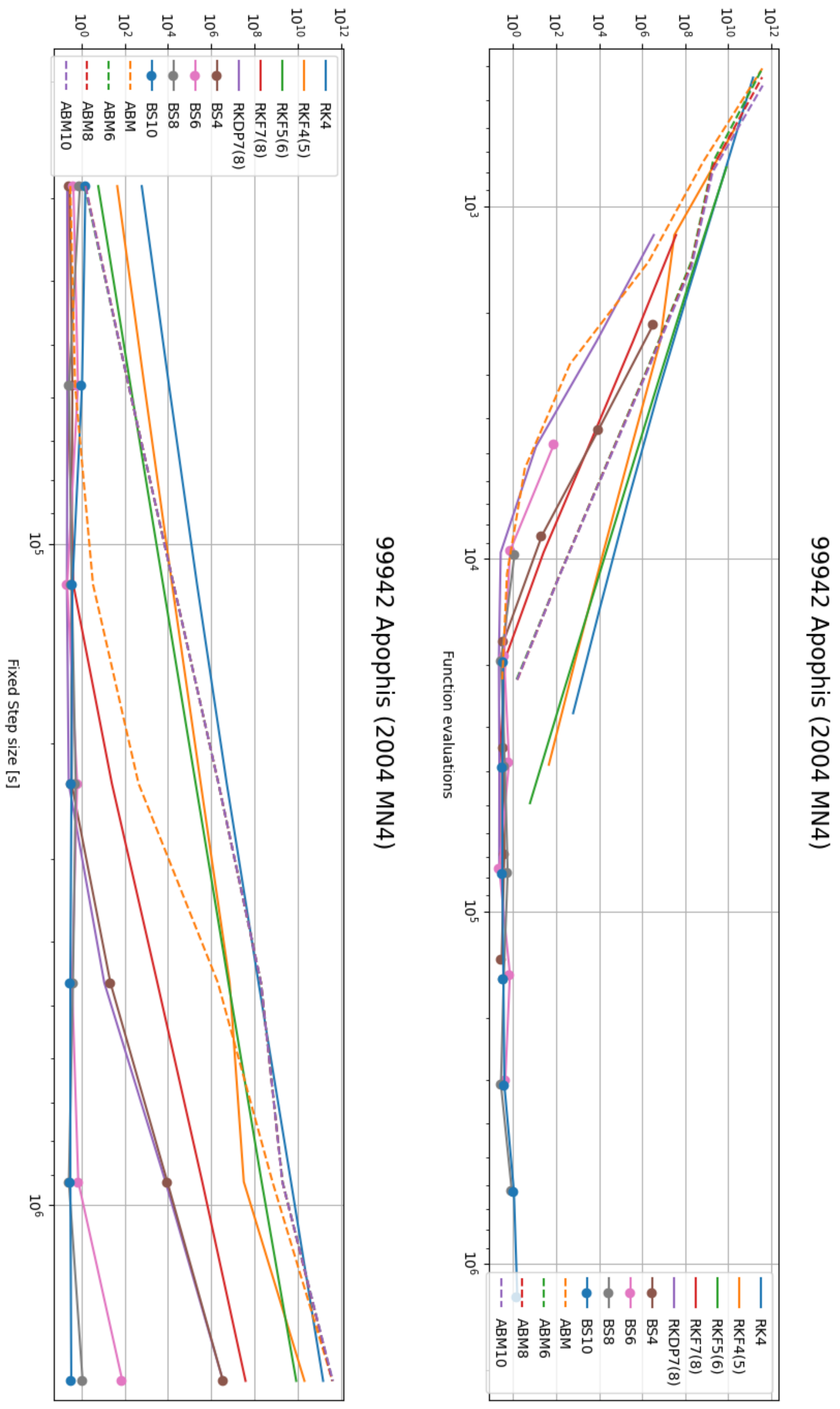


Figure 3.4: Maximum position error as a function of the fixed step size and the number of function evaluations for a number of integrators for the propagation of Apophis.

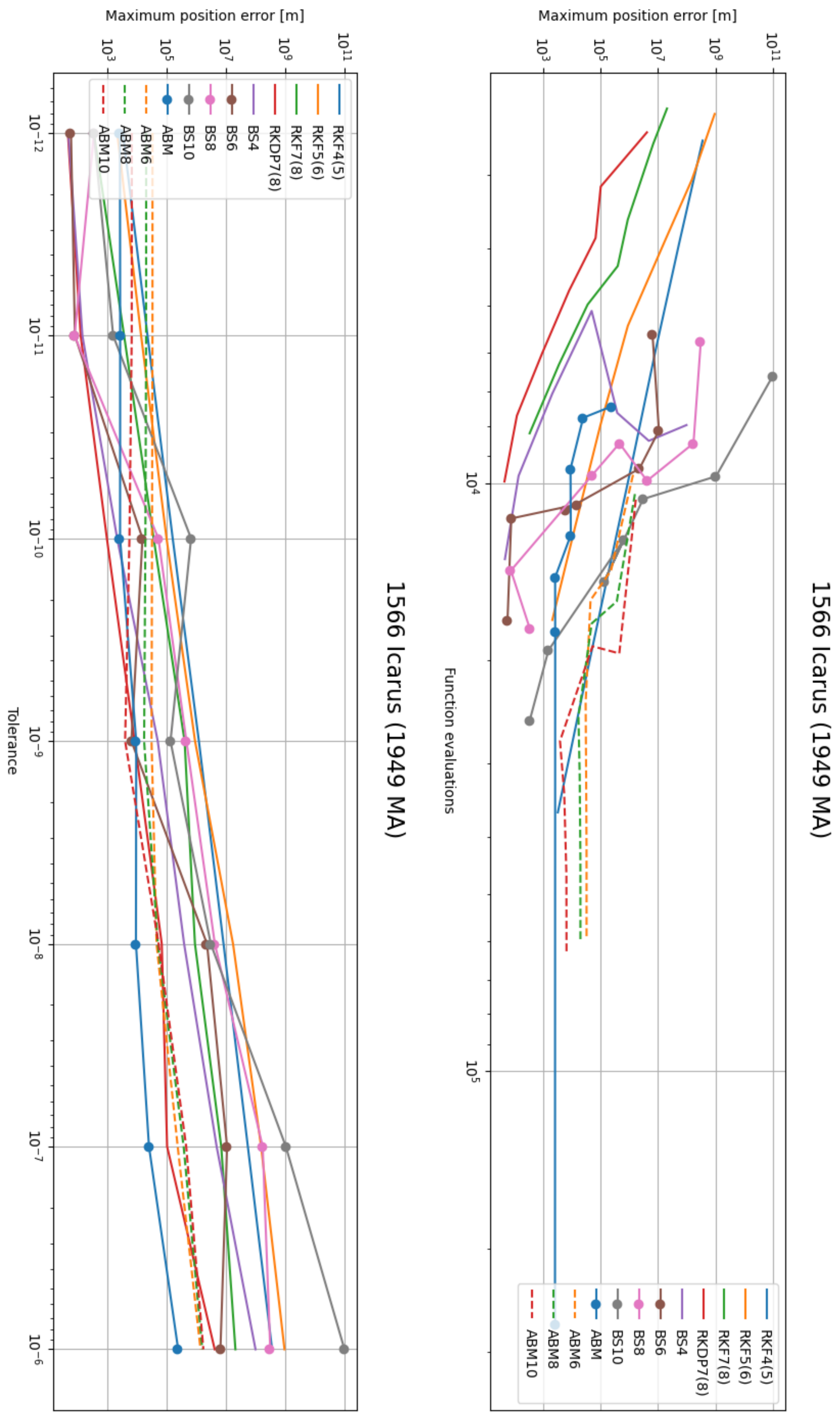
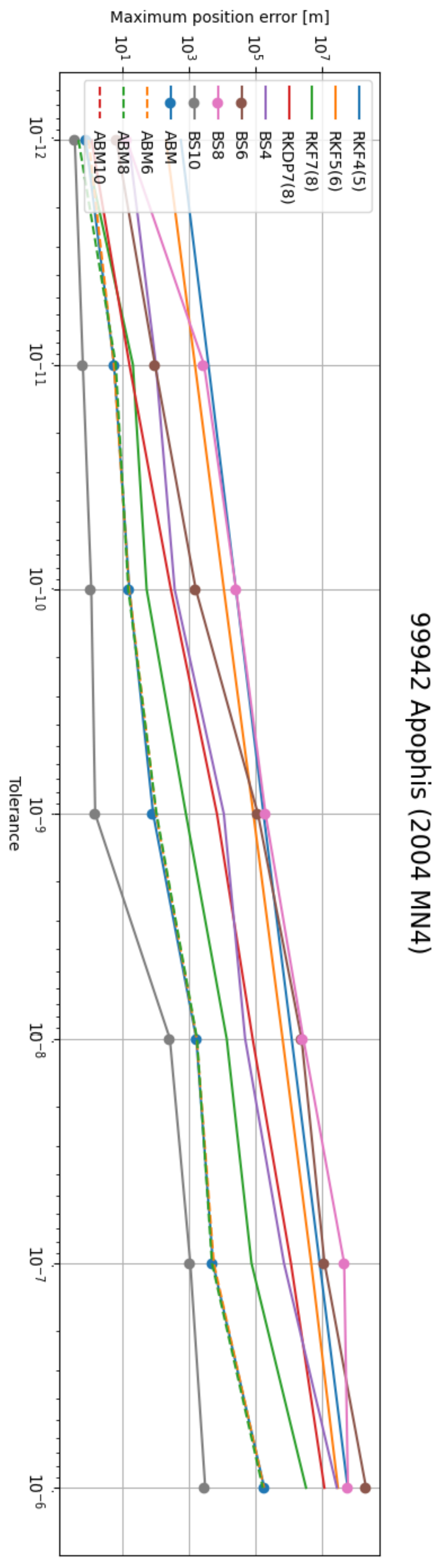


Figure 3.5: Maximum position error as a function of the tolerance and the number of function evaluations for a number of integrators for the propagation of Icarus.



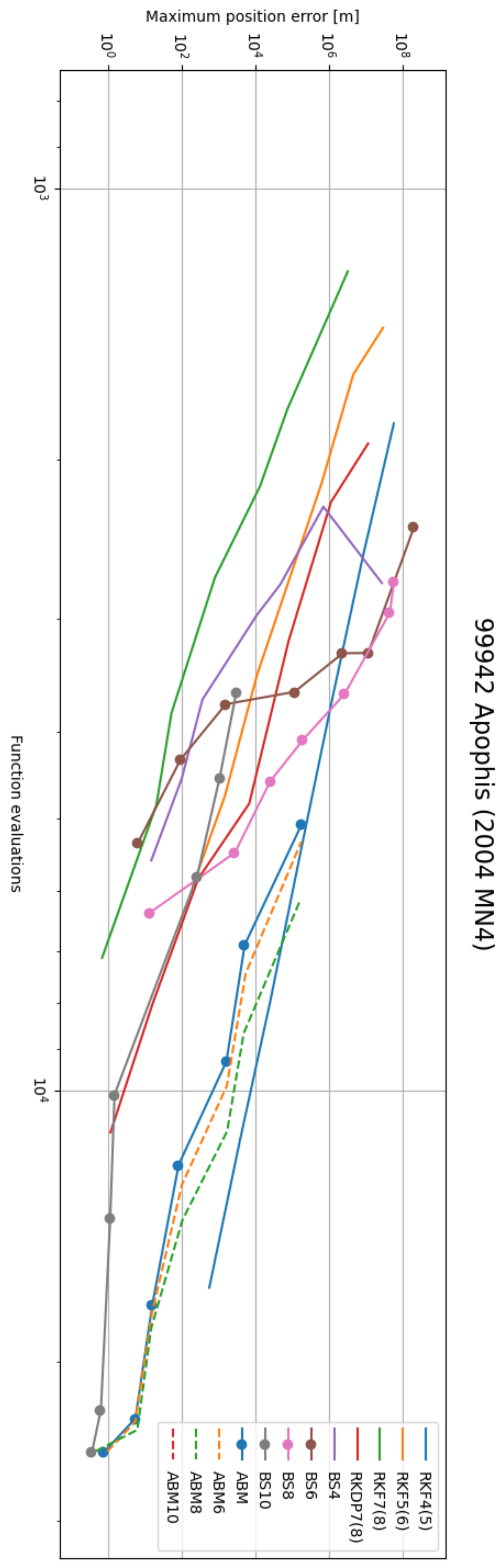


Figure 3.6: Maximum position error as a function of the tolerance and the number of function evaluations for a number of integrators for the propagation of Apophis.

The results of the variable step sizes are shown in Figures 3.5 and 3.6. These results show that the variable step sizes for Apophis require a slightly lower number of function evaluations for a similar order of magnitude error. Although it was previously mentioned that run time is not a good measure, it will be checked in order to see if the number of function evaluations correspond to the run time. The reason for this is that the run time seemed to be significantly larger for the computation of the maximum error of the variable step size integrators compared to the fixed step size integrators. A test was therefore performed by running all integrators for a specific step size (28,800 s) and a specific tolerance $(10^{-10}$. The run times and the number of function evaluations are summed for the fixed and variable step size integrators. The result is shown in Table 3.4 for Apophis and Icarus, where it was found that the run time for variable step sizes on average was more than ten times bigger than the run time of the fixed step size integrators.

Table 3.4: Run time vs function evaluations for 13 runs (all the integrators) for Apophis and Icarus for a variable and a fixed step size. A fixed step size of 28,800 s and a tolerance of 10^{-10} was used for these runs.

Apophis	Sum of run times [s]	Sum of function evaluations
Fixed step size	12.2	$2.66 * 10^6$
Variable step size	101	$1.32 * 10^5$
Icarus	Sum of run times [s]	Sum of function evaluations
Fixed step size	13.5	$2.66 * 10^6$
Variable step size	1335	$2.11 * 10^5$

The table shows that the run time for variable step sizes is significantly larger, even if the number of function evaluations is lower. For Apophis a factor of 10 less function evaluations result in a run time that is 10 times larger, meaning that the difference is somewhere in the order of 100. This is even 1000 for Icarus, and this means that the number of function evaluations in the figure does not resemble the truth at all. The function evaluations in the figures shows a factor of 10 in favour of a variable step size integrator, but this thus also means that the run time is actually 100 times worse. This also shows that variable step size integrators might cause run time issues when the state of all known PHAs need to be integrated. For that reason a fixed step size integrator will be selected.

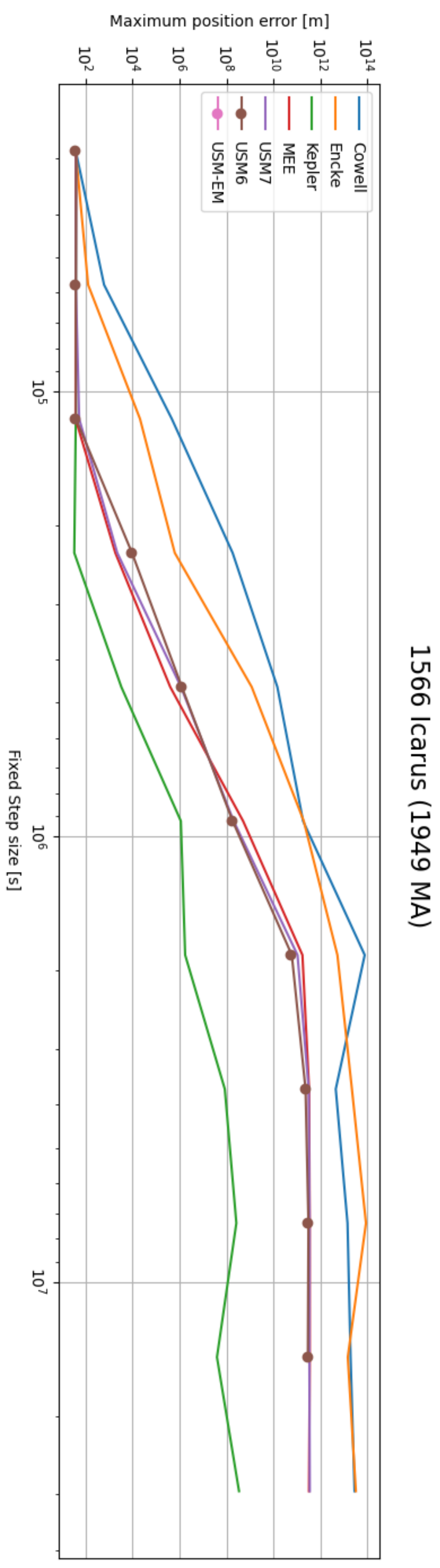
Since this analysis only considered two asteroids, a proper margin on the worst case (Icarus) needs to be taken on the selection of the integration settings. Therefore a step size of 28,800 seconds is selected and the RKDP7(8) integrator is used for the input of the selection of a propagator, which corresponds to an accuracy smaller than 100 m.

3.2.5. Propagator selection

The propagation of the asteroids also requires a propagator. A propagator here is defined as the state representation plus the formulation of the equations of motion. The propagators that will be considered are the following, described in more detail in the literature study: [4]

- Cowell
- Encke
- Kepler (Lagrange Planetary Equations)
- Modified Equinoctial Elements (MEE)
- Unified State Model - Quaternions (USM7)
- Unified State Model - modified Rodrigues parameters (USM6)
- Unified State Model - Exponential Map (USM-EM)

For the investigation of propagators it is necessary to adjust the problem slightly, since the Kepler propagator only propagates the perturbations with respect to a Kepler orbit. If no perturbations are included, it will always turn out as the best propagator, meaning that no fair conclusion can be drawn on that specific propagator. Therefore a solar radiation pressure model is added, which is also considered as one of the perturbations that is probably required in the final propagation. This will be discussed later in more detail (see Section 3.2.6). Figures 3.7 and 3.8 show the comparison between the different propagators for Icarus and Apophis for a fixed step size. It can clearly be seen that the Kepler propagator is the best. Therefore the current selection of propagator and integrator settings is modified: RKDP7(8)



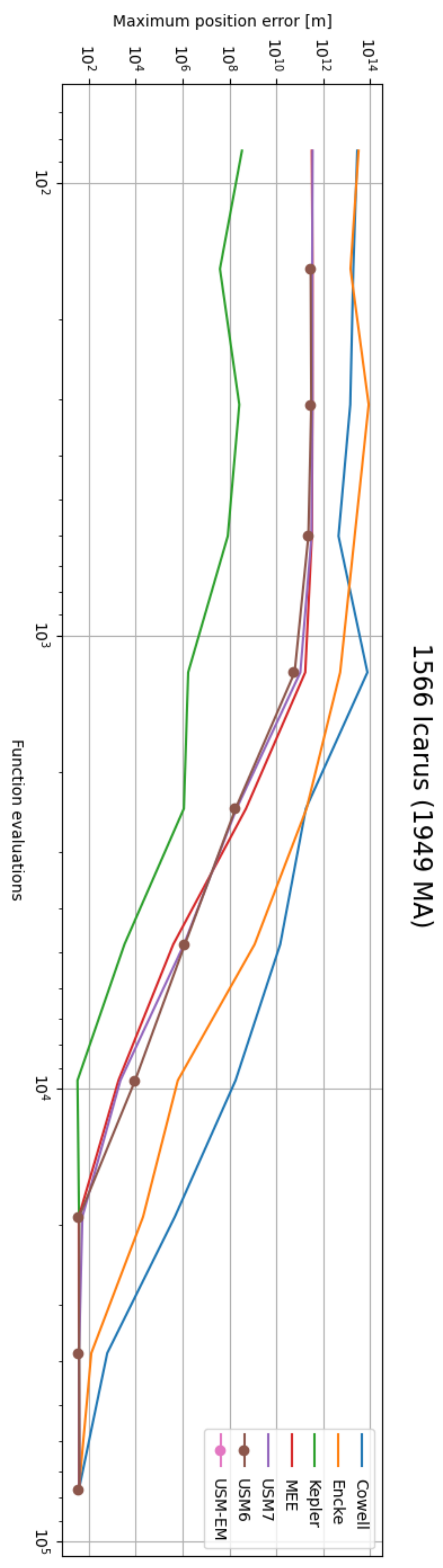


Figure 3.7: Maximum position error as a function of the fixed step size and the number of function evaluations for a number of propagators for the propagation of Apophis.

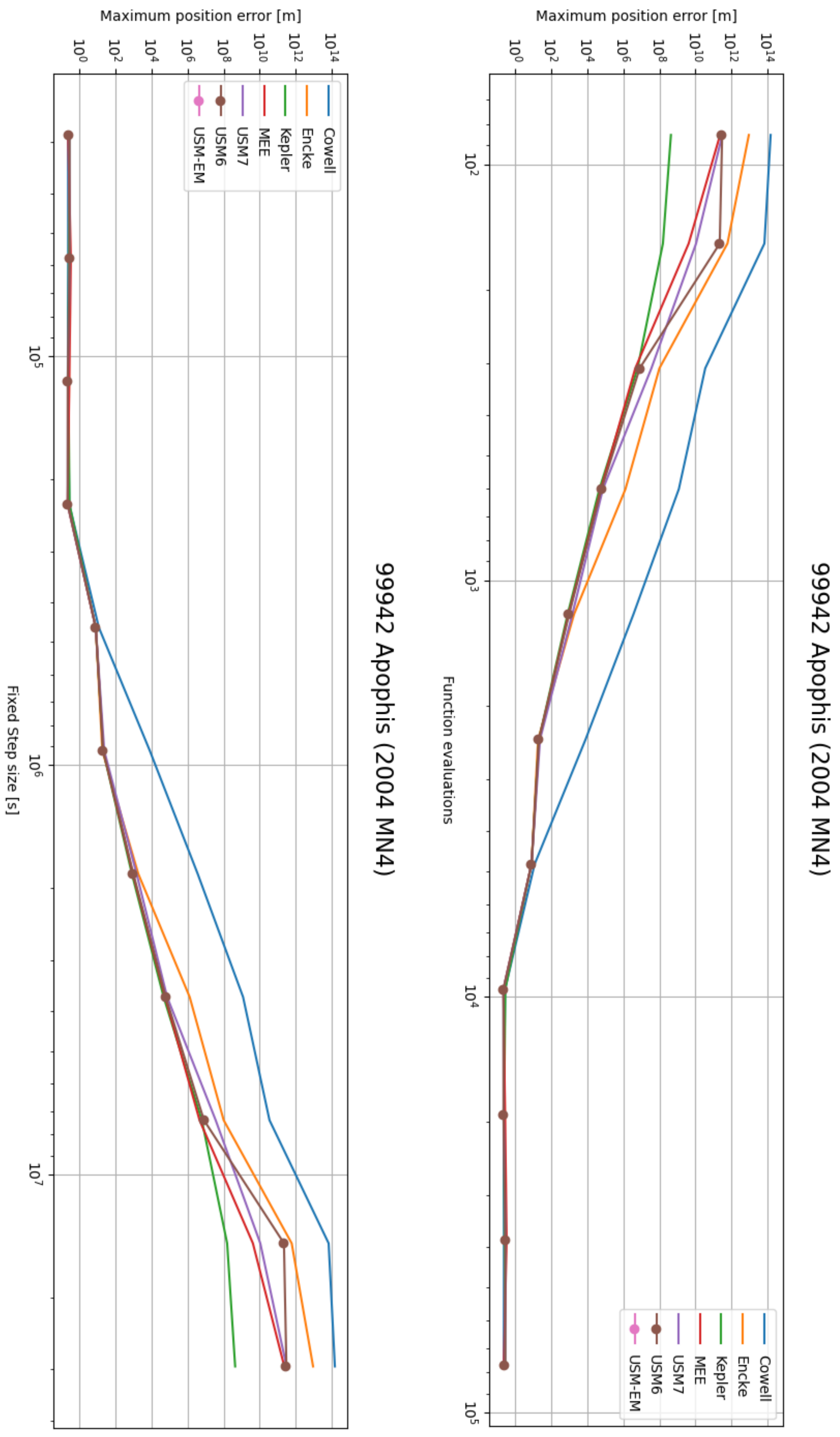


Figure 3.8: Maximum position error as a function of the fixed step size and the number of function evaluations for a number of propagators for the propagation of Apophis.

with a fixed step size of 230,400 s using a Kepler propagator, corresponding again to an accuracy level of 100 m. This will also be used for the investigation of the perturbations in the next subsection.

3.2.6. Perturbations

The integrator and propagator and their corresponding settings have now been selected. It is time to see what effect perturbations have on the orbit of PHAs. If for example no perturbations are required at all, the analytical Kepler orbit would be sufficient and would decrease the computational load by a significant amount. If perturbations are required, it is valuable to investigate which perturbed forces are dominant. A solar radiation pressure model is also tested, with a solar radiation pressure coefficient of 1.1. This value is derived from literature [4]. More information on the uncertainty in this component can be found in the next chapter.

The three asteroids that were selected earlier as the representation of the full data set of PHAs (Apophis, Icarus and 2001XQ) are used for the investigation of the perturbations. The reference is a model with no perturbations and each perturbation is tested by comparing the reference to a model that includes that perturbation. For example, the point mass gravity model of Jupiter is tested by comparing the solution from the reference by a model that includes the reference and the point mass gravity model of Jupiter. The position error is calculated for the differences between the solutions. The results for Apophis can be found in Figures 3.9 and 3.10. The results for Icarus can be found in Figures 3.11 and 3.12 and the results for 2001XQ can be found in Figure 3.13. Relativistic effects were not tested for 2001XQ, as it is expected to turn out similarly compared to the other two PHAs. This might not be the case for the point mass gravity models, as this is the main reason why this PHA is tested, as Neptune and Uranus might be relevant for orbits with a large orbital period.

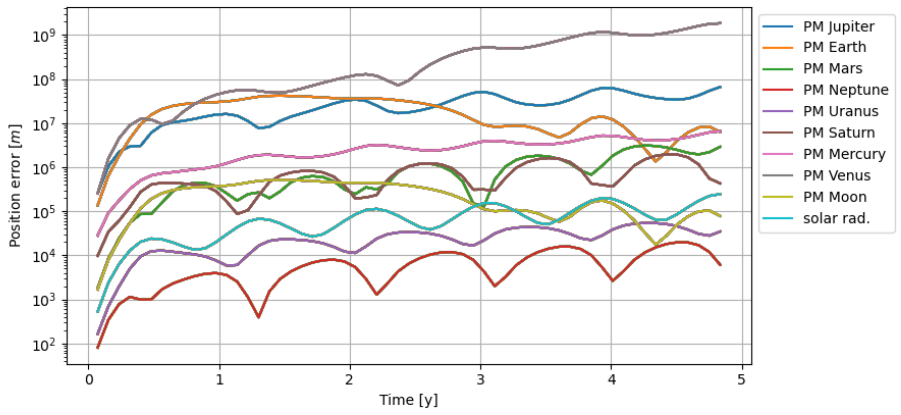


Figure 3.9: Effect of point mass (PM) gravity models and solar radiation pressure on the propagation of the orbit of Apophis. The first data point is left out since a log scale was used.

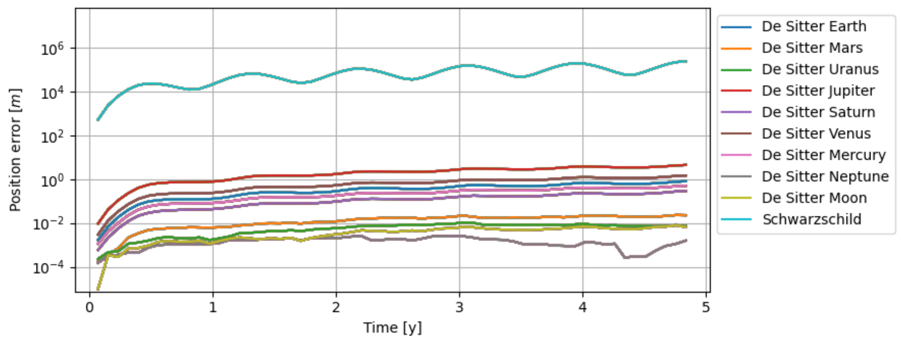


Figure 3.10: Effect of relativity (De Sitter and Schwarzschild) on the propagation of the orbit of Apophis. The first data point is left out since a log scale was used.

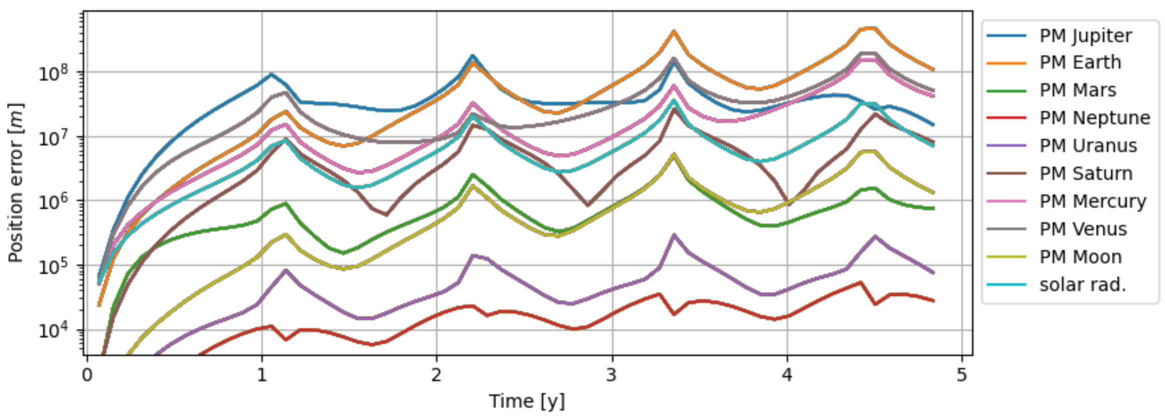


Figure 3.11: Effect of point mass (PM) gravity models and solar radiation pressure on the propagation of the orbit of Icarus. The first data point is left out since a log scale was used.

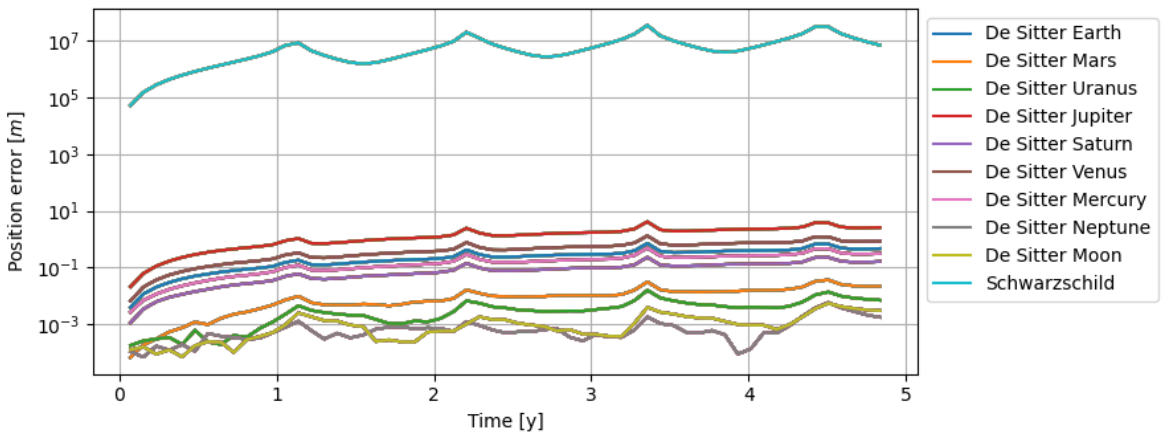


Figure 3.12: Effect of relativity (De Sitter and Schwarzschild) on the propagation of the orbit of Icarus. The first data point is left out since a log scale was used.

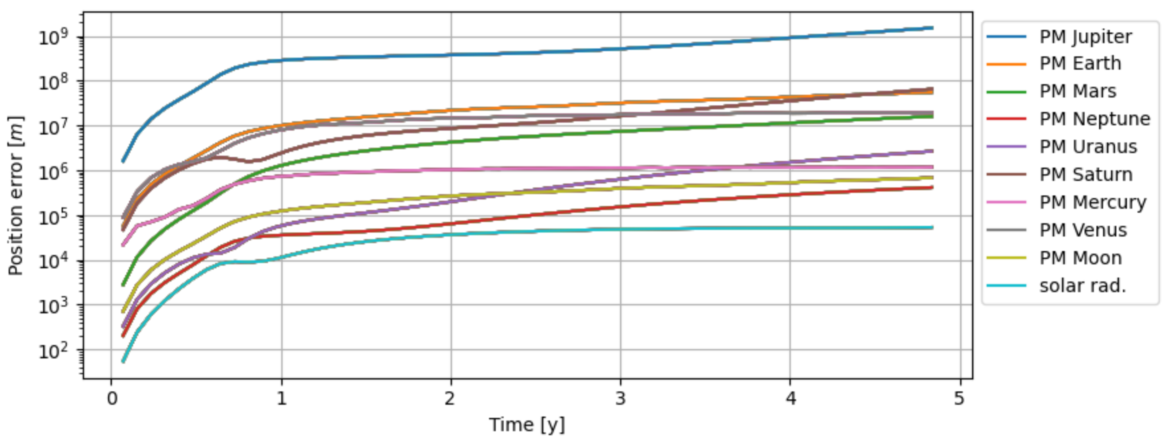


Figure 3.13: Effect of point mass (PM) gravity models and solar radiation pressure on the propagation of the orbit of ’2001 XQ’. The first data point is left out since a log scale was used.

By looking at the results for Apophis and Icarus it can be concluded that point mass gravity models of Jupiter, Venus, Earth, Mercury, Mars, Saturn and the Moon are required, since they have an error larger than the accuracy requirement (1000 km) for at least one of the two cases. A solar radiation pressure model and the relativistic effect Schwarzschild are also required.

From Figure 3.13 it can be seen that Uranus and Neptune can also be important for some cases. This is probably the case for asteroids with a large orbital period, which is the case for 2001 XQ. Therefore it is decided to take Neptune and Uranus also into account.

These results show that an analytical model lacks accuracy and that numerical propagation is required for the propagation of the orbit of PHAs. This means that the computation time becomes significant, and machine-learning techniques might be able to reduce this. The expected accuracy per day for the propagation of PHAs can be computed from the requirements (5 years propagation interval and an accuracy of 1000 km). It is in the order of 1000 / (5*365.25) = 0.5 km/day. From literature it was already concluded that for this accuracy level for interplanetary orbits a solar radiation pressure model is required, as well as gravity field models for planets that are close enough to the orbit [4]. This is line with the results provided here. Moreover, for standard asteroid propagation a Standard Dynamical Model (SDM) is used, which includes point mass gravity models of the Sun, the Moon and all the planets in the Solar System [17]. It also includes point mass gravity models of the three largest asteroids in the Solar System, which are Ceres, Pallas and Vesta, and relativistic effects. The only difference is that solar radiation pressure is left out and that Ceres, Pallas and Vesta are added. Solar radiation pressure is not a big issue for Apophis, only causing an error of 100 km (see Figure 3.9). The effect of Ceres, Pallas and Vesta is even smaller for Apophis, only causing a difference of 0.6 km [17]. It is therefore assumed that the effect of these other asteroids for PHAs in particular is negligible.

It should be noted that higher-order gravity models for planets were also not tested, but according to Georgini [17], this effect is not making any difference until a very close encounter occurs. Chapter 4 describes in more detail the effect of close encounters on the accuracy and whether the integration model needs any adjustment or not.

3.3. Summary

The numerical propagation is set up using Tudatpy, since it is providing all the required utilities. The initial Kepler elements and their uncertainties and physical quantities are imported from the JPL database.

Numerical propagation is required for the state propagation of PHAs, since the errors found for some of the perturbations were larger than the set accuracy requirement. The force models that need to be included and other propagation settings are summarized in Table 3.5. The table also specifies the requirements and the initial settings for the propagation. The model can be used to estimate the position of the asteroid at a future time instance, as well as to determine the effect of uncertainties (to be discussed in Chapter 4).

Table 3.5: Summary of astrodynamic elements required for a numerical propagation of PHAs.

Frame origin	SSB
Frame orientation	ECLIPJ2000
propagation interval [y]	5.0
Accuracy level [km]	< 10^3
Required # function evaluations	$5 * 10^3$
Integrator	RKDP7(8)
Propagator (state representation)	Kepler
Step size [s]	230,400
Accelerations	• Point mass gravity models of Sun, Jupiter, Venus, Earth, Moon, Mercury, Mars, Saturn, Uranus, Neptune. • Solar radiation pressure model (cannonball model). C_r = 1.1. • Relativistic effect Schwarzschild.

4 Uncertainty propagation

This chapter addresses the effects of uncertainties on the PHA trajectories and describes a method to propagate these uncertainties that also fits well within the model established in the previous chapter. This step is required in order to estimate the error induced by the initial uncertainties in the state and in the dynamics model, which serves as an input to the machine-learning process, as described in Chapter 5. This chapter also identifies the relevance of the inputs of the machine-learning step. First of all the procedure is explained and the results are shown in Section 4.1, followed by conclusive remarks on which variables to take into account. After that, the behaviour of the uncertainty is investigated in more detail in Section 4.2 and any changes to the integration model that are required due to the findings in this chapter are described in Section 4.3. The content of this chapter is summarized in Section 4.4.

4.1. Procedure and input variable selection

The initial state and its uncertainties and physical quantities and the uncertainty in the dynamics model are used here to assess the effect on the nominal trajectory of PHAs. Several techniques exist to propagate uncertainties, such as the propagation of a covariance matrix or a Monte-Carlo (MC) analysis. The goal of this thesis project is to replace the state and uncertainty propagation with a neural network and therefore the computational load is not the most important factor. A MC analysis will therefore be used, since it is relatively easy to implement. A state transition matrix is also not used because it can not take into account uncertainties in the dynamics model (e.g. uncertainty in the solar radiation pressure component). MC is a technique that randomly takes initial values from a normal distribution around the nominal value with a specified standard deviation (σ). The initial values are the initial state and the physical quantities, such as the asteroid radius. The standard deviation is defined by the provided uncertainties of these parameters. The first investigation consists of the effect of a single parameter. 50 random values for that parameter are selected and the results are plotted for the initial state for Apophis in Figure 4.1. The propagation time is 22 years and the state is propagated back in time from the provided state in January 2022. Day 8000 thus refers to the year 2022 and day 0 refers to the year 2000 in the figure. The same procedure is also followed for the other two asteroids (Icarus and 2001XQ), which are shown in Figures 4.2 and 4.3. It should also be noted that the propagation time is increased to 22 years to be able to catch a larger time-frame and the integration model was also checked again to make sure that it still is valid for this larger time interval. It turned out that the current integration model did not provide any problems due to the increased propagation time. The maximum error for the largest uncertainty values for the three asteroids is shown in Table 4.1.

Table 4.1: Maximum error of the effect of the uncertainty in the initial state for three different asteroids.

Asteroid	$\Delta a_0 \rightarrow \Delta$r [km]	$\Delta e_0 \rightarrow \Delta$r [km]	$\Delta i_0 \rightarrow \Delta$r [km]	$\Delta \omega_0 \rightarrow \Delta$r [km]	$\Delta \Omega_0 \rightarrow \Delta$r [km]	$\Delta M_0 \rightarrow \Delta$r [km]
Apophis	120	1.25	0.73	26	24	6.5
Icarus	600	31	68	64	40	207
(2001 XQ)	490	80	720	2300	410	1400

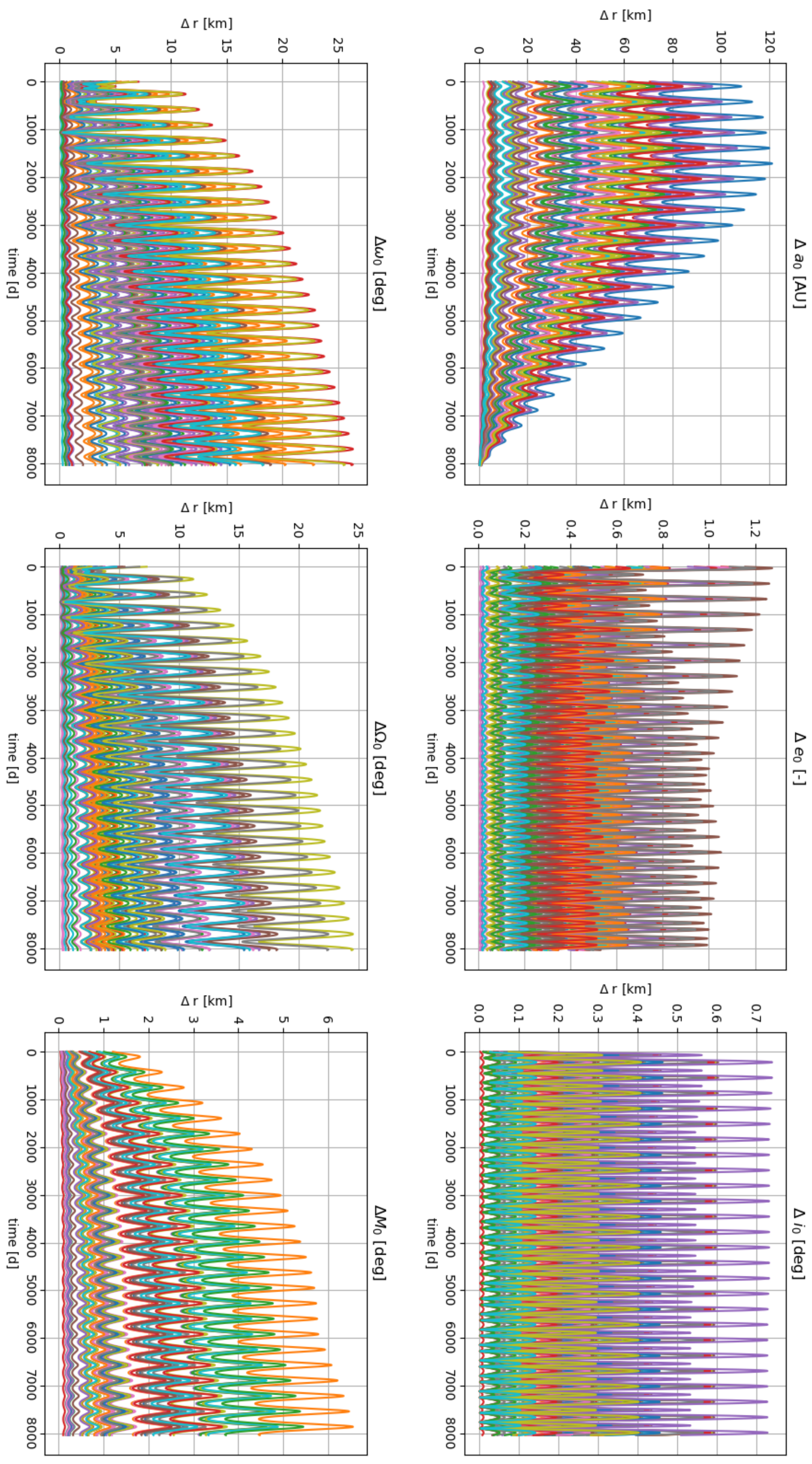


Figure 4.1: Effect of initial state parameters of Apophis. Every subfigure shows solely the effect of one parameter, e.g. the first subfigure shows what the effect a difference in semi-major axis (a) has on the nominal trajectory (Δr).

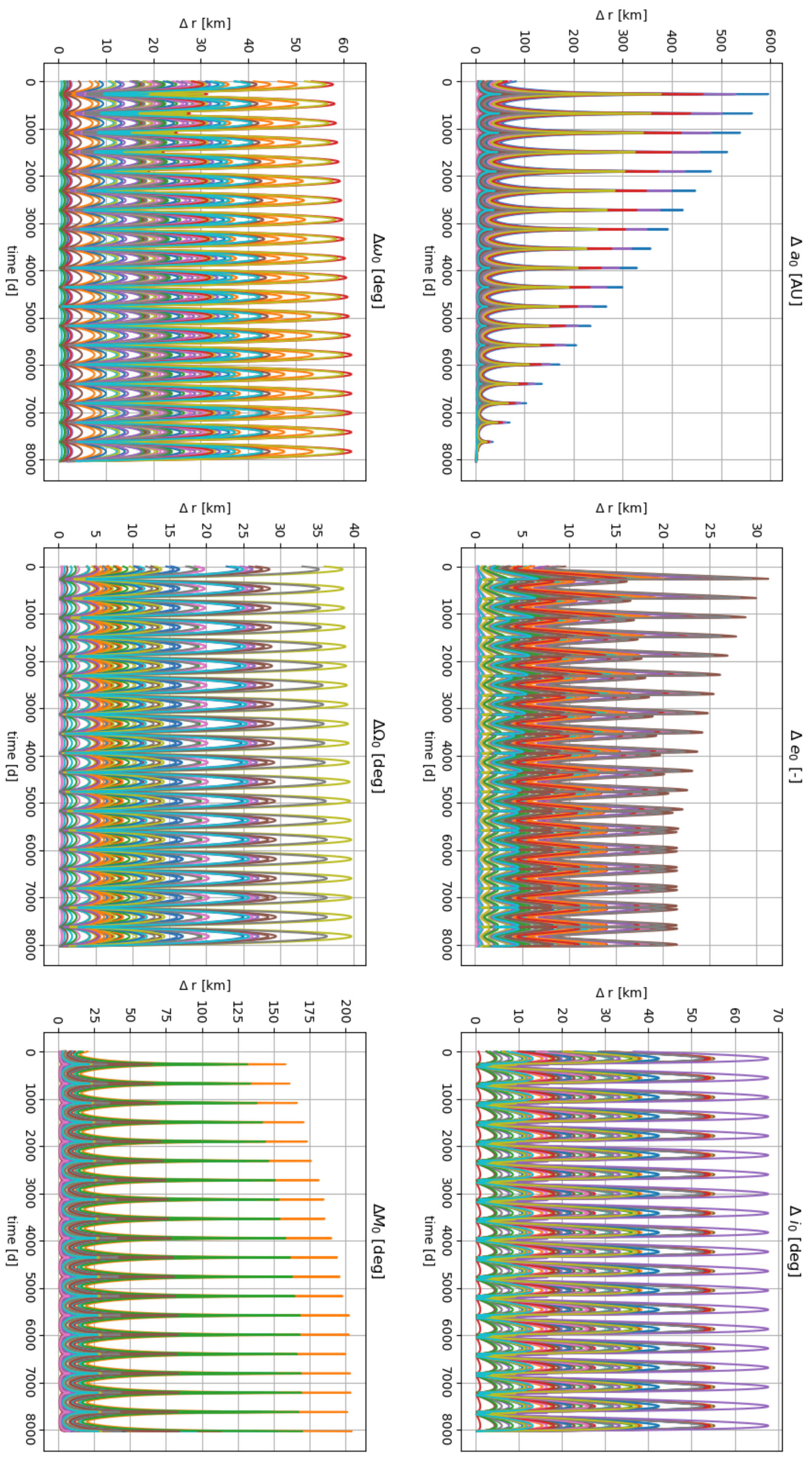


Figure 4.2: Effect of initial state parameters of Icarus. Every subfigure shows solely the effect of one parameter, e.g. the first subfigure shows what the effect a difference in semi-major axis (a) has on the nominal trajectory (Δr).

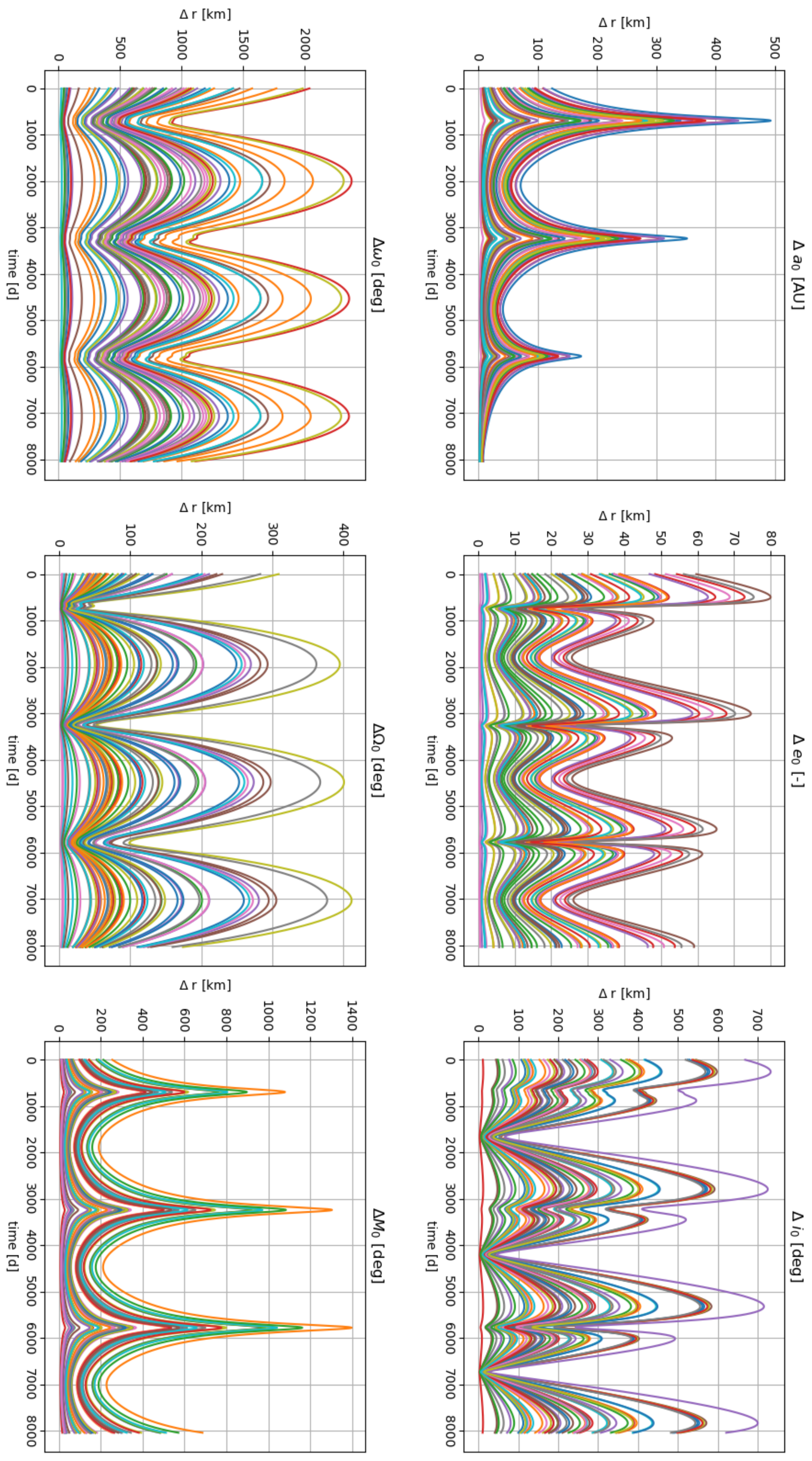


Figure 4.3: Effect of initial state parameters of (2001 XQ). Every subfigure shows solely the effect of one parameter, e.g. the first subfigure shows what the effect a difference in semi-major axis (a) has on the nominal trajectory (Δr).

From Table 4.1 it can be seen that the effect of the semi-major axis is the largest for Apophis and Icarus, whereas this is the argument of pericenter for (2001 XQ). It is difficult to draw a conclusion on the magnitude of the error since these are only three asteroids, but a general conclusion can be drawn. The effect on (2001 XQ) is completely different compared to Apophis and Icarus, and therefore every parameter is considered as important. The only exception is the eccentricity, the uncertainties of which have the smallest effect for all three asteroids and it could therefore be excluded. However the number of inputs reduces only by 2 (The eccentricity and the uncertainty in the eccentricity), which results in only a minor time advantage, as the model contains 15 inputs in total (the initial state parameters (6), the uncorrelated uncertainties in the initial state (6), the radius of the asteroid, the uncertainty in the radius of the asteroid and the propagation interval). Therefore it is decided to just use all these initial state parameters for the neural network.

The uncertainties in the dynamics model are mainly coming from the solar radiation pressure model, because it depends on many factors (e.g. solar flux intensity, geometry of the asteroid and mass of the asteroid) [27]. Uncertainty in the gravity field models is not applicable in this situation, since only point mass gravity models are considered, and it is assumed that the gravitational parameter μ is known with a high accuracy. This also follows from an investigation of the gravitational parameter of Mars, which has an uncertainty in the order of only 10^{-5}, which is relatively in the order of 10^{-9} [26]. This also holds for the uncertainty in the used ephemeris models of all the planets, since these ephemerides are known at meter level (e.g. Mars in DE431 is claimed to have an uncertainty of around 10 m [16]).

Therefore the uncertainty in the solar radiation pressure model is the only factor coming from the dynamics model that needs to be taken into account. The uncertainty of the asteroid radius can be considered to take care of the uncertainty in the solar radiation pressure model, as that perturbation is defined by the area-to-mass ratio. The radius is the only variable that can be changed, as the PHA density is set constant to 2 g/cm^3 for all PHAs [7] and the mass is computed from the radius and the density (see Chapter 3).

The uncertainty of the radius is also provided for some of the asteroids in the JPL database. If this value is not provided the average percentage uncertainty is taken ($\approx$ 15%). The density also contains uncertainty, and will be added on top of the uncertainty of the radius. The density usually varies between 1-3 g/cm^3, and therefore a 1-σ of 50% is taken for the density [7]. As the uncertainties are independent, the total uncertainty percentage is computed in a root-sum-squared way, which results in an average of 52.2%. A safety margin is applied to make sure that the model takes into account all expected uncertainties, and for that reason a percentage of 65% will be used. This means that the MC method needs to be modified, as the radius might become negative in some cases. Therefore the uncertainty is taken into account using Equation 4.1, where $R^{'}$ represents the adjusted radius:

$$\begin{aligned} R^{'} &= R * 2^{\Delta R(\%)+\Delta\rho(\%)} \\ R^{'} &= R * 2^{\frac{\Delta R}{R} \pm 0.5} \end{aligned} \tag{4.1}$$

If the uncertainty of the radius (ΔR) is 50%, then the 1-σ radius value is twice as large ($R' = R * 2^1$) at the right side of the normal distribution and half the radius at the left side of the normal distribution ($R^{'} = R * 2^{-1}$). This is shown more clearly and for all uncertainty percentages in Figure 4.4. The formula prevents a negative radius, since a fraction is determined. The area-to-mass ratio defines the magnitude of the solar radiation pressure model, and the effect of a changing radius can be formulated as shown in Equation 4.2.

$$\frac{A}{m} = \frac{\pi R^2}{\frac{4}{3} * \pi * R^3 * \rho} \propto \frac{1}{R * \rho} \tag{4.2}$$

This formula shows that the asteroid radius has an inverse relationship with the magnitude of the solar radiation pressure model. Thus, the expectation is that a smaller radius increases the magnitude of the solar radiation pressure component and thus the error. Figures 4.5, 4.6 and 4.7 show for again the three same asteroids the effect of the uncertainty in radius, as well as the direct relationship with the difference in radius.

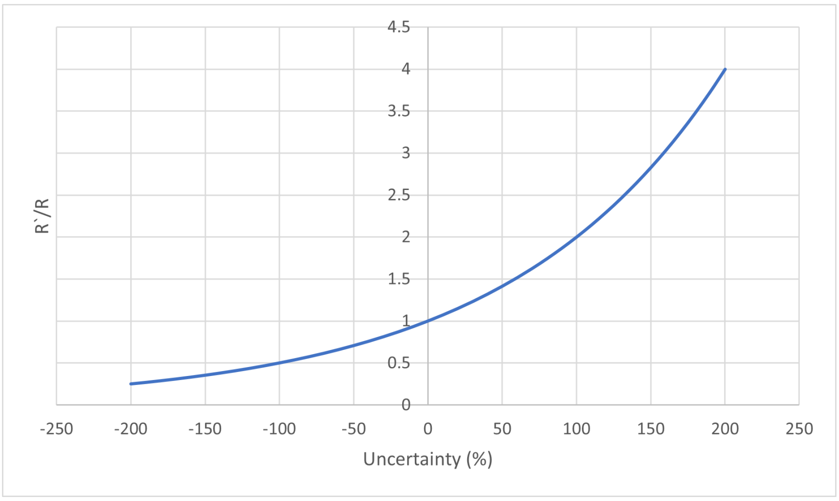


Figure 4.4: Relationship between the effect on the radius of a certain uncertainty percentage. Negative uncertainty percentages show the left-hand side of the normal distribution.

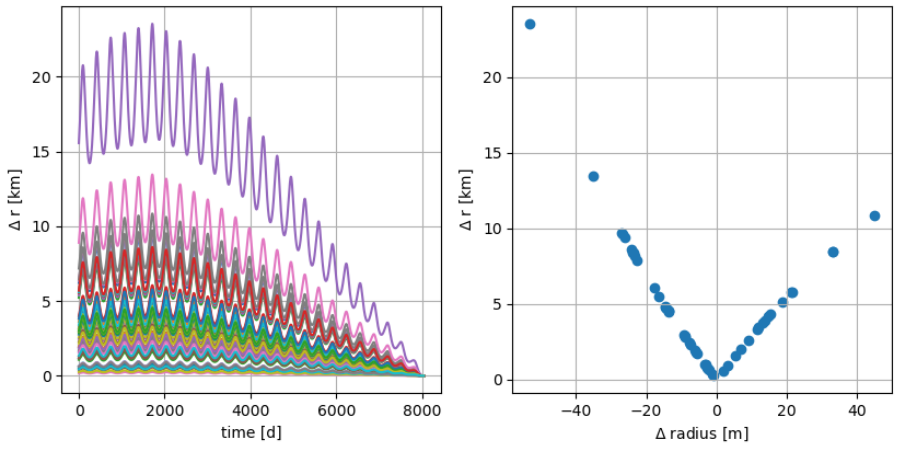


Figure 4.5: Effect of uncertainty in asteroid radius for Apophis. The error behaviour is shown on the left, with again a backward propagation over 22 years. The relation between a different radius and the maximum position difference is shown on the right.

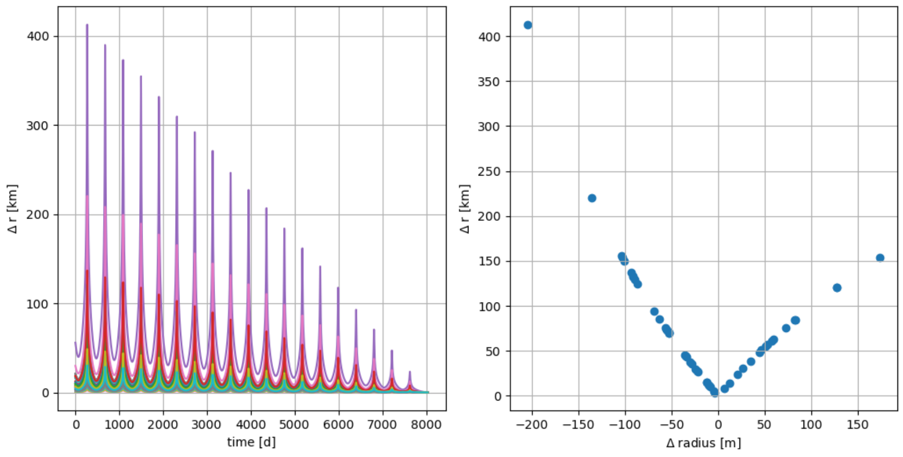


Figure 4.6: Effect of uncertainty in asteroid radius for Icarus. The error behaviour is shown on the left, with again a backward propagation over 22 years. The relation between a different radius and the maximum position difference is shown on the right.

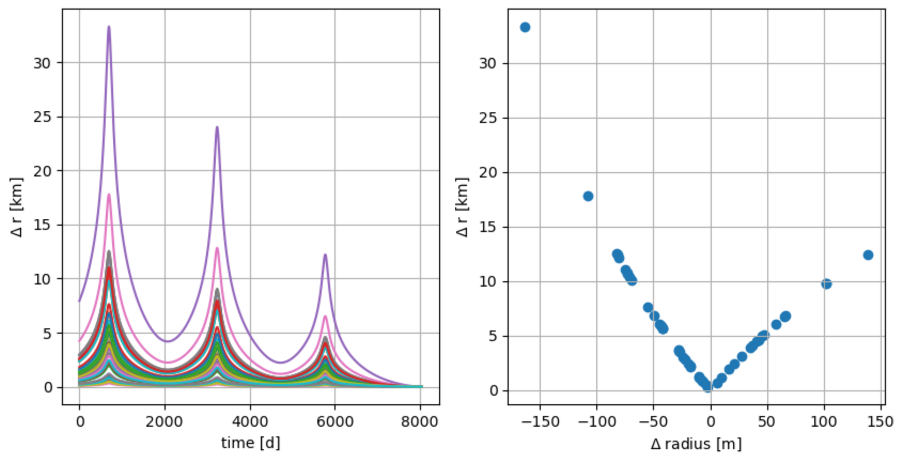


Figure 4.7: Effect of uncertainty in asteroid radius for (2001 XQ). The error behaviour is shown on the left, with again a backward propagation over 22 years. The relation between a different radius and the maximum position difference is shown on the right.

From Figures 4.5, 4.6 and 4.7 it can be seen that the left figures show periodic behaviour, which can be explained by the orbital period; Figure 4.7 shows only about three periods, which is in line with the orbital period of 6.97 years for 2001XQ (see Table 3.2).

The effect of the uncertainty is the largest for Icarus (see Figure 4.6), showing a value of about 400 km. This can be explained by the fact that Icarus is the most sensitive to the solar radiation pressure component, which was also identified in Figure 3.11, in comparison with the results for Apophis and 2001XQ (see Figures 3.9 and 3.13).

The right figures show the relationship with a different asteroid radius. It can be seen that a larger radius means that the error increases, but it can only increase to a certain limit. This limit is defined by the difference between excluding and including the solar radiation pressure model, as a huge radius causes the solar radiation pressure model to be irrelevant (due to the 1/R relationship, see Equation 4.2). When looking at smaller radii, it can be seen that the relation is not linear and that indeed the error becomes much larger. This grows exponentially which shows the sensitivity level of this force model element, making it definitely worth taking this component into account.

The relation between the input and output was also plotted for the initial state parameters and can be found in Appendix B, where a linear relation was found. It is now decided that all the discussed uncertainty parameters will be taken into account: the full initial state and PHA radius.

4.2. Uncertainty error behaviour

The discussed uncertainty parameters will be taken into account, but the behaviour of some of the uncertainties are interesting to investigate in more detail (see Figures 4.1, 4.2 and 4.3). It is assumed that the errors are uncorrelated, if only since there is no information regarding correlation available. The error is increasing over time for a and for R, and slightly for e, but this is not the case for all the other parameters.

Errors generally propagate with time [9]. This increase can be very small, but should be visible over a time period that is large enough. This should thus be visible for all initial state elements. The error however seems to decrease slightly for ω, Ω and M, which is clearly visible for Apophis (Figure 4.1). This is unexpected behaviour and therefore some checks need to be performed in order to make sure that these results are correct. The first check considers the backward propagation procedure. Since Tudat is usually used for forward propagation, it could raise doubts about the correctness of the backward propagation. To check this, the obtained state in the year 2000 found by the backward propagation was propagated forward in time, back to the original state in 2022. This should only result in a very minor difference, as this is only determined by the numerical integrator errors. This test resulted in a difference of less than 1 km, showing that the backward propagation is working fine, as it is way smaller than the set accuracy requirement of 1000 km. On top of that, the initial state uncertainty plots were mirrored compared to backward propagation, which was also as expected. There were some differences observed, but this can be explained by the fact that certain states are more sensitive to uncertainty than others, due to for example a different initial velocity (a larger velocity means a more sensitive start) or a different magnitude of third-body effects initially.

The third-body effect of Earth is expected to be the largest among the perturbations, based on the plots in Chapter 3 where Earth was one of the largest error sources. That is why a check was performed on the initial state uncertainties in a case where Earth is not included. The expectation is that the differences obtained in this situation should not have a trend at all, or at least much smaller than the trend found when including Earth. This was tested for Apophis for a change in ω, since the error is the largest for that initial parameter. The result of this test is shown in Figure 4.8. This figure shows that the trend is indeed not visible anymore and the differences fluctuate more around the same value. This leads to the understanding that Earth is indeed causing the decrease in error.

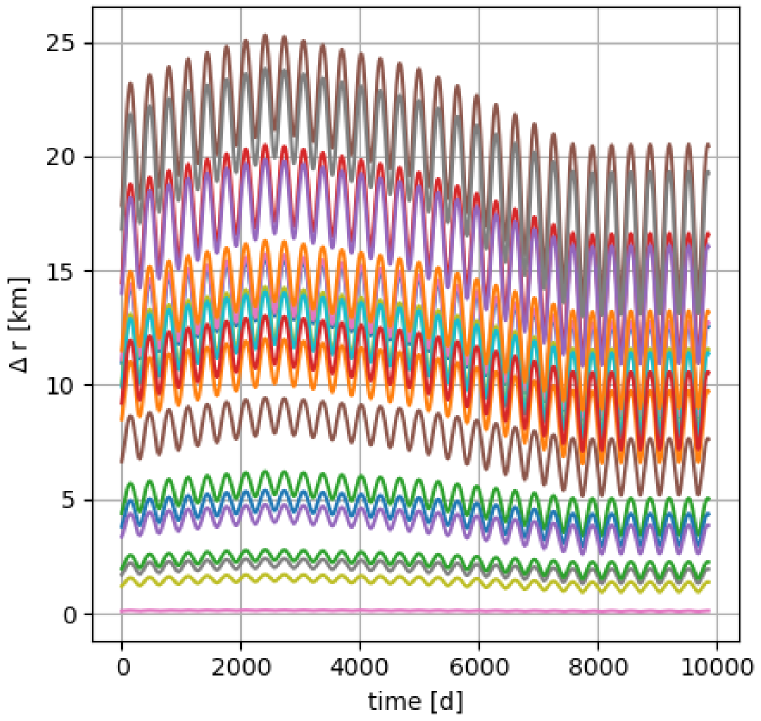


Figure 4.8: Effect of uncertainty in ω_0 for Apophis. Earth is excluded from the force model.

A different extreme case of Earth will now be investigated to gather more insight on the behaviour of the uncertainties. This is a close encounter with Earth, which happens to be the case for Apophis in 1997. The integration time was thus increased from 22 years to 27 years, and the result can be found in Figure 4.9.

Figure 4.9 shows that the differences decrease, as what was seen earlier, until a close approach with Earth occurs in 1997 (which corresponds to day 1000 roughly). After this point the error increases again and is very likely to overshoot the initial error after 27 years. The distance between Earth and Apophis at the time instance in 1997 is about 4*10^6 km (see Figure 4.10). This example shows that the error can also increase, but depends on the position of Earth. One argument against this statement can be that this behaviour occurs due to the fact that a close encounter occurs, which might be a problem for the integrator. To investigate the quality of the integrator in process, an even more extreme case is selected: a close encounter of Apophis in 2029.

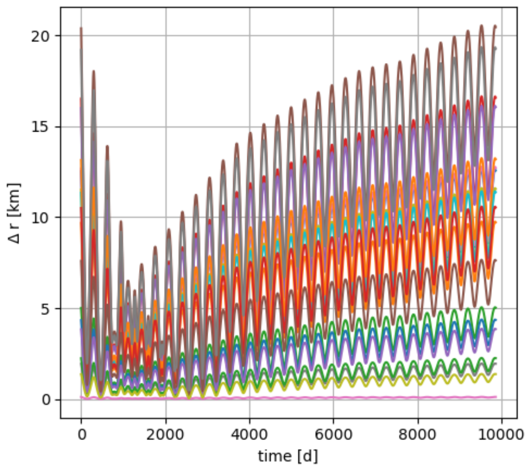


Figure 4.9: Effect of ω_0 on the state of Apophis for a propagation time of 27 years. The state is propagated back in time, starting at t = 10,000 d.

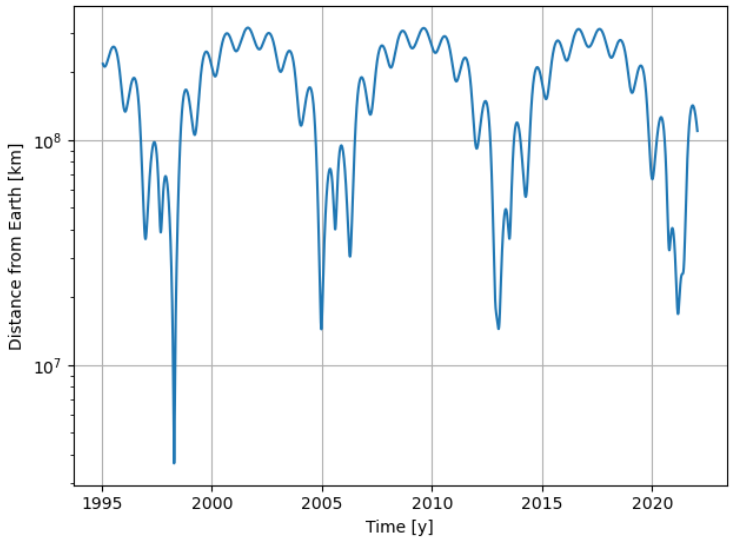


Figure 4.10: Distance between Earth and Apophis for the nominal trajectory. The state is propagated back in time, starting at the year 2022.

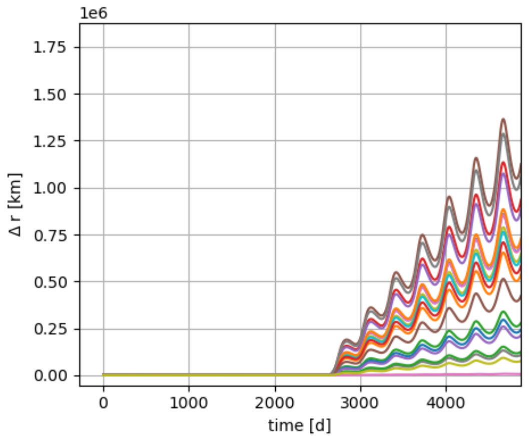


Figure 4.11: Effect of $\Delta\omega_0$ on the state of Apophis for a propagation time between 2022 and 2034 (forward).

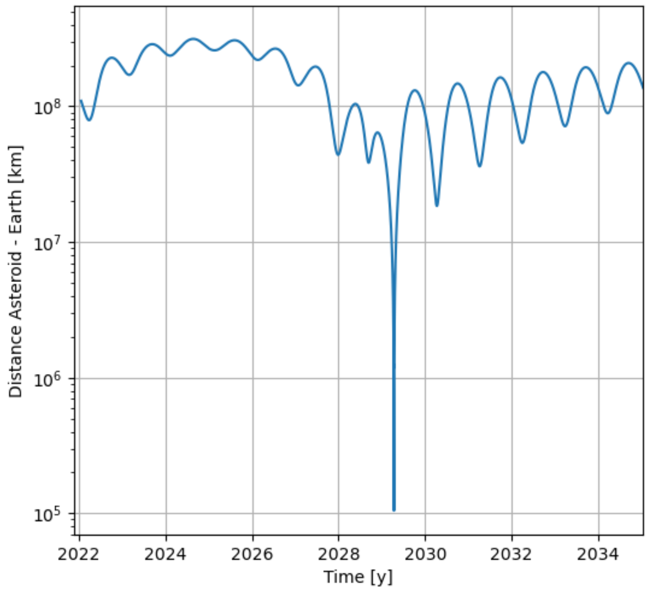


Figure 4.12: Distance between Earth and Apophis for the nominal trajectory.

Figure 4.11 shows that the error in 2029 (day 2700) suddenly increases to around 10^6 km, which is caused by the flyby of Earth in 2029. Figure 4.12 shows that the distance between Earth and Apophis at this time instance becomes 10^5 km. The error suddenly becomes enormous because of this flyby. This is expected, since a close approach means that the point mass gravity model component of Earth is larger and thus is having more effect on the orbit. However a large difference can also be explained by the integration error, as at a certain point the integration model can not predict the next state accurately; the time step is too big to be able to keep up with the fast-changing dynamics of the asteroid.

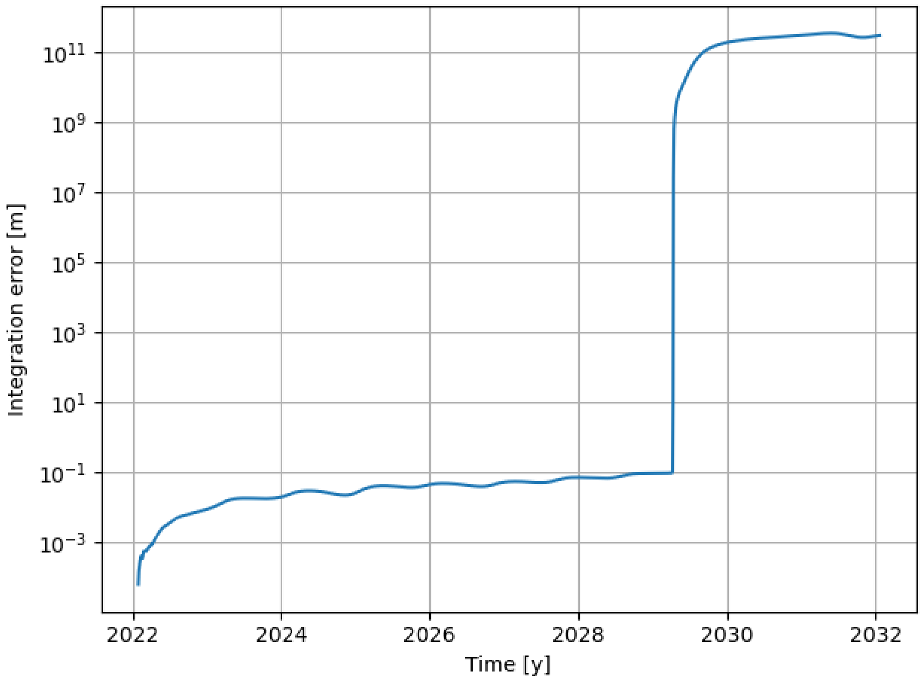


Figure 4.13: Integration error for a 10-year forward propagation of Apophis, using a step size of 230,400 seconds.

Figure 4.13 shows the error of the integration model for the close approach in 2029, and it can be clearly seen that the error increases significantly in 2029. The close encounter in 1997 however does not show this result. Thus, the behaviour of uncertainty errors are sensitive to a close-by planet and can cause the error to decrease. Moreover, very close encounters can cause the integration model to become very large, and to be able to also take into account these special cases, the current integration model needs adjustment.

4.3. Integration model adjustment

Clearly, the integration model needs to be adjusted to be able to catch limiting cases, as mentioned before. Since all the verification and validation is already performed for the current integrator, it is decided to ”manually” change the step size during the period of a close approach. The time step is adjusted when the distance between the asteroid and Earth becomes smaller than 10^7 km. The step size is reduced to 10,000 s between a distance of 10^6 and 10^7 km. The step size is reduced again if the distance becomes smaller than 10^6 km. The step size then becomes 1,000 s. This is done not only for Earth, but also for any other planets to make sure that no error is made in the integration model if the asteroid happens to have a close flyby with another planet. These specific numbers were chosen to stay within the accuracy requirement, as can be seen from Figure 4.14 for the specific Apophis case. The limits (10^6 and 10^7 km) were selected based on the velocity and the step size. The orbital velocity of an asteroid is on average 20 km/s [19]. When using a step size of 230,000 s, a distance of $ds = V * dt =$ 20*230,000 = 4.6*10^6 km can be travelled in one time step. In order to be on the safe side the limit is set at 10^7 km. The same reasoning holds for the second time step change.

Not only the integration model is different depending on the distance to planets, the perturbations that need to be taken into account can also be different. Earth is for example not a perfect sphere, and the force from Earth that acts on the asteroid can therefore slightly differ. This is not a problem for most of the cases, but for the close flyby in 2029 of Apophis this might be a problem, since this was already identified in literature [17]. Therefore the difference between a spherical harmonics gravity model with degree and order 2 and the nominal case (a point mass gravity model) is calculated for five years after the close flyby of 2029. The result can be found in Figure 4.15.

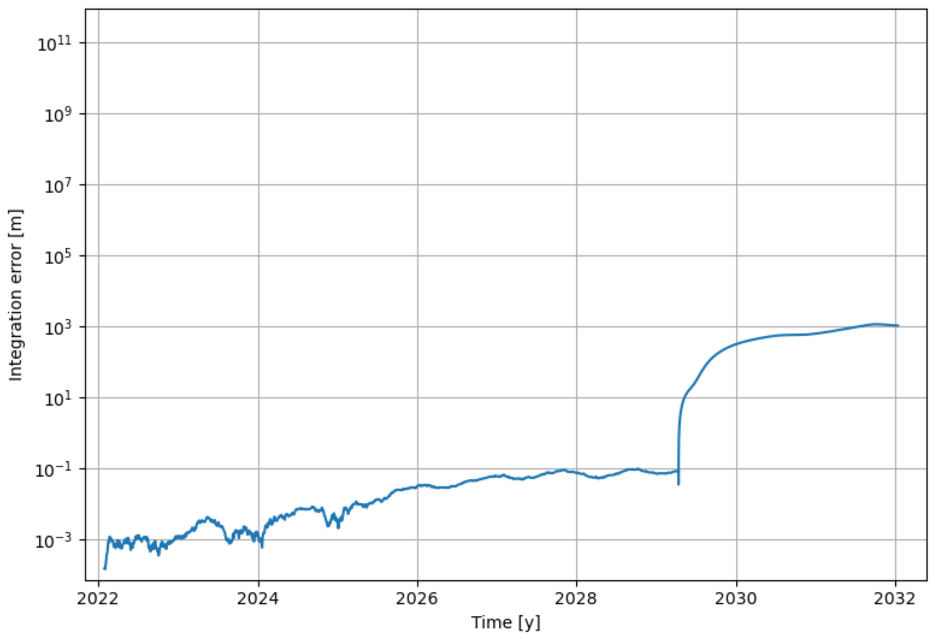


Figure 4.14: Integration error for a 10-year forward propagation of Apophis, using a step size of 230,400 seconds with and adjustment to 10,000 s and 1,000 s if Apophis becomes too close to Earth.

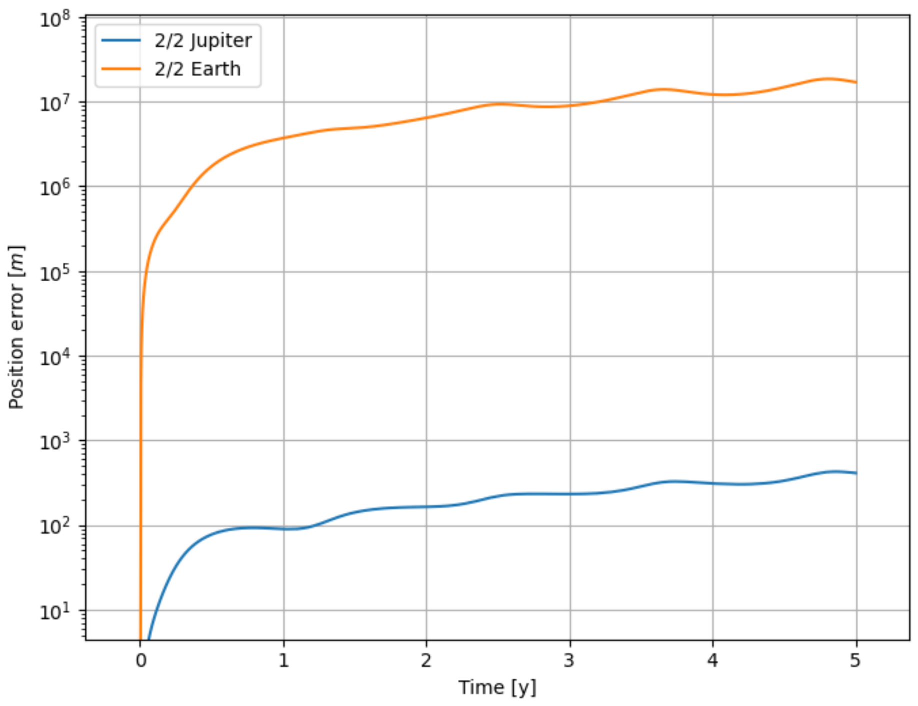


Figure 4.15: Difference between a spherical harmonics gravity model with degree and order 2 (so including the J2 effect) and the nominal case (a point mass gravity model) for Earth and Jupiter.

Figure 4.15 shows that the influence of the spherical harmonics gravity model indeed results in a difference larger than 1,000 km after a time period of 0.4 years after the close flyby for Earth. Jupiter is also shown as a comparison, which is not a problem at all, as expected. This shows that the spherical harmonics gravity model needs to be taken into account if the asteroid becomes close to Earth or any other planet. This is implemented in a similar way as was seen for the step size; once the distance between the asteroid and Earth or another planet becomes smaller than 10^7 km, not only the step size is adjusted, but the point mass gravity model is changed to a spherical harmonic gravity model with degree and order 2. The difference between the variable dynamics model and a model that includes J2 during the whole integration period is shown in Figure 4.16, which shows that the variable model is performing well.

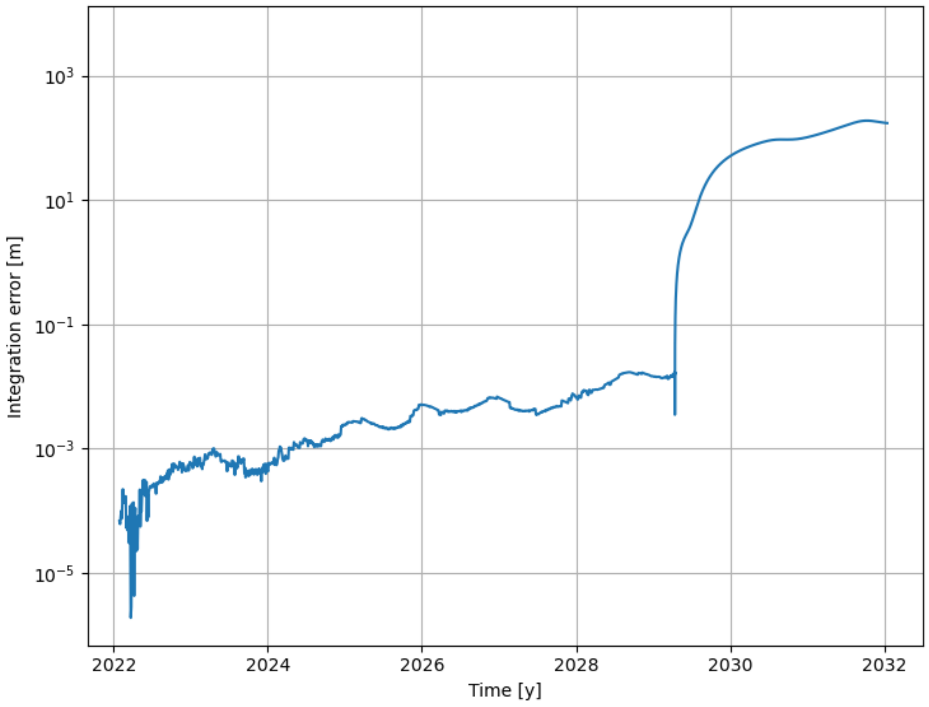


Figure 4.16: Difference between a model that includes J2 and a variable dynamics model for Earth for the specific Apophis case.

4.4. Summary

The propagation of the uncertainties in the state and the solar radiation pressure showed that all uncertainties are relevant in order to estimate the position error. This information will be used when the inputs will be selected for the neural network, which is discussed in Chapter 5. The behaviour of some of the uncertainty parameters is however not as expected, as the error decreases over time. This can be explained by the presence of flybys (close and further away) of Earth, since removing Earth from the perturbations showed that the error was not decreasing anymore. Another investigation showed that the error eventually increases after a relatively close flyby. It is therefore concluded that the unexpected behaviour can be explained by the fact that the Earth causes the orbit to be sensitive, since the error can also increase due to Earth.

Nevertheless, there appears to be an issue when a flyby is too close to Earth or any other planet in the Solar System. The integration error then becomes very large as the initial investigation and selection of the integrator did not take into account special cases. The integration model is therefore adjusted. Moreover, the spherical harmonics gravity model of that planet becomes relevant in the case of close flybys, and the relevant perturbations are therefore also modified. The changes are shown in Table 4.2, and are shown with respect to the original settings defined in Table 3.5.

Table 4.2: Step size adjustment to take into account close flybys.

Distance from planet	> 10^7 km	10^6 - 10^7 km	< 10^6 km
Step size Gravity model	230,400 s Point mass	10,000 s Point mass and J2	1,000 s Point mass and J2

5 Machine learning

This chapter provides the necessary information and selection of data that is required for the machine-learning step of the propagation. The data are generated by using the integration model for PHAs as explained in the previous chapters. First the method is explained in Section 5.1, after which the inputs and outputs are selected in Section 5.2. After that an initial investigation takes place in order to see how the neural network responds to the data, which is addressed in Section 5.3. After that the tuning process is described in Section 5.4. The best result that follows from the tuning process is used for the fine-tuning and interpretation, which will be discussed in the next chapter.

5.1. Method

Before explaining the machine-learning method, it is necessary to explain the reason why machine learning is used and not a different technique. For the purpose of this thesis project it is necessary to find a propagation technique that is either more accurate and/or faster (preferably both) to be able to end up with a more efficient propagation technique. More details on the efficiency are provided once the results are discussed, in Section 6.12. Machine learning is known to be very fast, as predictions can be computed almost instantly once the model has been derived. Machine learning is also a technique that can be used as part of a hybrid propagation technique and as a stand-alone model. This means that there are multiple options to try and compare. Machine learning also has the advantage of being a new method, which means that there still is enough to investigate [43]. Other similar techniques, such as statistical time-series models have been investigated more already and is mostly only applicable as part of the hybrid propagation technique.

Machine learning is capable of reproducing a certain behaviour, as long as it is not too complicated. Machine learning requires a data set with a number of inputs and outputs. The form of this data set can be seen more clearly in Equation 5.1, where p is the number of inputs, q the number of outputs, and n the number of data points.

$$\begin{aligned} f(i_{0,0}, i_{1,0}, \cdots, i_{p,0}) &= o_{0,0}, o_{1,0}, \cdots, o_{q,0} \\ f(i_{0,1}, i_{1,1}, \cdots, i_{p,1}) &= o_{0,1}, o_{1,1}, \cdots, o_{q,1} \\ \vdots \qquad & \qquad \vdots \\ f(i_{0,n}, i_{1,n}, \cdots, i_{p,n}) &= o_{0,n}, o_{1,n}, \cdots, o_{q,n} \end{aligned} \tag{5.1}$$

The inputs are used to estimate the output, but in order to do that the neural network needs to be trained. Therefore a fraction of the data points is usually taken as training data. Once the neural network is trained, i.e. the model determined, the rest of the data (referred to as test data) is used to investigate how well the neural network estimates the output. How exactly the neural network works will be described later, it is now only necessary to decide what type of behaviour will be captured by this neural network, which means what the inputs and outputs of the neural network will be.

There are multiple options regarding the selection and the setup of these inputs and outputs, and one of these options is to use the measurement data (azimuth, elevation, distance, velocity) directly. This means that an orbit solution needs to be fitted through all the data points, such that the differences for every measurement point can be extracted (see Chapter 2). Since the interest of this study is only PHAs, it is of high importance to know the position after a certain time period in the future to be able to predict with a certain probability if a PHA hits Earth or not. This means that after obtaining the orbit solution the asteroid state needs to be propagated somehow to end up at the desired epoch in the future to be able to estimate the corresponding position.

The second option would be to use the measurement data indirectly. An orbit solution is already fitted by JPL, as the orbital parameters and their uncorrelated uncertainties are available [33]. These can be used to propagate the dynamics of all PHAs in the database, to be able to estimate the state and the uncertainty at a future time instance. The first option thus requires an extra step and is therefore more difficult. It is only useful to increase the accuracy of the Kepler elements and uncertainties provided by the JPL database (expected under the condition that more observations are available; JPL did a good job undoubtedly) and does not influence the computational speed of the propagation. The second option will therefore be used in this research, with the side note that the other option might be a solid recommendation. The measurement data will still serve as a verification and validation method for the given Kepler elements and uncertainties, this is discussed in more detail in Chapter 7.

As mentioned before, machine learning can be used as part of a hybrid propagation technique, or as a stand-alone model. This thesis focuses on the performance of machine learning as a stand-alone model, because the Earth distance is the only measure that is dependent on the model, and it might thus only be an issue for the Earth distance. Moreover, a hybrid propagation technique requires another analytical or numerical model and it is difficult to define this model. The accuracy of this model should be in between an analytical solution and the numerical integration model defined in Chapter 3, and the computational speed gain depends on the chosen model. The model with the most gain in computational speed is a stand-alone model, as no other analytical or numerical solution is required. A stand-alone model is therefore selected, and only if there is enough time the other option will be selected as well to compare the solutions, if it turns out that the Earth distance can not be predicted within the requirements set (e.g. if the Earth distance is 10,000 km and the model predicts it with an error of 100,000 km, the model is useless). The performance of the neural network is therefore the most important measure, and this will be discussed in more detail in Chapter 6.

5.2. Inputs/outputs selection

The previous chapter already provided the information to include all the uncertainties for the propagation. Since this information is important to predict the position of an asteroid and the effect of the uncertainties, all these variables will be used as input variables for the neural network. This thus includes the initial state parameters (6), the uncertainties in the initial state (6; correlations are unknown), the radius of the asteroid and the uncertainty in the radius of the asteroid. The propagation time is another input, which is used to be able to estimate the state of a PHA at any arbitrary epoch. This means a total of at least 15 inputs are required for the neural network. The output must contain information regarding the minimum distance between Earth and the asteroid and the effect of the uncertainties on this distance. The minimum distance to Earth is however impossible to predict, since there is no information regarding the position of Earth in the input data. That is why the x,y and z-position of the PHA with respect to the Sun will be part of the output, instead of the minimum distance to Earth. The minimum Earth distance can then be calculated manually from the PHA position using the known ephemeris data of Earth. More details on the prediction and evaluation of the Earth distance can be found in Section 6.1. This brings the total number of outputs to four. These are the inputs and outputs that are selected for the preliminary phase, since machine learning is very problem specific [45]. This means that a different combination of inputs and outputs might result in a better model, which will be discussed later.

The values of the inputs and outputs are extracted from the integration, and the error due to the uncertainties is computed as an average error. An MC simulation is performed on the nominal trajectory of each asteroid to compute a series of errors and hence also an average error over the propagated time period. The MC simulation consists of 100 propagations. The seed is set to zero to make sure that the results are reproducible.

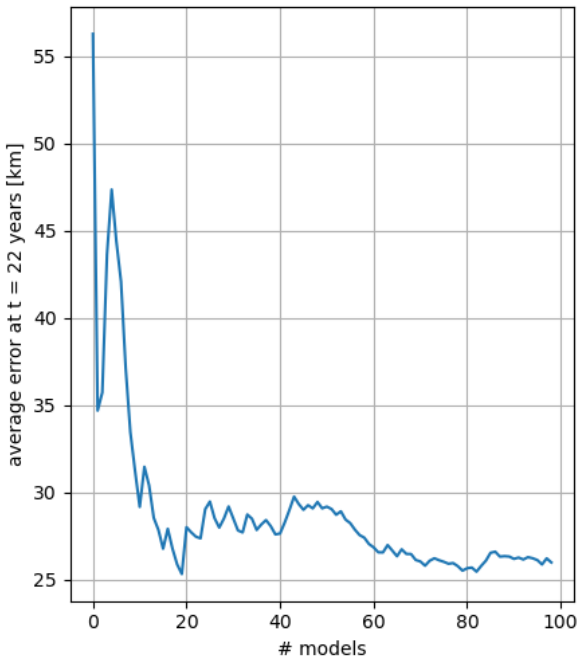


Figure 5.1: Average error of Apophis after an MC analysis with a certain number of propagations with uncertainty effects.

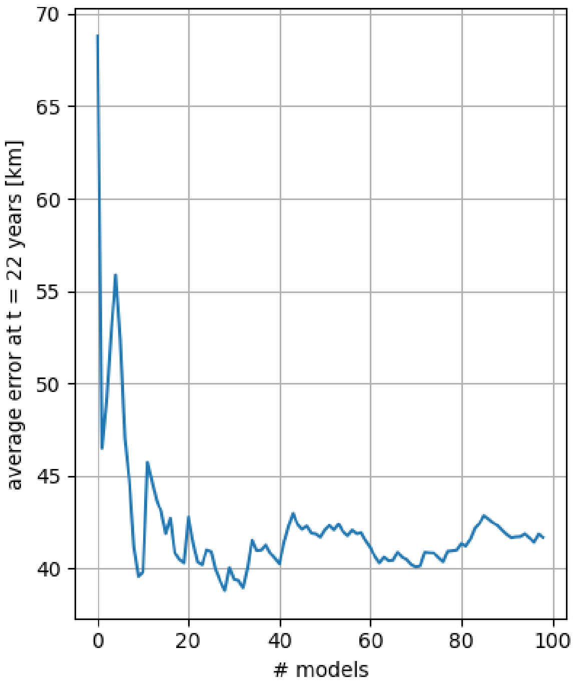


Figure 5.2: Average error of Icarus after an MC analysis with a certain number of propagations with uncertainty effects.

The number of simulations in the MC analysis to compute the error needs to be addressed, to end up with a realistic assessment of the effect of the uncertainties. Figure 5.1 shows the average error after propagation over a time span of 22 years for Apophis for various numbers of simulations in an MC analysis, and is showing that after 40-50 propagations the error does not change more than 10%. A similar behaviour is found for Icarus, shown in Figure 5.2. Therefore 50 propagations will be used and this means that in total 2200*50 = 110,000 propagations are required to create the input file for machine learning. Obviously, going for higher numbers would have consequences for CPU time.

The to be used data as input is extracted multiple times during the propagation. The most interesting points occur when the distance between the asteroid and Earth is the smallest, which are always included. How many data points are used will be decided later once more information is gathered about machine learning.

5.3. Initial investigation

The approach taken in this thesis project is to first investigate the behaviour of the neural network in more detail by testing only a small neural network with less inputs and outputs. The information extracted from this investigation will then help to define the inputs and outputs of the final neural network. The acquisition of the data is discussed first, after which the neural network is created and trained.

5.3.1. Data acquisition

A total of 100 asteroids will be taken as representative asteroids for the full data set. In order to already draw some conclusions based on the data set of 100 asteroids it is necessary to have a good representation of all the different types of orbits. Figure 5.3 shows the Kepler elements of the 2241 PHAs in blue, and the data set of 100 PHAs in orange. As can be seen from Figure 5.3, the 100 PHAs are a viable representation of the full data set. When looking at the full data set it is worth noticing that the

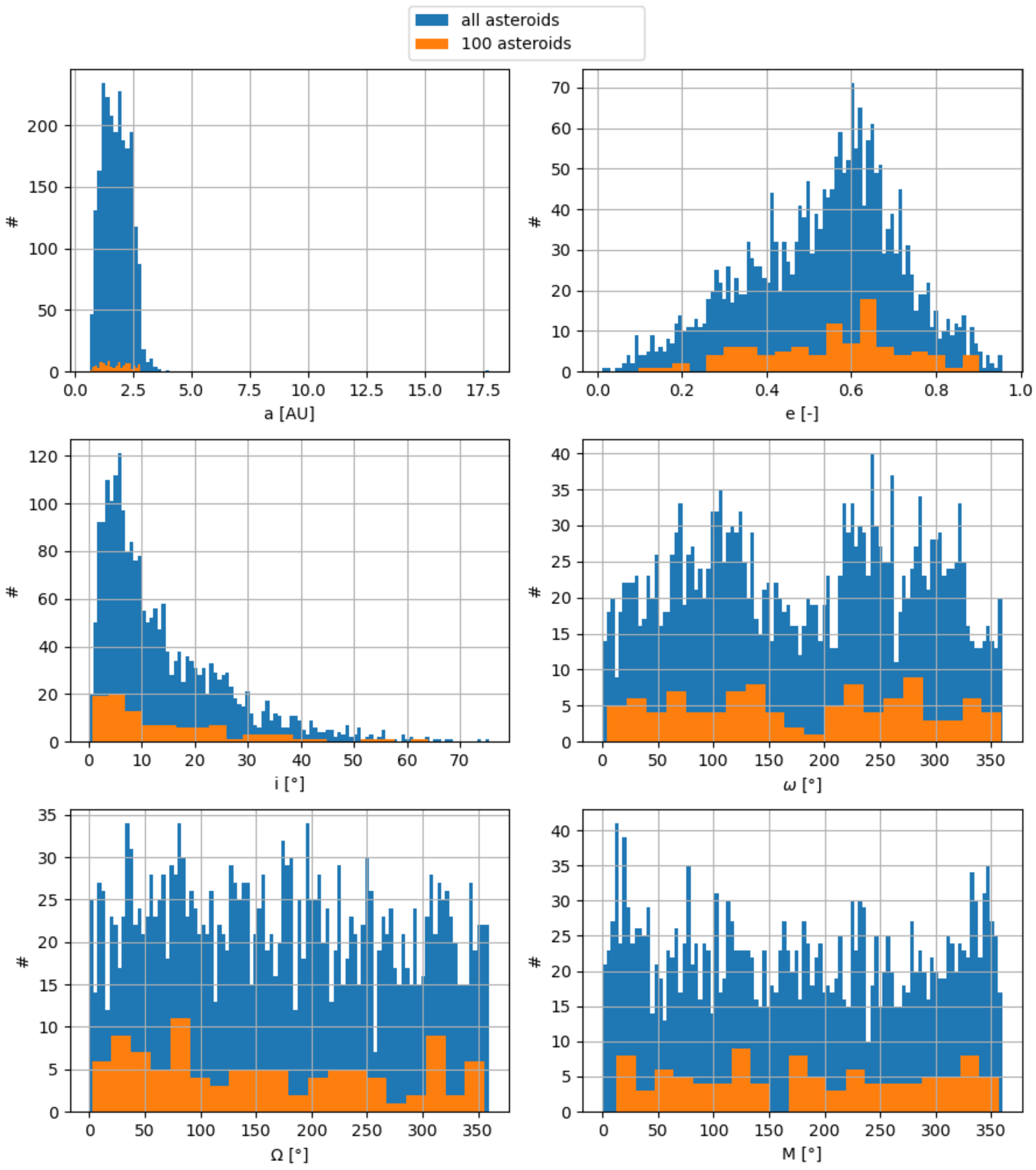


Figure 5.3: The range of Kepler elements of the initial states of 2241 known PHAs, shown in blue. The Kepler elements of a sample of 100 asteroids are shown in orange.

inclination only spans between 0-80 degrees, showing that all of these asteroids orbit in a prograde direction. From the 1.2 million known asteroids, only 150 (= 0.012%) asteroids orbit in a retrograde direction [33]. This shows that the range of inclinations is realistic. There are more asteroids in this data set with an inclination closer to zero, meaning that the spread over the range is not perfectly even, but there still is quite some variety over the spectrum. This is also the case for the eccentricity, where more asteroids are known with an eccentricity around 0.6. One last thing to note about this data set is that there is one asteroid with a semi-major axis of 17 AU (not directly visible in the figure). This is only one outlier and if this impacts the results then it could always be deleted later. It should be noted that deleting information should be handled very carefully, as it might mean that the data does not represent the behaviour of a PHA anymore and the outcome of the exercise might become biased.

The inputs for the data set of 100 asteroids will be the same as discussed in Section 5.2. The number of machine-learning inputs, data points, per asteroid will be set to 50, such that the data set contains 5,000 data points. These 50 points will be selected as follows: the distance between the asteroid and Earth will be taken as the main aspect element. All the minimum values in the time span of the integration are first taken as data points. This is done to make sure that the most interesting data points, the close approach points, are taken into account. The rest of the required number of data points is sampled from the time array of the integration. So for example, if 10 minimum values are found, and if the time array consists of 160 entries, then the 40 other data points are taken from the time array (the 4th, the 8th, the 12th, etc). This is done in order to make sure that all PHAs are equally represented.

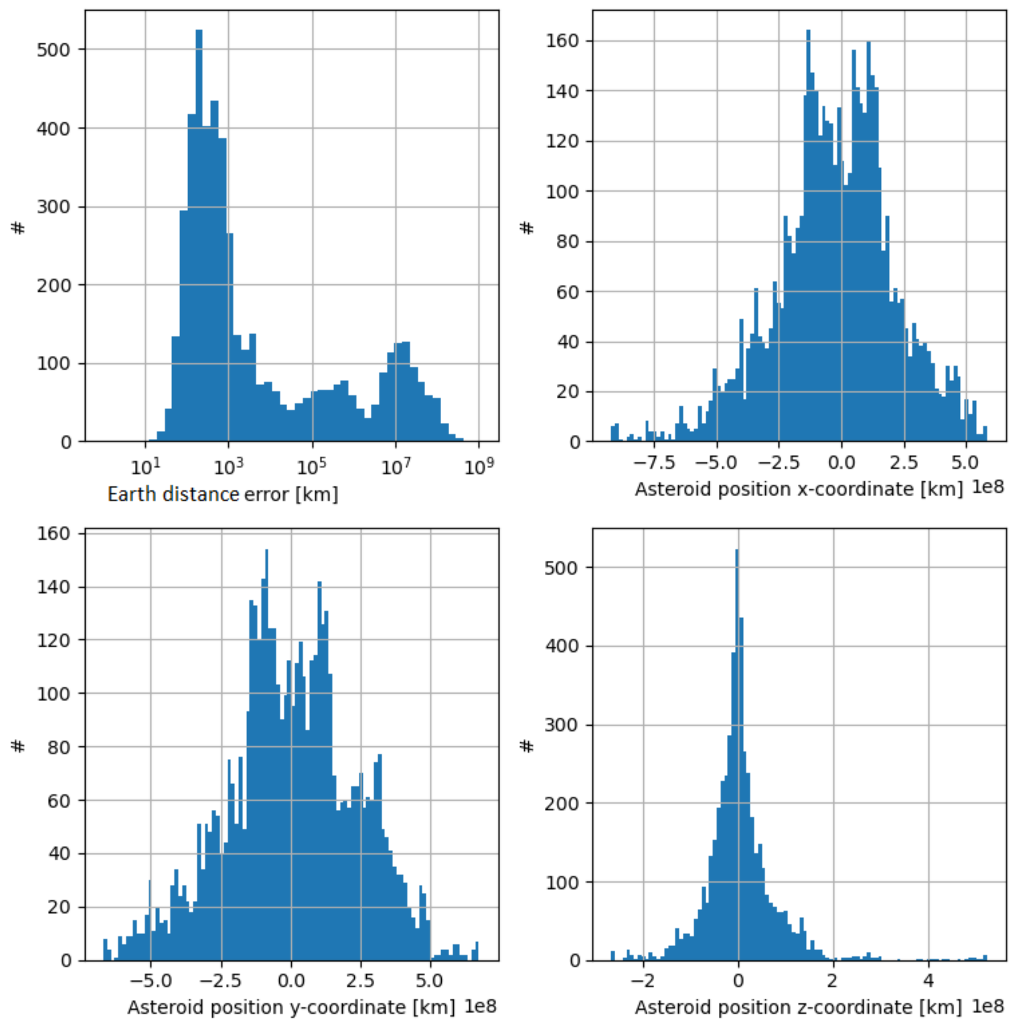


Figure 5.4: Range of error due to uncertainties in km and the range of distances of the asteroid position in km, for all 100 PHAs combined. These values are coming from the propagation model.

The outputs that will be selected for this preliminary investigation are the error due to the uncertainties and the x, y and z-position of the asteroid at a future time instance. This position is then used to estimate the distance to Earth, as the ephemeris of Earth is known with an accuracy of smaller than 100 meters [16]. More details on the prediction and evaluation of the Earth distance can be found in Section 6.1. The input file is generated for the 100 asteroids, so this means that the errors and distances to Earth are computed for all 5,000 data points. These two outputs are plotted to show the range that is covered by these outputs, as shown in Figure 5.4. It should be noted that these values are calculated using the propagation model, no values have been predicted yet using the neural network. From Figure 5.4 it can be seen that the asteroid positions are spread evenly. The distance to Earth is also computed for these values, which will be used to examine the results better and this is also visualised in Figure 5.5. The Earth distance has a mean of $2.9 * 10^8$ km and a 1-σ value of $1.75 * 10^8$ km. The error due to the uncertainties has a mean of $5.73 * 10^6$ km, and a 1-σ value of $2.23 * 10^7$ km. The median is 707.3 km, so 50% of the data has an error smaller than 707.3 km. This shows that the asteroid positions are perfectly fine, but there might be some issues with the error. Whether this is a problem will be discussed later once the results are shown.

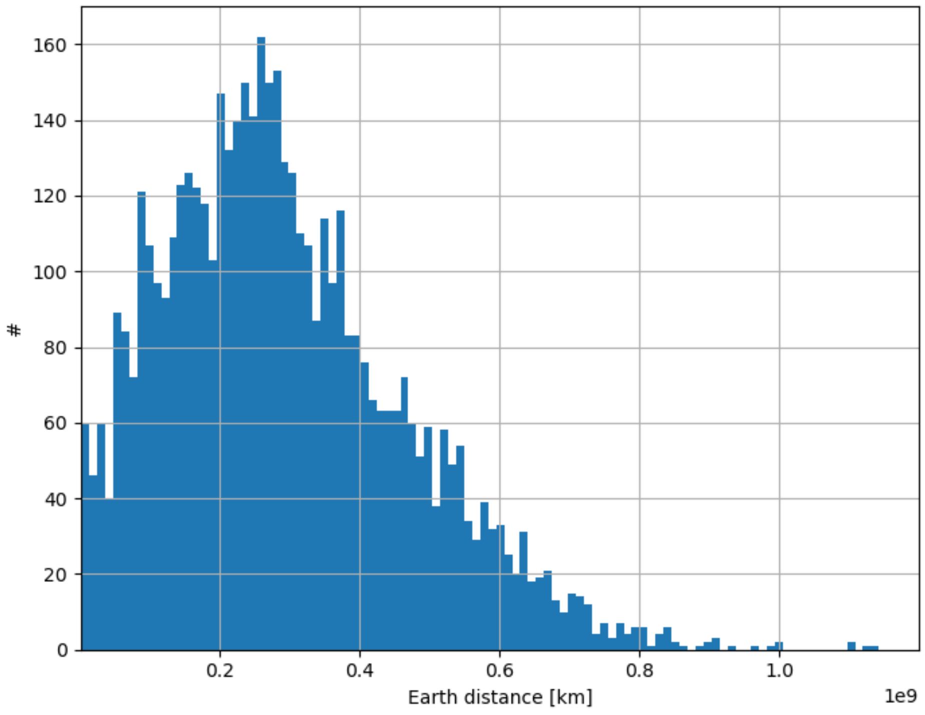


Figure 5.5: Range of Earth distances in km, for all 100 PHAs combined.

5.3.2. Neural networks

It is already known what the input file looks like for machine learning, as it was described in Section 5.1. Machine learning can be used to estimate output variables, based on some input variables by means of a neural network. It is using information where the inputs and outputs are already known for many data points, such that the neural network learns the behaviour of going from the input to the output variables. The neural network is trained in that way, and can be used afterwards to predict outputs for any specified (new) inputs. This method is usually effective when the behaviour is complex, and when the normal computation of the output for a new input case takes quite some time.

The neural network estimates the output variables, but how does it actually work? A neural network structures incoming information as input nodes, or neurons, as they are called officially. Every input has a weight and a bias, such that more importance can be given to one specific neuron, if that information is more reliable than others. An activation function is then used to determine the output. The activation function ensures that the linear combination of weights and biases is transformed such that non-linear behaviour can also be learnt [5]. The architecture of one neuron is shown in Figure 5.6, where x_1 - x_n are the input variables, so in this case x_1 would be the semi-major axis, x_2 would be eccentricity, and so forth. b is the bias and a is the learning rate. y is the value of the neuron in the hidden or output layer and determined by a weighted sum of the values of the input variables, the weights and the biases. More information on the structure and the exact working of the neural network will be described in the next paragraphs.

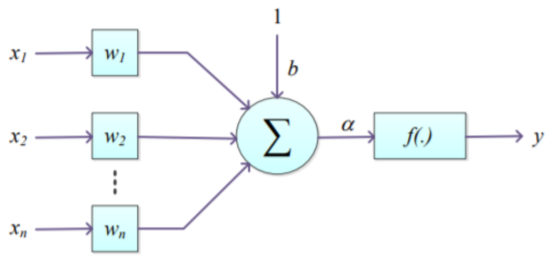


Figure 5.6: A single neuron diagram [43].

The machine-learning algorithm can also have multiple layers, that are connecting neurons between the input neurons and the eventual output. Layers in between the input and the output layer are referred to as hidden layers in a neural network. More neurons per hidden layer and more hidden layers usually means that the neural network has more space to store bits of relevant information. After a certain point adding more neurons does not increase the accuracy, this is usually the optimum, as more neurons also means more computation time. A neural network with one hidden layer is visualized more clearly in Figure 5.7.

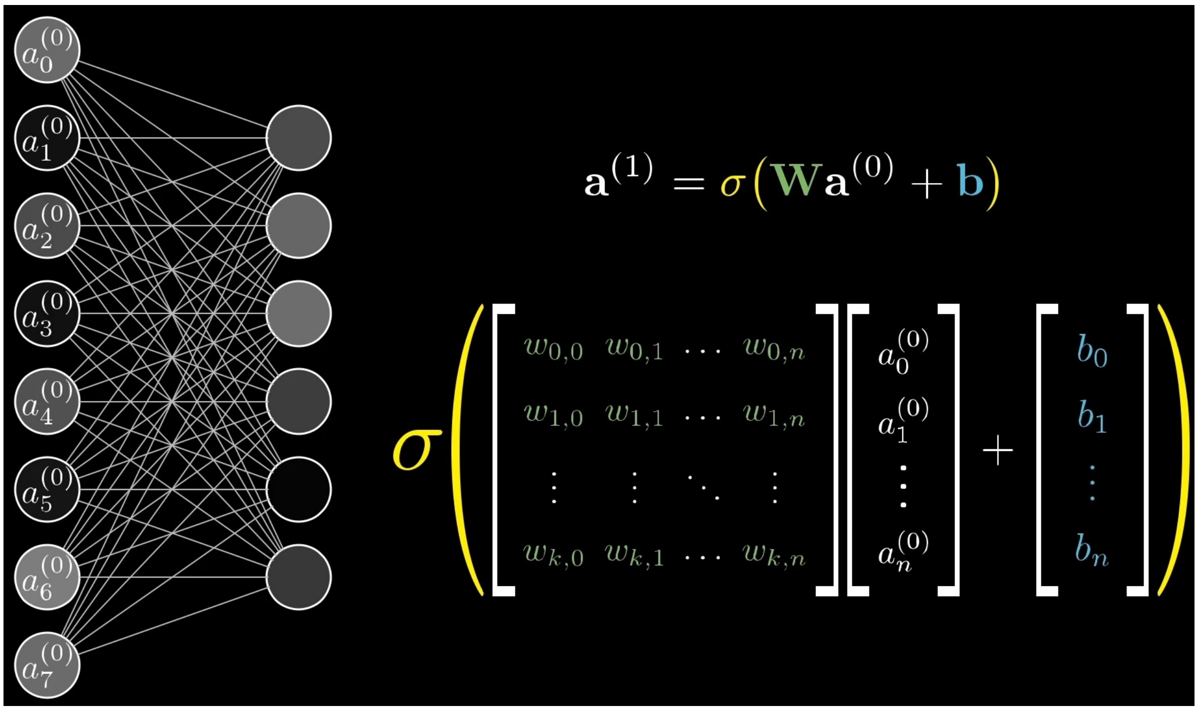

$a_0^{(0)}$
$a_1^{(0)}$
$a_2^{(0)}$
$a_3^{(0)}$
$a_4^{(0)}$
$a_5^{(0)}$
$a_6^{(0)}$
$a_7^{(0)}$
$\mathbf{a}^{(1)} = \sigma(\mathbf{W}\mathbf{a}^{(0)} + \mathbf{b})$
$\sigma\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k,0} & w_{k,1} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} a_0^{(0)} \\ a_1^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}\right)$

Figure 5.7: The connection of the first layer (input layer with 8 inputs) to the second layer (hidden layer with 6 neurons) of neurons using weights w, biases b and an activation function (σ) [1].

The number of weights and biases can also be deduced from the size of the network. The number of neurons in the input and output layer are set by the number of input and output variables, as each neuron represents one variable. The required number of weights per connection between two layers are defined as the number of neurons in the first layer times the number of neurons in the second layer (see the matrix in Figure 5.7). The required number of biases per connection between two layers are defined as the number of neurons in the second layer. Assuming that a network always has the same number of neurons per hidden layer, Equation 5.2 can be used to compute the number of weights and biases of a neural network:

$$\begin{aligned} n_w &= n_i * n_L + (m-1) * {n_L}^2 + n_L * n_o \\ n_b &= n_L * m + n_o \end{aligned} \tag{5.2}$$

where n_w is the number of weights, n_b the number of biases, n_i the number of inputs, n_o the number of outputs, n_L the number of neurons per hidden layer and m the number of hidden layers.

The neural network computes the outputs based on the weights and biases and the activation function by means of an iterative process. The weights are updated each iteration using a solver, which can be seen as a formula. The learning rate of a neural network defines then how large this update is, and the learning rate can also be adjusted during the process [18]. Time can also be saved to split up the input data into batches, which will be trained separately.

All these aspects (number of hidden layers, number of neurons per hidden layer, activation function specifications, initial learning rate and learning rate settings, the solver formula and the batch size) are inputs to the neural-network algorithm and are referred to as hyperparameters in machine learning. The hyperparameters are set before the training of the neural network. It is necessary to tune these hyperparameters such that the most optimal result can be obtained, as the most optimal hyperparameter settings are problem-specific.

The neural network learns using a so-called cost function, which determines how good the neural network is able to estimate the output variables. The objective is to minimize the cost function, and after a certain number of iterations it has reached an optimum, where the neural network is converged and no better results can be found. This optimum depends on the inputs and outputs and the hyperparameters, as mentioned earlier. More details on the exact working of some of these hyperparameters is explained once the different options for each hyperparameter are addressed, in Section 5.4.2.

5.3.3. Hyperparameters

The neural network is constructed using the scikit-learn Python package [38]. This package includes a wide range of different machine-learning functions, and can be divided in two groups, one for regression problems and one for classification problems. The aim of this study is to estimate output values that in principle can take on any value, which means that the problem can be seen as a regression problem. Moreover, a neural network is used, which means that the MLPRegressor() function will be used. This function is first called to build a model, after which the model is fitted and values can be predicted using the model. The regressor function does have many parameters, which are representations of the hyperparameters that were introduced in the previous subsection. The most important hyperparameters that can be set using the regressor function are shown below:

- **Hidden layer sizes:** The number of hidden layers of the neural network and the number of neurons per layer.
- **Activation:** activation function used in the hidden layers of the neural network.
- **Solver:** Solver for weight optimization. This solver defines how the weights are updated in each iteration.
- **Batch size:** size of minibatches for stochastic optimizers.
- **Learning rate:** learning rate schedule for weight updates. Can be constant or changing throughout the learning process.
- **Learning rate init:** initial learning rate which defines how fast the neural network learns.

Other important settings not included in the regressor function are listed below. These settings can be changed manually outside the regressor function.

- **Scaler:** Scaling of the input variables (preprocessing)

- **Training/test data split:** Fraction of training data from the total number of data points
- **Score:** Score that defines quality of neural network

The scaler defines how the input and output variables are scaled before the network is trained. This is necessary since some variables are quantified by large numbers (e.g. a distance in km is in the order of 10^8), whereas other parameters are small (e.g. the eccentricity ranges between 0 and 1). This makes it difficult for the neural network to learn properly. Therefore the data are scaled; the options for this scaling method will be discussed later.

The training/test data split defines the sample size, since the training samples are the only data that are trained by the neural network. The training sample size has a significant influence on the robustness and the accuracy of the model [25]. The sample size should not be too small, otherwise the model is not robust and overfitting occurs, but the sample size should also not be too large, otherwise underfitting might occur, where the model is not able to find causal relations between the input and output variables. More details on under/overfitting can be found in Section 5.3.5. This means that there is an optimal sample size, and thus the sample size needs to be a hyperparameter as well. It should be noted though that more data is usually better, and that the under/overfitting issues can be solved by other settings.

The score defines the quality of the neural network and is used internally to determine whether the model estimation has converged or not. The score type is also known as metrics in machine learning, which is discussed in more detail in Section 5.3.4.

Now that all the important hyperparameters have been mentioned, it is worthwhile looking at similar astrodynamics problems as a starting point. Table 5.1 shows three astrodynamics problems with the used hyperparameters [43] [44] [21].

Table 5.1: Hyperparameters of neural networks used for typical astrodynamics problems.

Hyperparameter \ Application	Hybrid propagation for orbits around Earth [43]	Optimal guidance profile [44]	Real-time optimal control for landing problems [21]
Learning rate	-	0.001 (adaptive)	0.001 (constant)
Sample size	86,401	100,000	13,500,000
Inputs $\rightarrow$ Outputs	1720 $\rightarrow$ 1	13 $\rightarrow$ 3	5-6 $\rightarrow$ 2
Hidden layers	1-2	4	5
Neurons per layer	74	200	32
Activation function	Maxout ($\approx$ ReLU)	ReLU/Tanh	ReLU/Tanh
Solver for weight optimization	SGD	ADAM	SGD
Batch size	-	64	8
Score type	-	MSE	MAE

Based on the values specified in this table the starting point will be selected for the neural network of this research. For the learning rate a value of 0.001 is chosen, since this is the only value that is provided. The required sample size is in all cases at least in the order of 10,000, showing that this is required for a larger number of inputs and outputs. The number of hidden layers varies between 1 and 5, so the number of hidden layers will be set to 3. The number of neurons per layer varies between 32-200, so a number of 100 will be used, which is also the default value in the scikit-learn package. The Rectified Linear Unit (ReLU) will be used as the activation function, as this is used for all the examples. The Stochastic Gradient Descent (SGD) method is taken as the solver for weight optimization, as two of the three examples use that method. The chosen batch size is 50, which is based on the values in the table, but also on the default value set by the scikit-learn function, which is 200.

5.3.4. Metrics

Regarding the score types or more commonly known as metrics, some more explanation is required. The neural network uses one metric throughout the training process to define the performance of the network, and this metric is also used to define overfitting or underfitting (see Section 5.3.5). Scikit-learn uses the Mean Squared Error (MSE) as the default metric. Multiple metrics can then be selected after the fit to see how well the predictions were made. The metrics usually selected for regression-type problems are MSE, the Mean Absolute Error (MAE) and the R^2 score [52]. These scores are calculated using Equations 5.3, 5.4 and 5.5 [39].

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2} \qquad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{5.3}$$

$$\text{MAE}(y,\hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}|y_i-\hat{y}_i| \tag{5.4}$$

$$\text{MSE}(y,\hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i-\hat{y}_i)^2 \tag{5.5}$$

where $\hat{y}$ are the predicted values and y are the true values. n is equal to the number of values to predict. It should be noted that a smaller value for MSE and MAE means a better result, which is opposite to R^2, where a value closer to 1 means a better result. Therefore 1 minus the MSE and MAE values will be considered as the metrics, to be able to compare these directly with R^2.

All three metrics are selected in order to have a more robust result, to avoid any bias by a certain metric. Besides, these scores are capable of identifying different behaviour. MSE is dependent on the unit of the outputs and compared to MAE it generally performs worse when outliers are large. When outliers are not present, MSE usually performs better [46]. These two metrics however only capture the size of the residuals. R^2 measures a percentage that describes how much the neural network is able to capture the variance of the predicted output [23].

5.3.5. Under/overfitting

A neural network does not always find relations between input and output variables, even if they do exist. This is called underfitting, which is visible by the fact that the predictions are unrealistic. Overfitting can also take place, in which the model copies the behaviour of the training data perfectly. However the output is much worse, when performing predictions on the test data, since too much focus lied on the training data set. Under- and overfitting can be prevented by selecting proper values for hyperparameters, such as the sample size as described in Section 5.3.3. Another common way is to define a threshold when the network needs to stop training. A fraction of the training data is not used for training, but is used to validate the performance throughout the learning process of the neural network. The score is evaluated every iteration. If the score does not improve after a few iterations, the process will stop. The process will also stop once the training data score is not improving anymore, which is how the network decides where to stop using the default settings. This threshold is known as the early-stopping parameter, which will be used for all tests performed in this study.

5.3.6. Preliminary results

The training/test data split is taken as 0.8, meaning that 4,000 data points are used for training and 1,000 for testing. 4,000 is not much compared to the examples, so it should be kept in mind that more data (using all the asteroids) could potentially provide better results. The settings that were defined in the previous sections are all summarized in the list below:

- **Hidden layer sizes:** 3 x 64
- **Activation function:** ReLU
- **Solver:** SGD
- **Batch size:** 50
- **Learning rate init:** 0.001
- **Learning rate:** adaptive
- **Scaler:** MinMaxScaler (normalization)
- **Training/test data split:** 0.8
- **Score:** MSE
- **Early stopping:** True

First two networks will be trained using these hyperparameters, one using only eight inputs (removing the uncertainties in the state (6, uncorrelated) and the uncertainty in the PHA radius) and three outputs (xyz position of the asteroid). This is compared with the case where all inputs are used together with the same output. The results are shown in Figures 5.8 and 5.9.

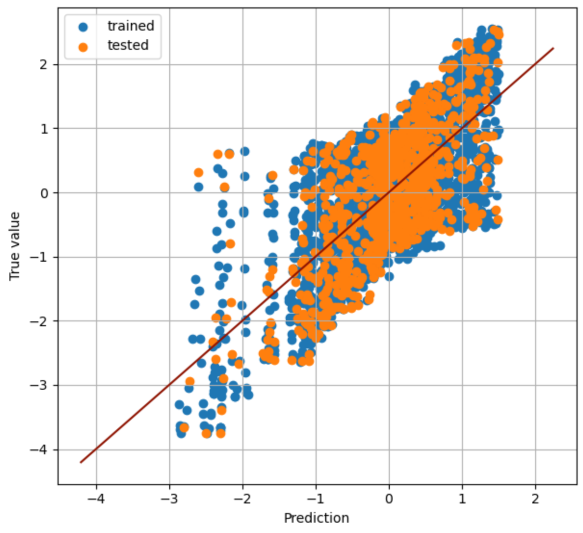


Figure 5.8: Predicted vs true values of the x,y and z position of the asteroid using 8 inputs. 1,000 data points out of the total 5,000 were taken as test points.

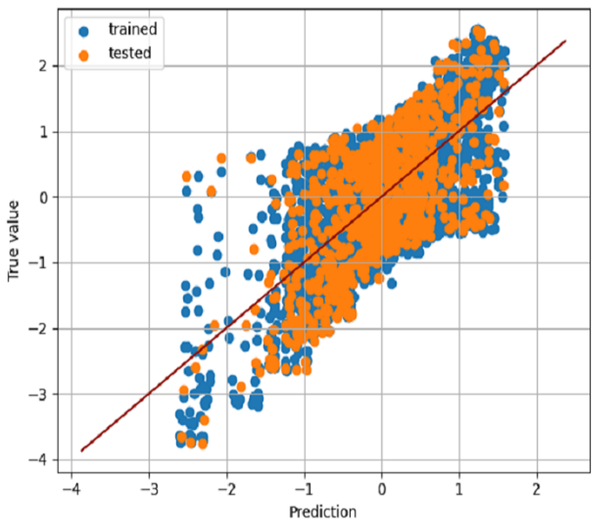


Figure 5.9: Predicted vs true values of the x,y and z position of the asteroid using 15 inputs. 1,000 data points out of the total 5,000 were taken as test points.

The figures show the three outputs in one figure. The differences shown in the figures are normalized and Equation 5.6 can be used to transform the normalized values back to a distance, which also shows the derivation.

$$
\begin{aligned}
y_i' &= \frac{y_i - \mu}{\sigma} \\
\hat{y}_i' - y_i' &= \frac{\hat{y}_i - \mu}{\sigma} - \frac{y_i - \mu}{\sigma} = \frac{\hat{y}_i - y_i}{\sigma}
\end{aligned}
\tag{5.6}
$$

where $\hat{y}$ are the predicted values and y are the true values. μ is the mean and σ is the standard deviation. The standard deviation of the average of the three outputs is $5.86 * 10^7$ km, so a difference of 1 in the normalized case means a difference of $5.86 * 10^7$ km in individual coordinate component (0.4 AU). This is already large and also compared to the mean value ($4.8 * 10^7$ km or 0.32 AU) this means a difference of > 100%, which means that these results are definitely not good enough. This is also concluded from the scores that were given to these results, which are summarized in Table 5.2.

Table 5.2: MSE, MAE en R^2 scores for the specified inputs and outputs.

Inputs/outputs	8 $\rightarrow$ 3 ($xyz_{asteroid}$)	15 $\rightarrow$ 3 ($xyz_{asteroid}$)	8 $\rightarrow$ 1 (ϵ)	15 $\rightarrow$ 1 (ϵ)
R^2 score	0.455	0.454	0.688	0.779
MAE score	0.382	0.381	0.801	0.855
MSE score	0.456	0.454	0.698	0.780

The table also shows scores for the second output, the error caused by the uncertainties in the state and the dynamics model (ϵ), for which two extra networks were trained with the same inputs for comparison and one output instead of three. This is also visually shown in Figures 5.10 and 5.11. A few conclusions can be drawn from these results. The number of inputs is not relevant for the prediction of xyz_{earth}, whereas this does make a difference for the prediction of ϵ. More inputs result in a better model, which shows the importance of these inputs, and also might mean that more data are required to obtain better results, as more input variables usually require more data. It was also expected that more inputs result in a better model for the prediction of ϵ, since there is information regarding uncertainties in the input data. Another observation is that the prediction of xyz_{earth} is significantly worse compared to the prediction of the distance error. This will be investigated in the next section in more detail.

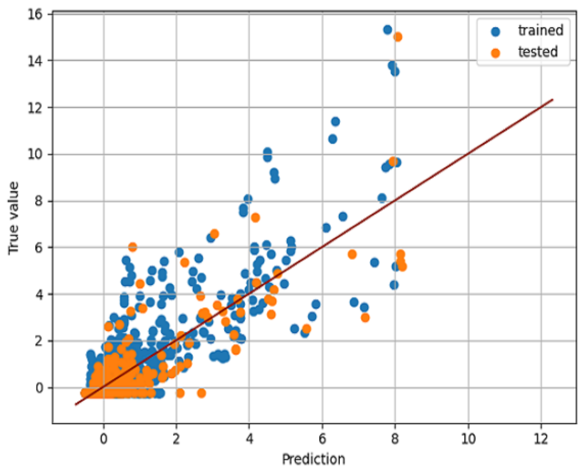


Figure 5.10: Predicted vs true values of the error due to the uncertainties using 8 inputs. 1,000 data points out of the total 5,000 were taken as test points.

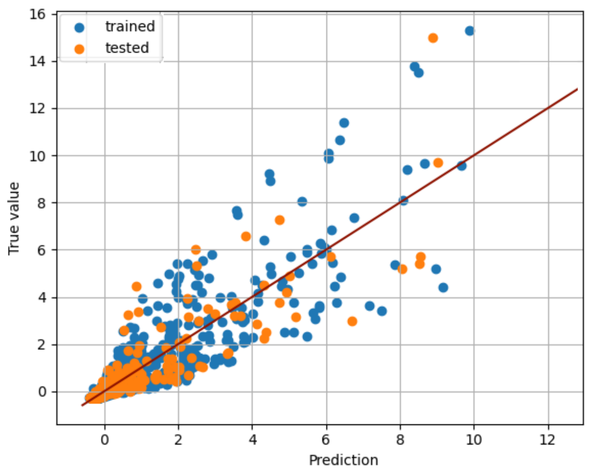


Figure 5.11: Predicted vs true values of the error due to the uncertainties using all 15 inputs. 1,000 data points out of the total 5,000 were taken as test points.

5.4. Neural network tuning

The results so far were obtained for a given set of hyperparameters, i.e. without any tuning, and with a limited sample of input data. All the 2241 PHAs will now be used to train a neural network, to be able to see the effect of the full data set. This also includes the tuning process, which is aimed at finding the most optimal hyperparameters. The hyperparameters were initially selected based on similar astrodynamics problems, but the optimal values are different for every problem. Therefore a selection of hyperparameters will first be made, after which the tuning process will be discussed and the results will be provided.

5.4.1. Input file generation

Tuning depends on many factors and one of them is the number of data points; this determines how large the neural network should be (number of neurons per layer and number of layers). The number of data points is completely different comparing the small data set and the full data set, therefore the tuning will only be performed on the full data set. The input file is thus generated for all the 2241 PHAs, with 50 data points per asteroid, meaning that the data file contains 112,050 data points.

Besides this data file, a number of inputs and outputs were selected for the neural network. This is slightly different from the initial investigation in the previous section, as the results shown there were not perfect. This can have many reasons, and is also difficult to judge when no tuning is applied. However, a number of cases will be listed here that will be investigated for the full data set. The inputs and outputs as described in the previous section will be chosen as the first test case. The second test case uses Cartesian elements instead of Kepler elements. This could potentially provide better results, as Cartesian elements usually do have a larger variety of values, which makes it possible to capture the behaviour of PHAs in a more generalised form. It is however expected that Kepler elements perform better, as Kepler elements provide more fundamental information on the orbit type. Next to these options, three inputs can be added to provide information about the position of Earth. This can then be used to change the three outputs regarding the position of the asteroid to one output, which is the distance between the asteroid and Earth. This means less outputs and a potential better result.

Other minor changes to these test cases will also be evaluated, but this will only be done for the best test case, to save time. This will be addressed in Chapter 6. The main test cases are listed in Table 5.3.

Table 5.3: Possible cases for input and output variables in order to investigate if better results can be found for the neural network.

Cases	# Inputs	# Outputs
1. Using preliminary selected variables	15 (Kepler + Δ, R, ΔR, t_p)	4 (ϵ, $[x, y, z]_{asteroid}$)
2. Using Cartesian instead of Kepler elements	15 ($\mathbf{x}$, $\Delta\mathbf{x}$, R, ΔR, t_p)	4 (ϵ, $[x, y, z]_{asteroid}$)
3. Using more inputs and less outputs	18 (Kepler + Δ, R, ΔR, t_p, $\mathbf{x}_E$)	2 (d_E, ϵ)

5.4.2. Hyperparameter selection

The most important hyperparameters are the sample size and the learning rate [25] [18] and will therefore definitely be selected for tuning. The number of hidden layers and the number of neurons per layer will also be selected, as those could also have a significant impact, and are selected more often [43]. The ranges of those hyperparameters will be based on the data obtained from literature (see Section 5.4.1).

The options for solvers for weight optimization are ADAM, Stochastic Gradient Descent (SGD) and Limited-memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS). ADAM and SGD are different types of gradient-descent methods. A gradient-descent method computes the gradient of a function, used to update the current guess of the parameters. If this considers one variable, this can be seen as finding the minimum of a function, as shown in Figure 5.12.

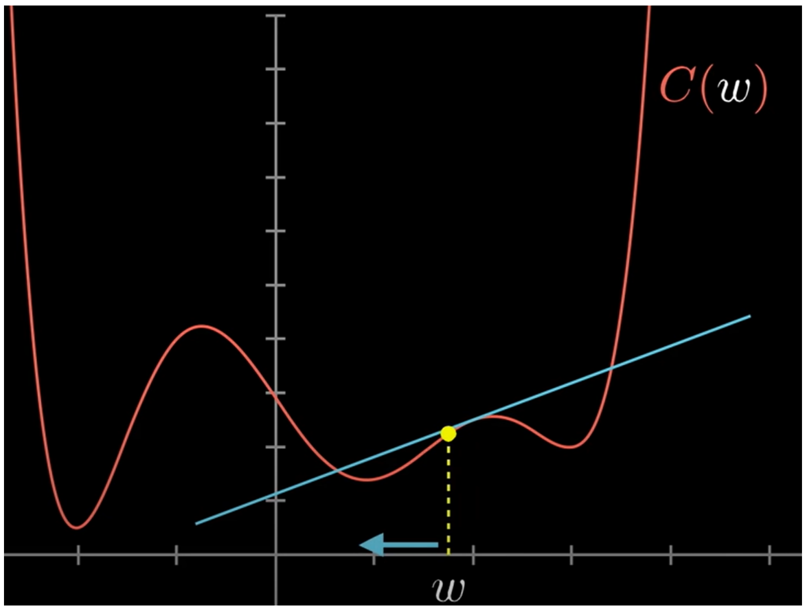


Figure 5.12: Gradient-descent method [2].

First a guess on the function is made, after which the direction is determined; a positive slope means that the optimum value is smaller than the guess value, and a negative slope means that the optimum value is larger than the guess. The rate at which this value is changed is determined by the learning rate, which is another hyperparameter. The drawback is that it can happen that a local optimum is found instead of the global optimum. In case of a neural network there are many values which can be changed throughout the iterative process of finding the most optimal values, and these values are defined by the number of weights and biases. For example, a network with three hidden layers and 64 neurons per hidden layer, as was used in the initial investigation results in a total of 9,604 weights and biases, using Equation 5.2. This number is thus also equal to the number of values that can be changed during the iterative process. The gradient-descent method SGD can be formulated as a formula: [37]

$$\theta_{\mathsf{i}} = \theta_{\mathsf{i-1}} - \alpha \frac{\partial L}{\partial \theta_i} \tag{5.7}$$

where θ_i is the vector of the ith iteration value of all variables. L is a function having a number of variables equal to the number of weights and biases. α is the learning rate and the gradient is thus multiplied by the learning rate to update the value of the variables.

Slightly more complicated is ADAM, which takes into account momentum. ADAM is formulated as shown in Equation 5.8 [37].

$$\begin{aligned} m_i &= \beta_1 m_{i-1} + (1-\beta_1)\frac{\partial L}{\partial \theta_{i-1}} \\ v_i &= \beta_2 v_{i-1} + (1-\beta_1)\left(\frac{\partial L}{\partial \theta_{i-1}}\right)^2 \\ \widehat{m_l} &= \frac{m_i}{1-\beta_1}, \quad \widehat{v_l} = \frac{v_i}{1-\beta_2} \\ \theta_i &= \theta_{i-1} - \frac{\alpha}{\sqrt{\widehat{v_l}}+\epsilon}\widehat{m_l} \end{aligned} \tag{5.8}$$

where β_1 and β_2 are the exponential decay rates for the momentum, which should be between 0 and 1 and are using the default settings set to 0.9. m and v are the momentum vectors, and ϵ is the value for numerical stability which is taken as a constant, set to 10^{-8}.

Based on theory it can be concluded that SGD usually generalizes better, which means that overfitting is less likely to happen. ADAM on the other hand converges faster and therefore it is interesting to test these methods [37]. The method not explained yet is LBFGS. LBFGS is a quasi-Newton method that uses the curvature of the parameter space [29]. If there is not much data available, the parameter space does not have many curvatures in it, and this makes LBFGS a very strong method. However in this particular case data are generated and it is therefore likely that SGD or ADAM performs better, as the data are not sparse. Therefore it is decided to only test ADAM and SGD for this research.

The options for activation functions are limited to four in scikit-learn, which are shown in Equation 5.9 [38] [5].

$$\begin{aligned} {'identity'} &\rightarrow f(x) = x \\ {'logistic'} &\rightarrow f(x) = \frac{1}{1+e^{-x}} \\ {'tanh'} &\rightarrow f(x) = tanh(x) \\ {'ReLU'} &\rightarrow f(x) = max(0,x) \end{aligned} \tag{5.9}$$

The activation function is a linear combination of the weights and biases (see Figure 5.7). An identity matrix thus results in a linear relation between the inputs and outputs and this means that non-linear behaviour can not be modelled and that the number of layers does not have any effect, as the last layer stays a linear combination of the weights and biases of the first layer [5]. Since the problem does have non-linear behaviour the identity option will be removed, and the other options will be used in the tuning process. It should be noted that ReLU usually performs the best out of these activation functions and is therefore used in most problems (see also Table 5.1, where all methods used ReLU as the activation function for the hidden layers).

Many options are available for scaling parameters in scikit-learn. As there is not enough time to test all the scaling options a selection will be made. The most common scaling options are normalization and standardization. Normalization scales the values of the variables to a range of [0, 1]. This is done by using Equation 5.10 [24].

$$x^{(i)}_{norm} = \frac{x^{(i)} - x_{\mathsf{min}}}{x_{\mathsf{max}} - x_{\mathsf{min}}} \tag{5.10}$$

This formula is known in scikit-learn as the StandardScaler (SS). Standardization wraps the data to a normal distribution, with a mean of 0 and a standard deviation of 1. This is formulated as shown in

Equation 5.11. [24]

$$x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x} \tag{5.11}$$

This formula is known in scikit-learn as the MinMax Scaler (MMS). To also include a scaler that takes into account non-linearity and therefore completely different from both standardization and normalization, a QuantileTransformer (QT) is selected as the third and final scaler [38]. It maps the data to a uniform distribution in a range of [0,1].

The batch size usually determines the robustness versus the computational speed of the training of the neural network. The influence of the batch size is not so significant and is therefore investigated after the tuning process separately [22]. The batch size is set to 50, the same as was used for the initial investigation.

The selection of hyperparameters is summarized in Table 5.4.

Table 5.4: Hyperparameter selection.

Sample size	[5000, 25000, 45000, 65000, 90000]
Hidden layers	[1, 2, 3, 4, 5]
Hidden neurons	[4, 8, 16, 32, 64, 128, 256]
Learning rate	[0.01, 0.005, 0.001, 0.0005, 0.0001]
Learning rate type	[constant, adaptive, invscaling]
Activation function	[ReLU, Tanh, logistic]
Solver for weight optimization	[ADAM, SGD]
Scaler	[MinMaxScaler, StandardScaler, QuantileTransformer]

The number of layers and the number of neurons per layer are mainly determined by the number of data points, as already mentioned in the previous subsection. It is therefore decided to first tune the network based on these two variables, after which the rest will be tuned.

Testing all the different combinations in two phases would result in a total of $(5*5*7)+(5*3*3*2*3) = 445$ options. The total computation time for one case (selection of input and output variables) would then be 2.5 hours, assuming that a similar run time per setting is used as for the 100 asteroids data set (20 s). The computational speed is also the reason why selections were made on how and which hyperparameters to take into account.

The different cases regarding the input and output variables that were mentioned in the previous section will be tested separately using the procedure of testing all the different combinations in two phases.

5.4.3. Results - Case 1

Case 1 listed in Table 5.3 was tested first for the number of data points and the size of the neural network. ReLu as the activation function, SGD as the solver, an adaptive learning rate with an initial value of 0.001 and a Standard Scaler were used. The relation between the size of the neural network and the score was found to be similar for every number of data points. The result of a sample size of 10,000 can be found in Figures 5.13 and 5.14. Figures of other sample sizes are listed in Appendix D.

The results obtained with 10,000 data points show that more neurons per layer means a better result generally, and at some point it seems to converge to an optimum. The number of layers does not seem to have any effect, and the reason for that could be that the relation between the inputs and outputs can be covered by only one relation type. A relation type is usually covered by a different layer, for example a linear component or a quadratic component. The maximum scores for this calculation, and also for the larger selected sample sizes are summarized in Table 5.5. Only the R^2 scores are listed, which will also be done for all the upcoming comparisons, to be able to clearly see the differences and the fact that R^2 is independent on the scaler, which is a hyperparameter that will be varied in phase 2.

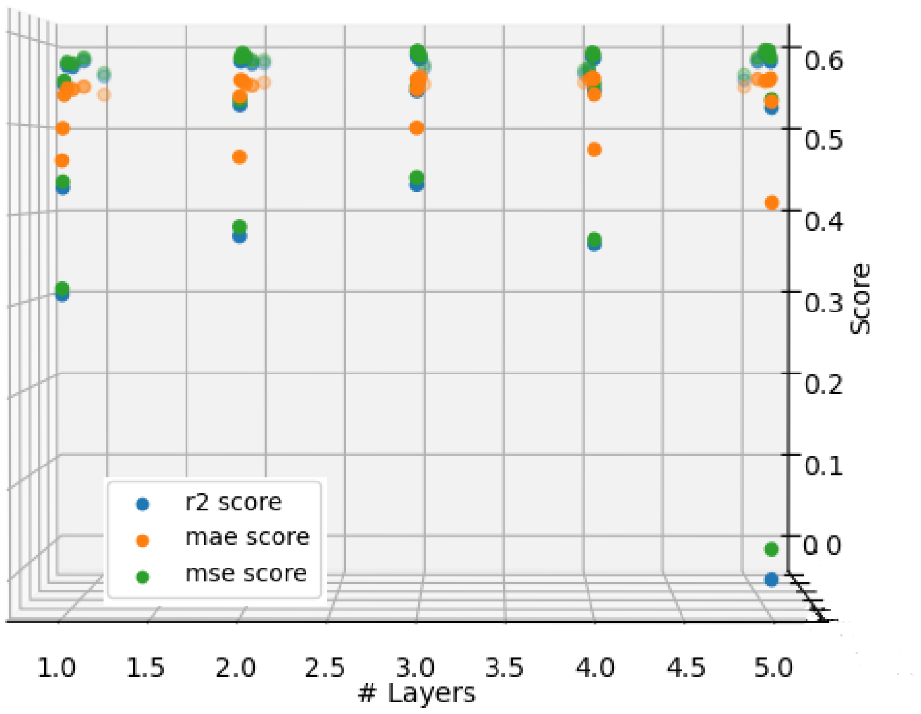


Figure 5.13: Number of layers for a sample size of 10,000 - Case 1.

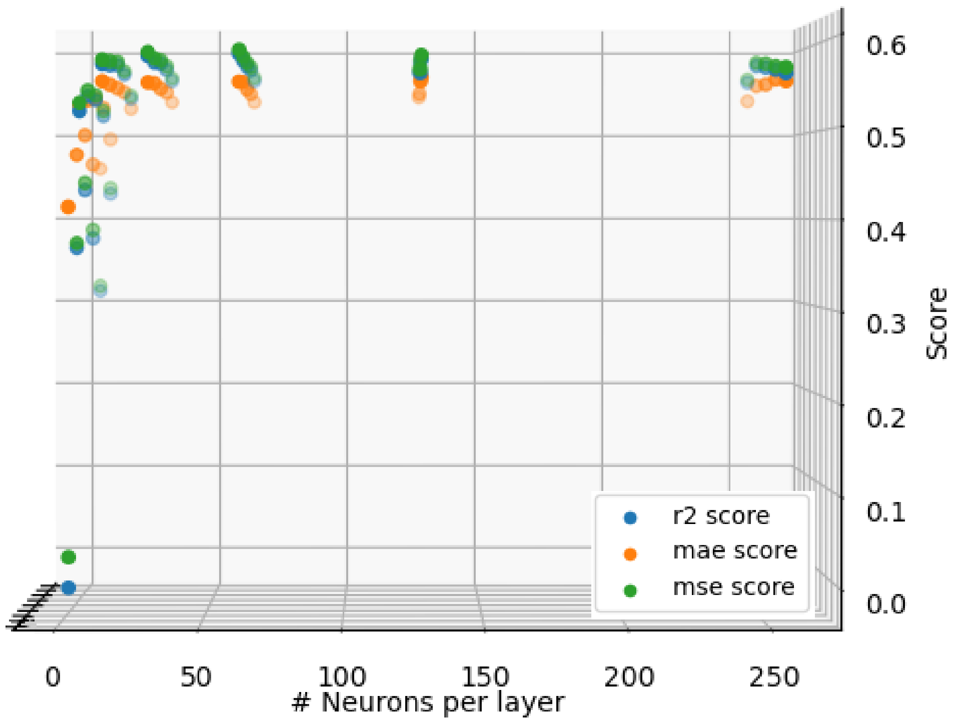


Figure 5.14: Number of neurons per layer for a sample size of 10,000 - Case 1.

Table 5.5: Optimal and reasonable scores of case 1, phase 1.

	optimal			*reasonable*		
# samples	R^2	time [s]	hidden layer size	R^2 score	time [s]	hidden layer size
10,000	0.570	60	5x256	0.560	15	2x64
25,000	0.596	221	5x256	0.570	30.2	1x32
45,000	0.593	88	2x128	0.585	29	1x128
65,000	0.595	46	2x64	0.587	26.2	1x128
90,000	0.610	2200	5x256	0.600	30	1x128

From Table 5.5 it can be concluded that the R^2 values are not attractive, as the preliminary results already showed larger values (see Table 5.2). It can also be concluded that the score does not improve much for a larger sample size. It is however always better and therefore only a sample size of 90,000 will be used in the upcoming cases, unless specified otherwise. The table also shows reasonable scores and shows similar scores and due to time constraints it is decided to pick a solution from the reasonable options to use as a comparison for phase 2. The size of the network that will be used for phase 2 will be chosen as 1 layer with 128 neurons, using a sample size of 45,000. 90,000 is not yet chosen here, due to time constraints, as there are still many options considered, and to prevent any unforeseen scenarios, as more data usually requires more time, and this might become only visible for a different set of hyperparameters. The sample size will change for the upcoming cases, but not yet for this case.

Phase 2 unfortunately failed to run for the case where the learning rate was too large. It resulted in infinity values and caused a crash. Therefore the learning rate of 0.01 was removed from the hyperparameter options. More variables are varied in phase 2 and therefore a side view is shown for every variable, showing the variation of all combinations between that variable and the corresponding score. This is shown in Figures 5.15 - 5.18.

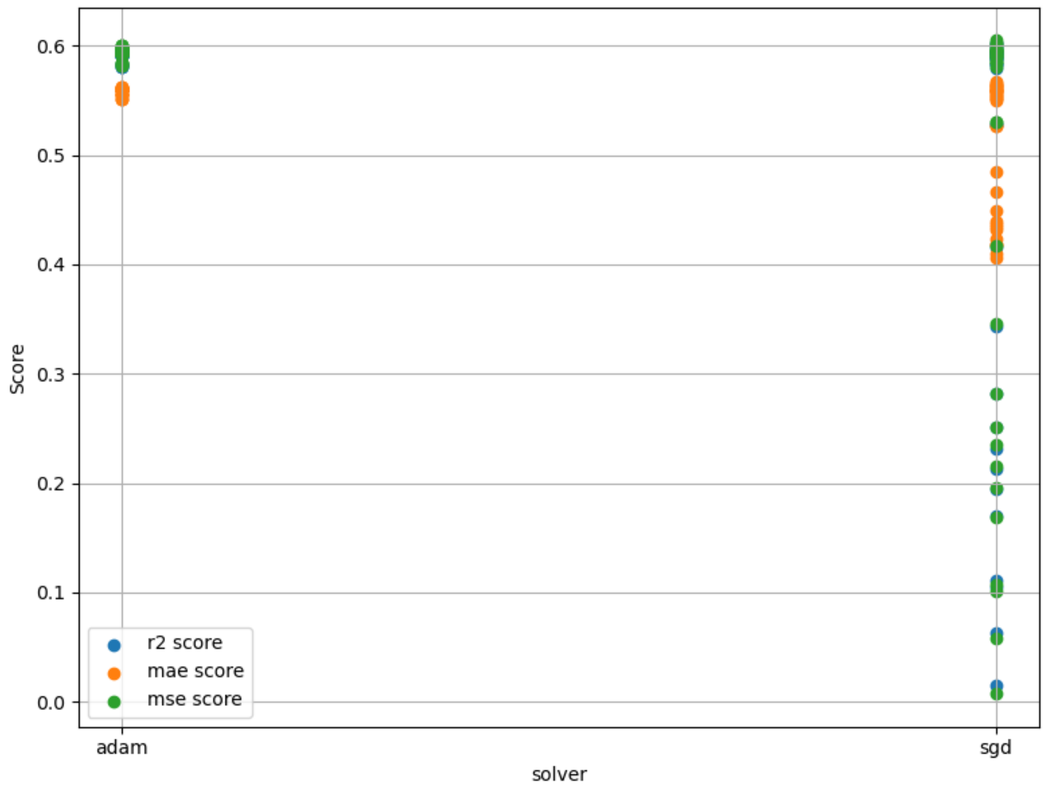


Figure 5.15: Scores for different solvers using the StandardScaler for case 1.

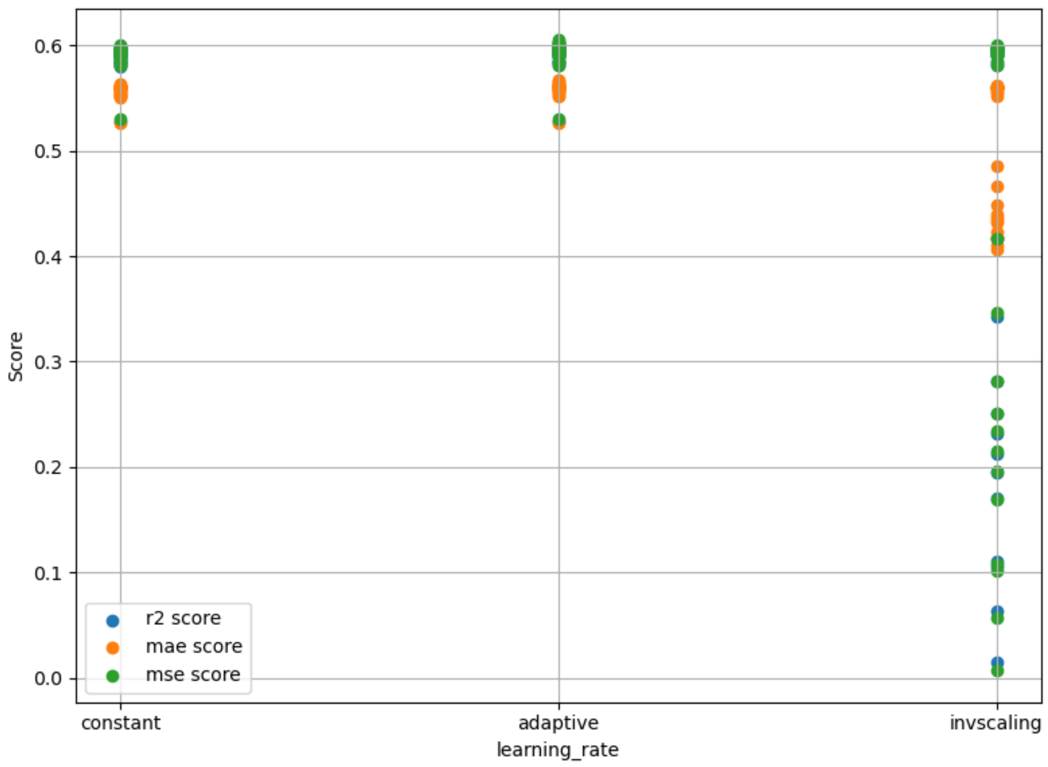


Figure 5.16: Scores for different learning rate types using the StandardScaler for case 1.

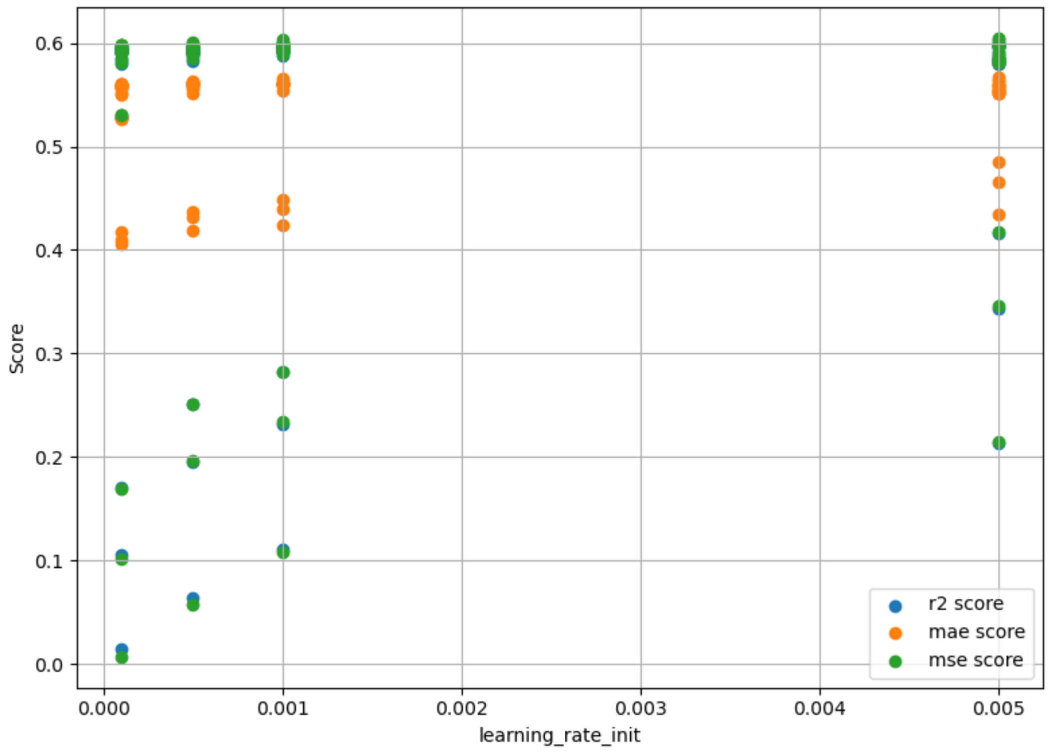


Figure 5.17: Scores for different learning rates using the StandardScaler for case 1.

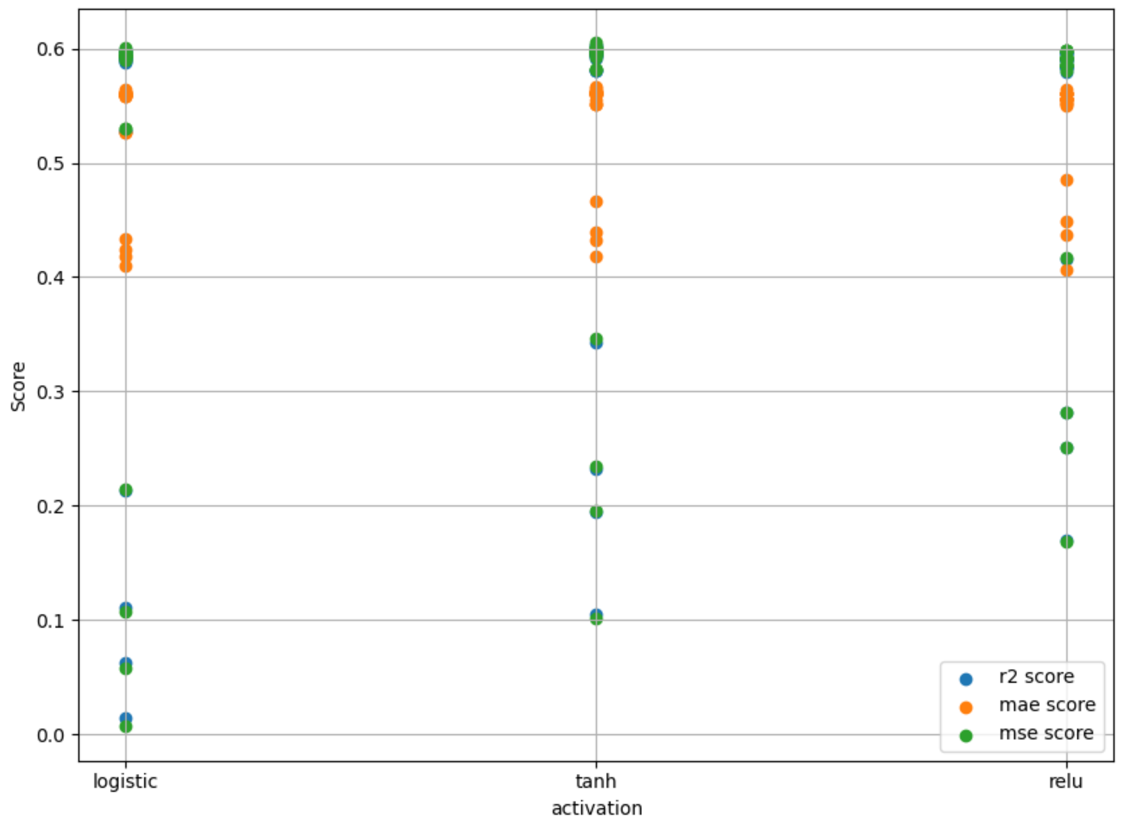


Figure 5.18: Scores for different activation functions using the StandardScaler for case 1.

Figures 5.15 - 5.18 show the variation of scores for four different hyperparameters. Each figure shows all the combinations of the hyperparameters, such that a possible relation can be observed between a hyperparameter and the scores. It should be noted that these figures only include scores that used the SS; figures for the other scalers were created separately as the scaler is one of the preprocessing steps making it more difficult to plot the results in one figure. The results of the MMS and the QT can be found in Appendix D. These figures show a much higher score for MAE and MSE generally (up to 0.998 compared to 0.6 for the SS). However this is due to the fact that MSE and MAE are biased by the scaling method. The SS scales the variables to a normal distribution as mentioned before in Section 5.4.2, which is different to MMS, which scales it to a range of [0,1]. MAE and MSE are dependent on the magnitude of these numbers, and thus the values for MAE and MSE seem to be better for QT and MMS, but this is not true. The R^2 score is however independent on the magnitude, and can therefore be used to properly distinguish the performance of different scalers. The MSE and MAE values can still be used to compare scores for the same scaler.

From the results it can be concluded that unfortunately not a score higher than 0.6 can be found for R^2, also taking into account the other scalers. When comparing R^2 scores between the different scalers, it can be concluded that QT performs worse compared to SS and MMS (0.5 w.r.t. 0.6). When looking at hyperparameters, it shows that all the combinations using SGD as the solver and 'invscaling' as the learning rate type perform much worse compared to other combinations.

Phase 2 was tested for a neural network of 128 neurons per layer and one hidden layer. To increase the probability of finding the global optimum, the best result of phase 2 is tested again for a variety of different layers and different neurons per layer. This will be tested for a sample size of 45,000 to be able to directly compare the different scores. As the scores in phase 2 were very similar to each other, most of the hyperparameters will be kept the same. Only one hyperparameter will be changed: the solver changes from SGD to ADAM. The results can be found in Figures 5.19 and 5.20.

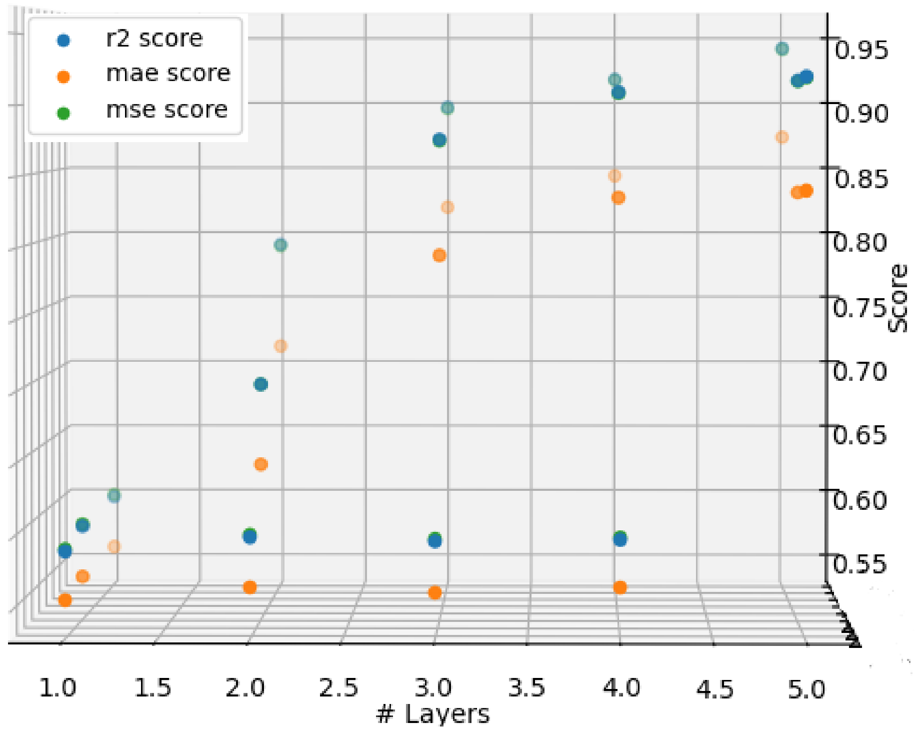


Figure 5.19: Number of layers for case 1, using the same hyperparameters as phase 1, but now with ADAM instead of SGD as the solver.

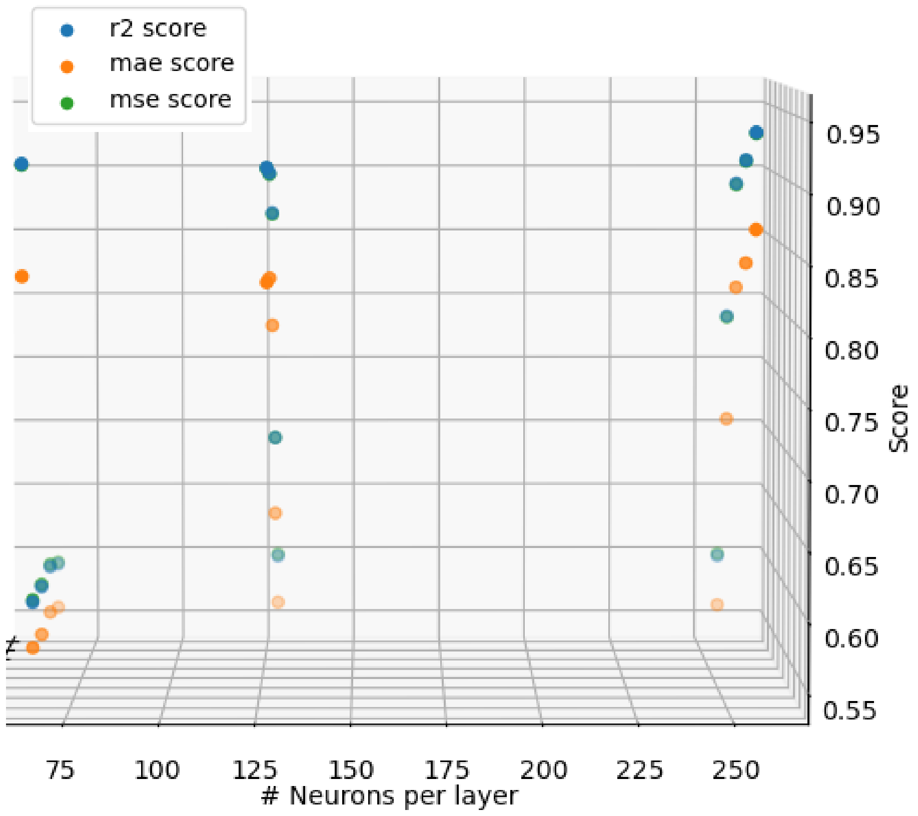


Figure 5.20: Number of neurons per layer for case 1, using the same hyperparameters as phase 1, but now with ADAM instead of SGD as the solver.

From the figures it can be concluded that ADAM performs much better compared to SGD, as R^2 scores of almost 0.95 are obtained. As SGD performs worse, the nominal case will be changed for the upcoming cases with ADAM replacing SGD. All the changes that will be made are summarized below:

- A sample size of 90,000 is used as the nominal case, and other options are discarded
- Scaler QT will be removed
- Learning rate type 'invscaling' will be removed
- Learning rate of 0.01 will be removed
- Solver ADAM replaces SGD in the nominal case, SGD is still part of the tuning process of case 2 and 3

Removing options will decrease the run time, which makes it possible to test for example a case in phase 2 with more neurons and layers, but this will be investigated during the process of the upcoming cases. The three best options of case 1 are shown in Table 5.6.

Table 5.6: Hyperparameters and scores of best options for case 1.

Hyperparameter \ Options	A	B	C
Learning rate	0.001 (adaptive)	0.001 (adaptive)	0.001 (adaptive)
Sample size	45,000	45,000	45,000
Inputs $\rightarrow$ Outputs	16 $\rightarrow$ 4	16 $\rightarrow$ 4	16 $\rightarrow$ 4
Hidden layers	5	5	5
Neurons per layer	64	128	256
Activation function	ReLU	ReLU	ReLU
Solver for weight optimization	ADAM	ADAM	ADAM
R^2 score	**0.92**	**0.92**	**0.94**

5.4.4. Results - Case 2

Case 2 uses Cartesian states instead of Kepler elements (see Table 5.3). The number of layers and the number of neurons per layer are tested first and the other hyperparameters are equal to case 1, with ADAM replacing SGD as the solver for weight optimization. The result of phase 1 is summarized in Table 5.7. The figures were also made for this test case, and can be found in Appendix D.

Table 5.7: R^2 scores for case 2 using 90,000 data points for various numbers of neurons per layer and layers.

Neurons per layer	1 layer	2 layers	3 layers	4 layers	5 layers
4	0.38	0.38	0.40	0.25	0.00
8	0.39	0.40	0.42	0.43	0.44
16	0.41	0.45	0.51	0.53	0.55
32	0.48	0.51	0.53	0.55	0.58
64	0.55	0.55	0.60	0.65	0.70
128	0.58	0.62	0.67	0.74	0.75
256	0.60	0.64	0.70	0.77	0.79

Table 5.7 shows one option with a score of 0.00; the results of this calculation are so bad compared to other calculations that the score is not even greater than zero. This is also coming back later for other cases and will be addressed if and where necessary. For now the only conclusion that can be drawn is that too few neurons per layer is not an attractive option. Table 5.7 also shows an improving score for more neurons per layer and for more layers. Using a smaller number of data points resulted in smaller R^2 scores and are therefore not shown here. The optimal R^2 scores are however smaller than the R^2 scores found at the end of phase 1, which means that case 1 is performing better. 5 layers and 64 neurons per layer are selected for phase 2, since it is a realistic and reasonable option. 256 neurons per layer is much more time-consuming and is therefore not selected for phase 2, as for phase 2 it is only required to find the best hyperparameters, which is assumed to be obtainable with 64 neurons as well. The results are shown in Table 5.8.

Table 5.8: R^2 scores for case 2 using **SS** and 90,000 data points, 5 layers and 64 neurons per layer for various activation functions, learning rates and solvers.

Activation function	Initial learning rate	Solver	R^2 Score
Logistic	0.001	ADAM	0.62
Logistic	0.001	SGD	0.00
Logistic	0.0005	ADAM	0.61
Logistic	0.0005	SGD	0.00
Logistic	0.0001	ADAM	0.59
Logistic	0.0001	SGD	0.00
Logistic	0.005	ADAM	0.58
Logistic	0.005	SGD	0.00
Tanh	0.001	ADAM	0.59
Tanh	0.001	SGD	0.60
Tanh	0.0005	ADAM	0.59
Tanh	0.0005	SGD	0.60
Tanh	0.0001	ADAM	0.58
Tanh	0.0001	SGD	0.58
Tanh	0.005	ADAM	0.49
Tanh	0.005	SGD	0.58
ReLU	0.001	ADAM	0.70
ReLU	0.001	SGD	0.62
ReLU	0.0005	ADAM	0.67
ReLU	0.0005	SGD	0.61
ReLU	0.0001	ADAM	0.63
ReLU	0.0001	SGD	0.59
ReLU	0.005	ADAM	0.62
ReLU	0.005	SGD	0.55

Table 5.9: R^2 scores for case 2 using **MMS** and 90,000 data points, 5 layers and 64 neurons per layer for various activation functions, learning rates and solvers.

Activation function	Initial learning rate	Solver	R^2 Score
Logistic	0.001	ADAM	0.00
Logistic	0.001	SGD	0.00
Tanh	0.001	ADAM	0.43
Tanh	0.001	SGD	0.43
ReLU	0.001	ADAM	0.62
ReLU	0.001	SGD	0.50

The tables do not show the learning rate type, as the scores for SS were almost identical and were also for this reason not tested for MMS. MMS was also not tested for the learning rate, as the learning rate for SS was not showing a large difference in score and it saved time.

The results of case 1 did not show a large variation of scores, but these results (case 2) do show some differences. An initial learning rate of 0.001 seems to be the best option. The limitations of the learning rate type can be explained by the early stopping feature, which stops the iterations if no further improvement can be made in the validation score (see Section 5.3.3). This causes the adaptive learning rate to be more or less equal to a constant learning rate, and this also follows from the results. Regarding the activation functions, ReLU seems to perform the best and the logistic activation function seems to be not very robust. This can also be explained by the fact that these options use SGD as the solver for weight optimization, as SGD seems also not very robust, also taking the results of case 1 into account. Regarding scalers, MMS performed worse compared to SS.

To summarize, a learning rate of 0.001 and using an adaptive learning rate type seems to be the best. The expectation is also that it does not influence the results that much. Regarding activation functions, no conclusion is drawn yet, as logistic turned out to be sometimes better than ReLU, and this also holds for SGD with respect to ADAM for the choice of the solver. SS outperformed MMS for this test case, but it will still be varied for test case 3 to see if any different conclusions can be drawn.

5.4.5. Results - Case 3

Case 3 uses the position and velocity of Earth as inputs and uses two outputs instead of four (see Table 5.3). The PHA position is replaced by the minimum distance to Earth. The other hyperparameters are equal to case 1, with ADAM replacing SGD as the solver for weight optimization. All the results of test case 3 can be found in Appendix D, and the results of phase 1 are summarized in Table 5.10.

Table 5.10: R^2 scores for case 3 using 90,000 data points for various numbers of neurons per layer and layers.

Neurons per layer	1 layer	2 layers	3 layers	4 layers	5 layers
4	0.62	0.62	0.62	0.62	0.00
8	0.63	0.68	0.64	0.66	0.68
16	0.64	0.68	0.73	0.71	0.73
32	0.65	0.73	0.83	0.86	0.85
64	0.67	0.77	0.85	0.91	0.92
128	0.70	0.80	0.88	0.91	0.91
256	0.73	0.83	0.92	0.94	0.93

The table shows that a higher score can be obtained using more layers and more neurons per layer, which is a similar result compared to cases 1 and 2. The option that is selected to be varied for phase 2 is the one using 5 layers and 64 neurons per layer, as that option resulted in a score of 0.92 and had a reasonable computation time. The results of phase 2 are shown in Figures 5.11 and 5.12.

Table 5.11: R^2 scores for case 3 using **MMS** and 90,000 data points, 5 layers and 64 neurons per layer for various activation functions, learning rates and solvers.

Activation function	Initial learning rate	Solver	R^2 Score
Tanh	0.001	ADAM	0.67
Tanh	0.001	SGD	0.61
Tanh	0.0005	ADAM	0.67
Tanh	0.0005	SGD	0.57
Tanh	0.0001	ADAM	0.68
Tanh	0.0001	SGD	0.52
Tanh	0.005	ADAM	0.67
Tanh	0.005	SGD	0.58
ReLU	0.001	ADAM	0.85
ReLU	0.001	SGD	0.64
ReLU	0.0005	ADAM	0.72
ReLU	0.0005	SGD	0.67
ReLU	0.0001	ADAM	0.88
ReLU	0.0001	SGD	0.30
ReLU	0.005	ADAM	0.63
ReLU	0.005	SGD	0.57

Table 5.12: R^2 scores for case 3 using **SS** and 90,000 data points, 5 layers and 64 neurons per layer for various activation functions, learning rates and solvers.

Activation function	Initial learning rate	Solver	R^2 Score
Logistic	0.001	ADAM	0.91
Logistic	0.001	SGD	0.00
Logistic	0.0005	ADAM	0.73
Logistic	0.0005	SGD	0.00
Logistic	0.0001	ADAM	0.73
Logistic	0.0001	SGD	0.00
Logistic	0.005	ADAM	0.90
Logistic	0.005	SGD	0.00
Tanh	0.001	ADAM	0.91
Tanh	0.001	SGD	0.73
Tanh	0.0005	ADAM	0.94
Tanh	0.0005	SGD	0.66
Tanh	0.0001	ADAM	0.73
Tanh	0.0001	SGD	0.92
Tanh	0.005	ADAM	0.67
Tanh	0.005	SGD	0.92
ReLU	0.001	ADAM	0.93
ReLU	0.001	SGD	0.84
ReLU	0.0005	ADAM	0.92
ReLU	0.0005	SGD	0.73
ReLU	0.0001	ADAM	0.73
ReLU	0.0001	SGD	0.73
ReLU	0.005	ADAM	0.83
ReLU	0.005	SGD	0.67

The tables show the result of using the scalers SS and MMS. It shows not much variation in scores, which is similar to cases 1 and 2. SGD in combination with the logistic activation function resulted in a score 0.0, which shows the lack of robustness of this option, this was also visible for case 2. The best options for case 3 are listed in Table 5.13.

Table 5.13: Hyperparameters and scores of best options using case 3.

Hyperparameter \ Options	A	B	C
Learning rate	0.0005 (adaptive)	0.001 (adaptive)	0.001 (adaptive)
Sample size	90,000	90,000	90,000
Inputs $\rightarrow$ Outputs	18 $\rightarrow$ 2	18 $\rightarrow$ 2	18 $\rightarrow$ 2
Hidden layers	5	5	4
Neurons per layer	64	64	256
Activation function	Tanh	ReLU	ReLU
Solver for weight optimization	ADAM	ADAM	ADAM
Batch size	50	50	50
R^2 score	**0.94**	**0.93**	**0.93**

The scores obtained of the three test cases will be compared in the next subsection.

5.4.6. Tuning result

Based on the results it can be concluded that case 2 performed the worst. Case 1 and 3 scored similarly, but it should be noted that case 1 only used half the number of data points compared to case 3. Thus, the expectation is that case 1 is the best, but this will be made sure by testing case 1 using the same number of data points. This basically means a limited rerun of phases 1 and 2 of case 1, but now using the proper number of data points. This leads to the result as shown in Table 5.14.

Table 5.14: R^2 scores for case 1 using 90,000 data points for various numbers of neurons per layer and layers.

Neurons per layer	3 layers	4 layers	5 layers
4	0.41	0.41	0.40
8	0.57	0.58	0.58
16	0.60	0.60	0.60
32	0.60	0.85	0.90
64	0.86	0.90	0.94
128	0.93	0.94	0.95
256	0.93	0.96	0.96

A reasonable case from phase 1 is selected to test phase 2 again of case 1. The combination of 5 layers and 64 neurons per layer is selected. The result of phase 2 is shown in Table 5.15.

Table 5.15: R^2 scores for case 1 using 90,000 data points for various activation functions, learning rates and solvers.

Scaler	Activation function	Initial learning rate	Solver	R^2 Score
Standard	Logistic	0.001	ADAM	0.61
Standard	Logistic	0.001	SGD	0.40
Standard	Logistic	0.0005	ADAM	0.61
Standard	Logistic	0.0005	SGD	0.40
Standard	Tanh	0.001	ADAM	0.59
Standard	Tanh	0.001	SGD	0.61
Standard	Tanh	0.0005	ADAM	0.60
Standard	Tanh	0.0005	SGD	0.61
Standard	ReLU	0.001	ADAM	0.94
Standard	ReLU	0.001	SGD	0.61
Standard	ReLU	0.0005	ADAM	0.94
Standard	ReLU	0.0005	SGD	0.60
MinMax	Logistic	0.001	ADAM	0.40
MinMax	Logistic	0.001	SGD	0.40
MinMax	Logistic	0.0005	ADAM	0.40
MinMax	Logistic	0.0005	SGD	0.40
MinMax	Tanh	0.001	ADAM	0.58
MinMax	Tanh	0.001	SGD	0.54
MinMax	Tanh	0.0005	ADAM	0.60
MinMax	Tanh	0.0005	SGD	0.49
MinMax	ReLU	0.001	ADAM	0.88
MinMax	ReLU	0.001	SGD	0.52
MinMax	ReLU	0.0005	ADAM	0.88
MinMax	ReLU	0.0005	SGD	0.50

Figures 5.14 and 5.15 show that case 1 indeed is the best case, as a score of 0.96 has not yet been obtained before. The best option is summarized in Table 5.16. Case 1 was also expected to turn out to be the best method, since Kepler elements provide more details about the behaviour of orbits compared to Cartesian elements. This also means that Cartesian elements have more variety, making it more difficult to catch the behaviour. Cartesian elements could however still be useful in cases where a (nearly) singular orbit needs to be propagated. In those cases the Kepler elements do not vary that much along the orbit, and thus less information can be extracted from that data. In this thesis project the behaviour of multiple orbits need to be modelled, and that is why using Kepler elements in theory should turn out as a better method. It was also expected that case 1 was better than case 3, since in case 3 more behaviour needs to be modelled by the neural network, which makes the problem slightly more complicated. More details on the performance of the best option will be discussed in the next chapter.

Table 5.16: Hyperparameter values and score of best option of tuning process.

Hyperparameter \ Option	Case 1
Learning rate	0.001 (adaptive)
Sample size	90,000
Inputs $\rightarrow$ Outputs	15 $\rightarrow$ 4
Hidden layers	5
Neurons per layer	256
Activation function	ReLU
Solver for weight optimization	ADAM
Batch size	50
R^2 score	**0.96**

6 Results

This chapter describes the main results coming from the neural network as described in the previous chapter, which is described in Section 6.1. After that some fine-tuning is performed on the result to see if a better result can be obtained. Section 6.2 describes the effect of the batch size on the performance of the neural network. Section 6.3 describes the status of overfitting and underfitting on the current solution. Section 6.4 describes the effect of the number of data points used per asteroid. Section 6.5 describes the importance of input variables. Section 6.6 describes the effect of predicting the outputs separately. Section 6.7 describes the effect of using only input data that is of high interest. Section 6.8 describes a different scaling method applied to predicting the error caused by the uncertainties in the state and the dynamics model separately. Section 6.9 addresses result bias and Sections 6.10 and 6.11 show, analyze and tune the performance of the best result. The contents of this chapter are summarized in Section 6.12.

6.1. Main result

Until this point the focus of the neural network and the tuning process lies on finding a solution with the largest R^2 score. A score however does not provide any insight in the performance of the model and therefore a closer look will be taken at how this transforms to predictions of an actual Earth distance and an error caused by uncertainties in the state and the dynamics model. The result used in the previous chapter obtained after the tuning process showed a best score of 0.96, but this chapter will use a result with a similar performance (R^2 0.95, 4 layers and 256 neurons per layer, see Table 5.14) to reduce the computational load of the fine-tuning processes. The best result at the end will clearly be used, see Section 6.11. The mean and standard deviation of the error made by the machine-learning process is calculated for the result of a score of 0.95 for both the Earth distance and the error caused by the uncertainties in the initial state and the dynamics model. This is shown in Table 6.1.

Table 6.1: Mean and standard deviation of the machine-learning error for a score of 0.95 of the Earth distance and the error caused by the uncertainties in the initial state and the dynamics model.

	Machine-learning error in Earth distance	Machine-learning error in error made by uncertainties
Mean	$3.42 * 10^7$ km	$1.17 * 10^6$ km
Standard deviation	$3.83 * 10^7$ km	$4.65 * 10^6$ km

The mean of the actual Earth distances is $2.9 * 10^8$ km, meaning that the error of the Earth distance is on average 11.67%. This could be a reasonable result, but the standard deviation shows that this result depends to a huge extent on the data point. This can also be visualised, and is shown in Figure 6.1. The figure shows that the machine-learning error is reasonable for large Earth distances ($> 10^8$ km). The error becomes on average bigger for smaller Earth distances, and this means a larger percentage when comparing this to the actual distance. To put the values in perspective: 10^8 km corresponds to 0.67 AU.

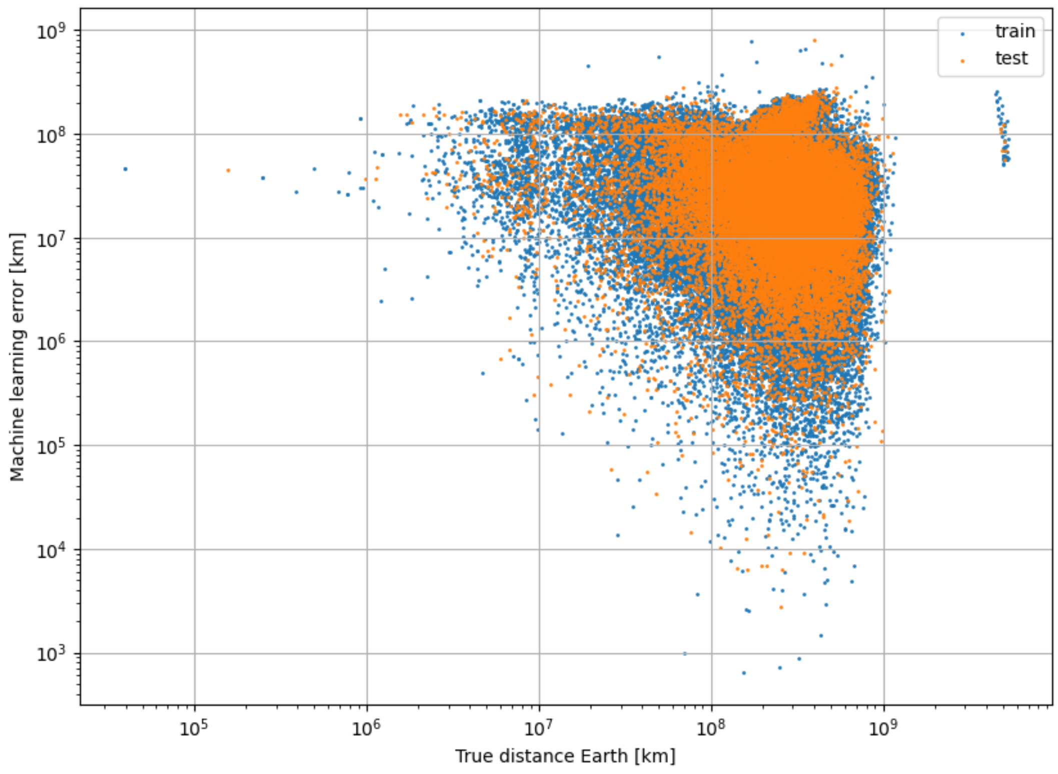


Figure 6.1: Error made by machine learning compared to the actual Earth distance for every data point for the trained and tested data set.

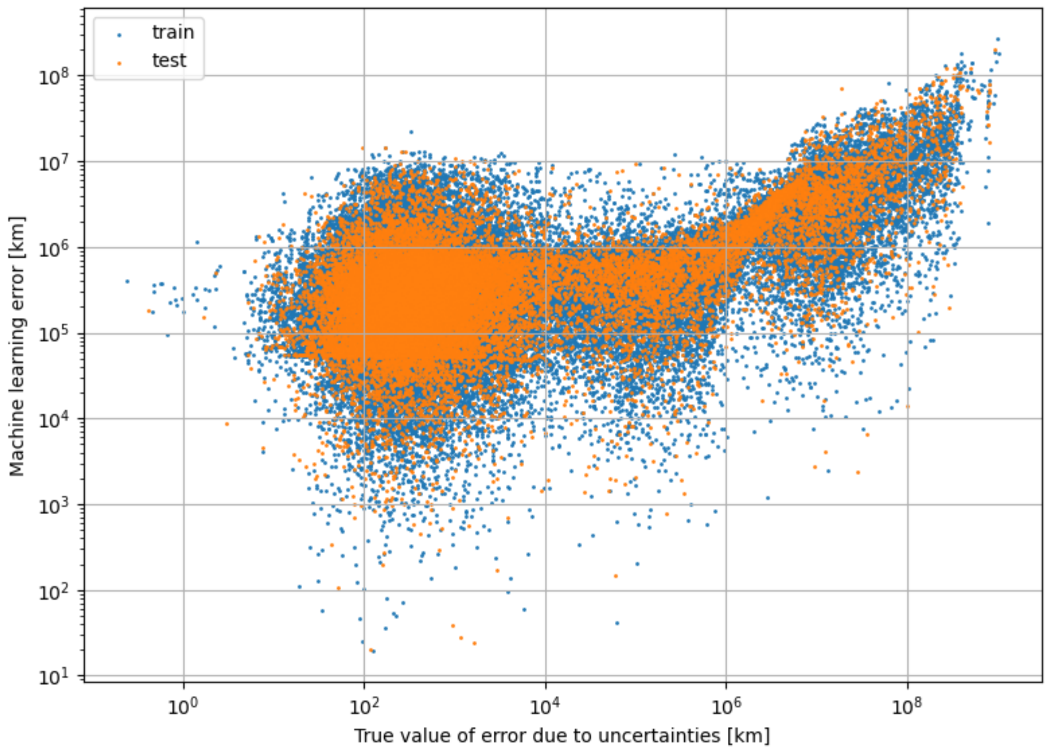


Figure 6.2: Error made by machine learning compared to the actual error caused by the uncertainties in the state and the dynamics model for every data point for the trained and tested data set.

Moreover, the error is in the order of 10^7 km on average and it was found in Chapter 3 that perturbations only cause an error in the order of 10^6 km. This shows that a pure Kepler orbit solution is even more accurate than the machine-learning model. It can thus be concluded that the machine-learning model found here can not be used to reliably predict Earth distances.

The machine-learning error made in the error caused by the uncertainties in the state and the dynamics model has an even bigger relative standard deviation (see Table 6.1) and the results are shown in Figure 6.2. It shows an even more arbitrary behaviour and this could be explained by the fact that the number of outputs in this final case was four, and three of them were the position of the asteroid and only one was considered as the error caused by the uncertainties in the state and the dynamics model. This means that more focus lies on estimating the asteroid position. On top of that, the magnitude of this error is on average much smaller compared to Earth distances, and is therefore more difficult to estimate well.

This however only shows the result of one specific case, with a score of 0.95. Therefore it is investigated what the effect is or might be when a higher or lower score is obtained. This is visualised in Figure 6.3.

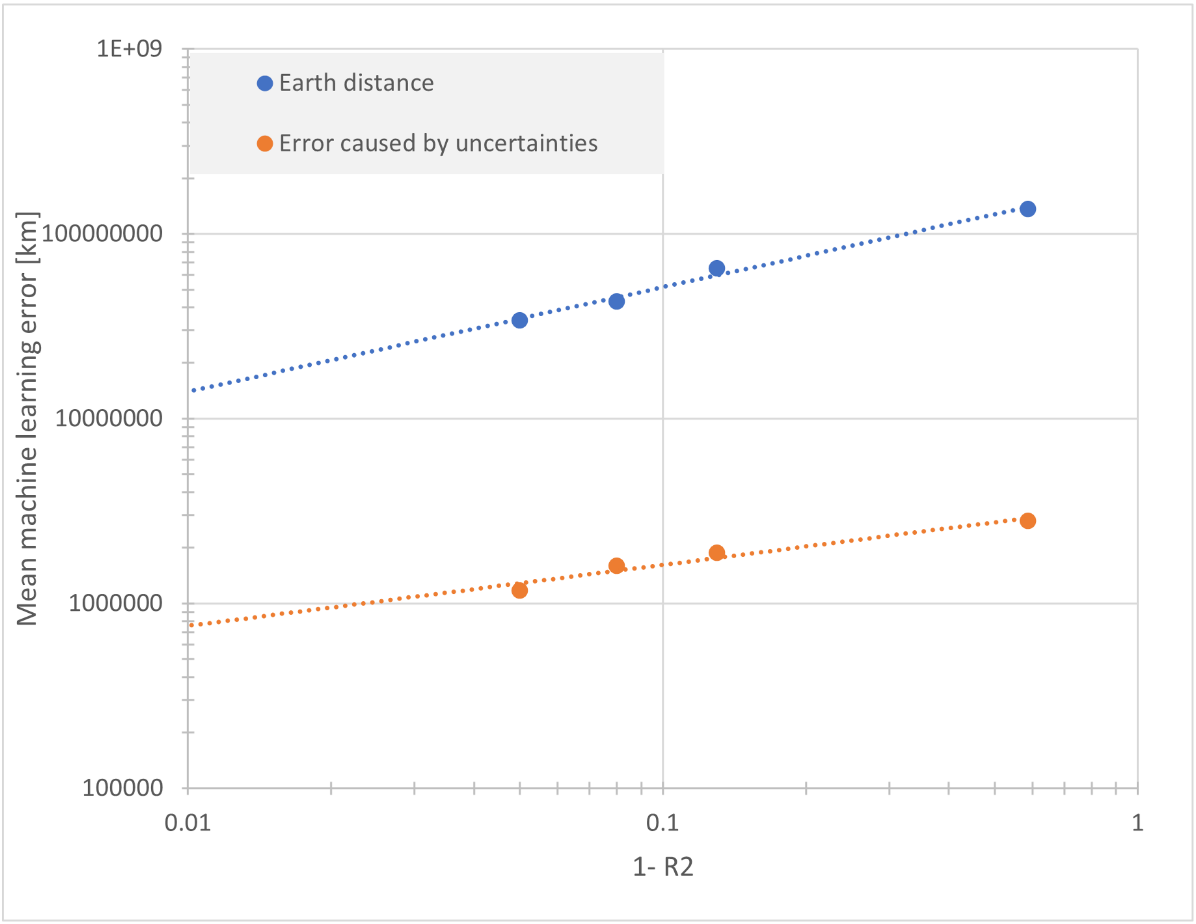


Figure 6.3: Score with respect to mean machine-learning error for both the Earth distance and the error caused by the uncertainties in the state and the dynamics model. The y-axis shows the 1-R^2 score and the dotted lines are trend lines.

The figure shows that a score of 0.95 (most left points) still has a mean larger than 10^7 km and a score of 0.99 will result in a hypothetical mean error of $1.5 * 10^7$ km according to the trend line, which is approximately two times smaller than the case with a score of 0.95. The error is still large for a score of 0.99, but the mean error does not represent the true situation completely, as the standard deviation also defines a part, and that is even not the full picture of the performance of the model. Nevertheless, it shows that a difference of 0.01 or smaller does affect the results only by a marginal amount and is probably not worth spending too much time on. It also shows that a different approach should be used in order to decrease the mean error significantly. This holds for both outputs.

It should be noted that the accuracy is discussed in this result, but not the computation time. The computation time is however not important, since the training of the neural network takes place only once. After the model has been trained, the model can be used and computes the solution for an arbitrary new input almost instantly. The accuracy is thus the only key parameter that defines the performance of the result.

Figures 6.1 and 6.2 show the machine learning error for all the test data, but how is the prediction exactly applied to a newly discovered PHA? The Earth distance is predicted for several propagation intervals, and in this way the most critical point can be determined, which is the point where the PHA becomes the closest to Earth. The uncertainty value is also predicted for these points, and this is visualised for the unnumbered asteroid (2021 TG4), shown in Figure 6.4.

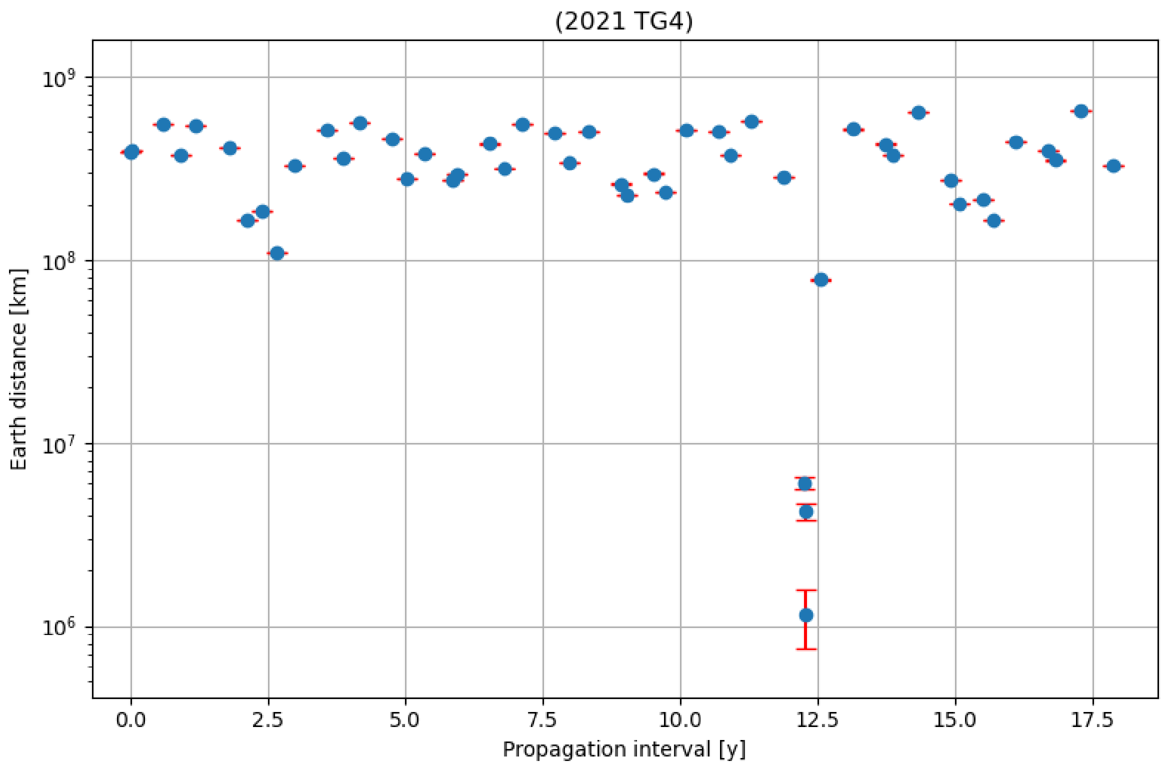


Figure 6.4: Predicted Earth distance and uncertainty for the asteroid 2021 TG4.

The figure shows that the PHA becomes the closest to Earth at a propagation time of 12.3 years (in 2035), which is just above a distance of 10^6 km. The uncertainty is shown as an error bar in red, which is also predicted.

6.2. Batch size

The most important hyperparameters are now set, but not all the hyperparameters were investigated yet. One of these is the batch size, which was currently set at 100. The batch size defines the size of the minibatches used in the iterative process of the neural network. The batch size is varied for the calculation also used in Table 5.15 and this leads to the result as shown in Figure 6.5.

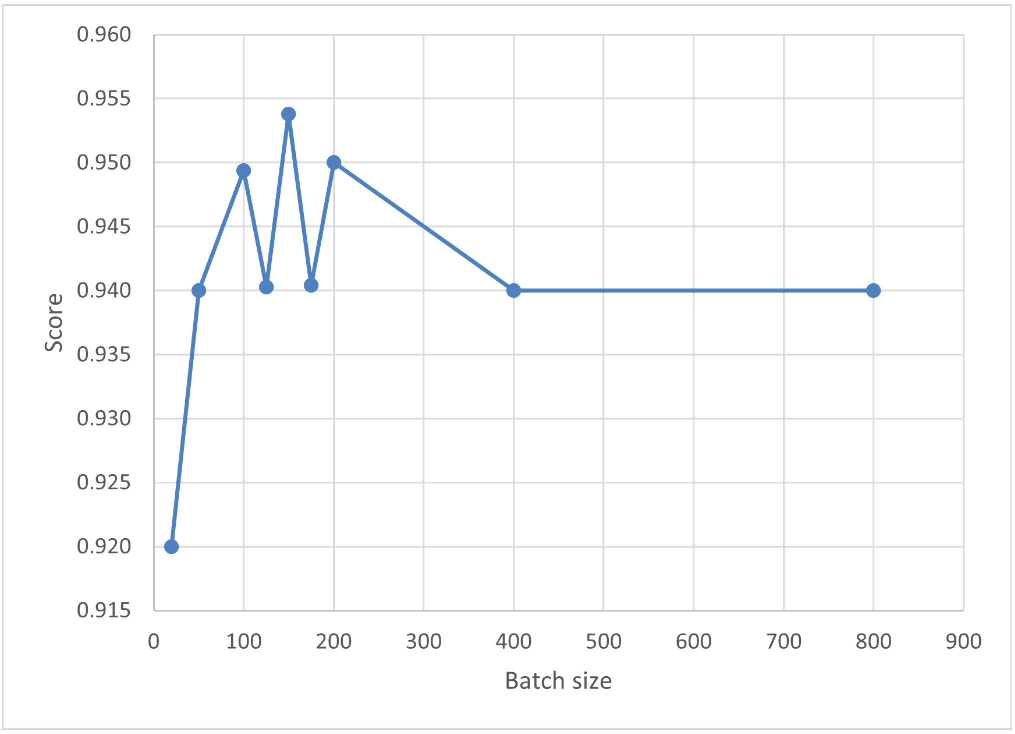


Figure 6.5: Effect of batch size on score.

Figure 6.5 shows the effect of the batch size and it can be seen that the effect is not large, which was expected as this hyperparameter was considered as less important. Larger batch sizes were not tested, as the expectation is that the score becomes much worse for a too large batch size, as the model does not generalize in that case [22]. The batch size seems to have an optimum around a small value, usually between 1 and 500, and for this specific case it seems that the most optimal value lies between a batch size of 100 and 200. However since the scores are so close to each other and the fact that the machine-learning process is not perfect it is difficult to choose the best batch size. The batch size for the upcoming analyses will be kept at 100 and will not be changed anymore, as the score increase is very marginal and can not lead to a significant better result (see Figure 6.3).

6.3. Under- and overfitting

Another fine-tuning procedure considers the observation of underfitting or overfitting. This is done by observing the internal validation score every iteration to see if the iterative process stops too late or to run more iterations to see if the validation score increases. The iteration currently (also used for the whole tuning process) stops after 20 iterations if the validation score does not decrease anymore by more than 10^{-4}. The validation score did not drop significantly after 20 iterations (in the order of 10^{-3}) and overfitting does thus not take place. The number of iterations is increased to observe underfitting, but this resulted in an R^2 score increase of 0.005 (0.956 to 0.961). The R^2 score for training the data is 0.98 for the current best option and increased to 0.99 when running more iterations. Since the difference becomes larger, overfitting becomes slightly more visible. Taking this in mind and the fact that the score only increased by 0.005, it is decided not to change the number of iterations and this also means that under- or overfitting is currently not present in the training process of the neural network, which is desired.

6.4. Data points per asteroid

Another investigation is on the effect of the number of data points per asteroid. Section 5.3 explained the current approach, which is 50 data points per asteroid, combining minimum distances and an even sampling for the remainder. A random and a structured approach will be tested based on this data. A random approach takes randomly n data points out of the total 50 for each PHA, whereas the structured approach takes the first n data points out of the total 50 for each PHA. Comparing the structured approach to the random approach results in Figure 6.6.

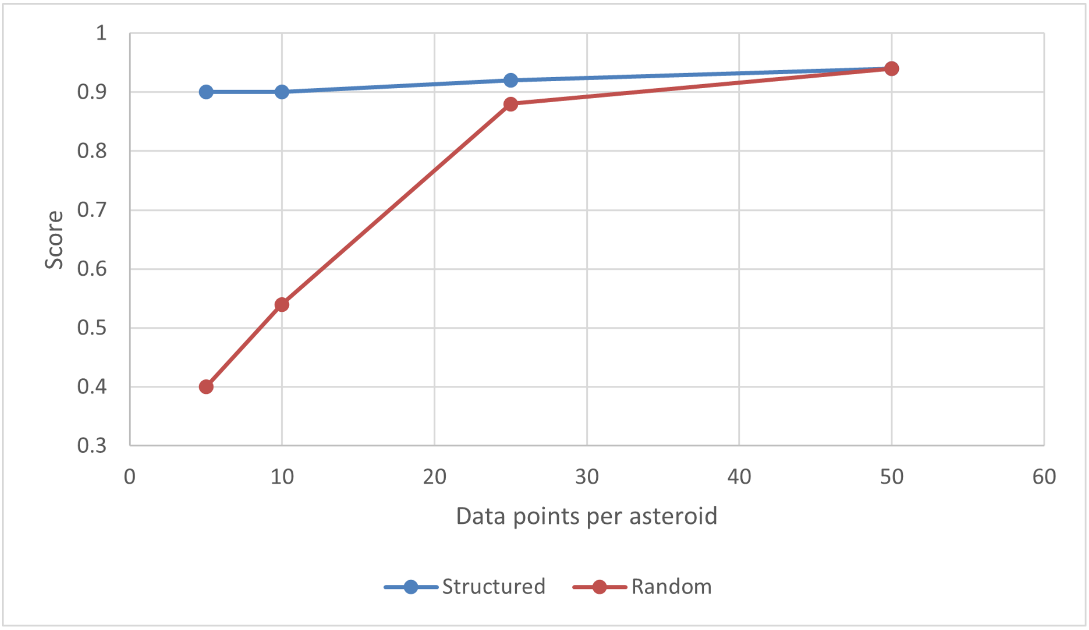


Figure 6.6: Scores for number of data points per asteroid for a structured and a random approach.

Figure 6.6 shows that the structured approach gives more consistent results. The sparseness of the random approach does seem to have a large effect on the score. This means that options using less data points in a structured approach can be used for time-consuming tuning processes. It should be noted however that this option only uses data points with an integration time up to a certain year (e.g. from 0 to 5 years), and can thus not be used when the integration time is longer. This result also means that maybe an even better score can be obtained when more data points are used in the same time-frame. It would nevertheless take more time to train the data, since more data are used. It also takes extra time to generate that data, and is due to this time constraint not considered for this research.

6.5. Feature importance

One last minor fine-tuning idea is to investigate the effect of the input variables. The influence of input variables may provide insight in the data. It also could potentially speed up the training process, as a variable that does not contribute that much can be removed. The importance of input variables is measured by two different methods. One is to delete an input variable and see what the score becomes if that input variable is missing. The second method is more sophisticated and also a function in sci-kit learn [38], called feature importance. The importance of a feature is determined by shuffling the weights connected with an input variable (the neurons of the first hidden layer are computed by weights of the neurons in the input layer, see Section 5.3). The score is computed for various numbers of shuffles and the mean is compared to the optimal score (no shuffle). The impact of each input variable is measured in this way. The scores obtained by these methods are provided in Table 6.2.

The left-hand side of Table 6.2 shows that the most important input variables are the Kepler elements and the propagation time. This was also expected, since these variables contribute to all outputs (Earth distance and uncertainty). The radius of the asteroid has the smallest effect, because the radius only contributes to the solar radiation pressure component, which is a small part in the calculation of the total trajectory. The feature importance shows a similar result, the only difference is the inclination being less important and the Δa being more important. Nevertheless, more or less the same result is found.

Table 6.2: Effect of missing input variables.

Input variable	R^2 score - deleting input variable	R^2 score - Feature importance
a	0.640	0.126
e	0.942	0.897
i	0.580	0.758
ω	0.570	0.000
Ω	0.570	0.068
θ	0.590	0.291
R	0.958	0.950
Δa	0.944	0.718
Δe	0.940	0.828
Δi	0.940	0.938
Δ ω	0.944	0.937
Δ Ω	0.939	0.941
Δ θ	0.943	0.947
Δ R	0.954	0.950
t_p	0.570	0.240

From this information it could be concluded that some variables are very small contributors to the overall score, but it should be noted that this problem is based on four outputs, and three of them represent the position of the asteroid, which means that the uncertainty variables only contribute to one output and are having therefore already less impact. It would thus be worthwhile looking at a neural network with separate outputs in order to investigate the need of these variables in more detail. In Section 6.1 it was already identified that a score increase of only 0.01 does not improve the result that much. Thus, if separate outputs does not change the result obtained in this section, then some variables might become irrelevant (e.g. the PHA radius and the uncertainty in the PHA radius).

6.6. Separate outputs

The investigation of separate outputs is split up into two options, one about the prediction of the Earth distance and one about the prediction of the error caused by the uncertainties in the state and the dynamics model. The first alternative considers the distance to Earth. The network is trained for the case using 5 layers and 64 neurons per layer, which is also used to compare with the results obtained in the previous section. Figure 6.7 shows the result and it is very similar to that of Figure 6.1, and the mean of this result is 9.5% of the mean of $2.9 * 10^8$ km, which is slightly better than the 11.67% obtained using multiple outputs. The result is better, but still far from reasonable. Especially for smaller Earth distances the error is much larger than 100%. It is therefore concluded that using separate outputs does not result in a significant better outcome for the Earth distance.

The prediction of the error caused by the uncertainties in the state and the dynamics model will also be investigated separately, which also hopefully improves the importance of the uncertainty variables (see Table 6.2). The result of this is shown in Figure 6.8. The figure shows that the error is similar as what was seen before when using multiple outputs (see Figure 6.2). It can thus be concluded that separate outputs do not necessarily mean that a better result is obtained.

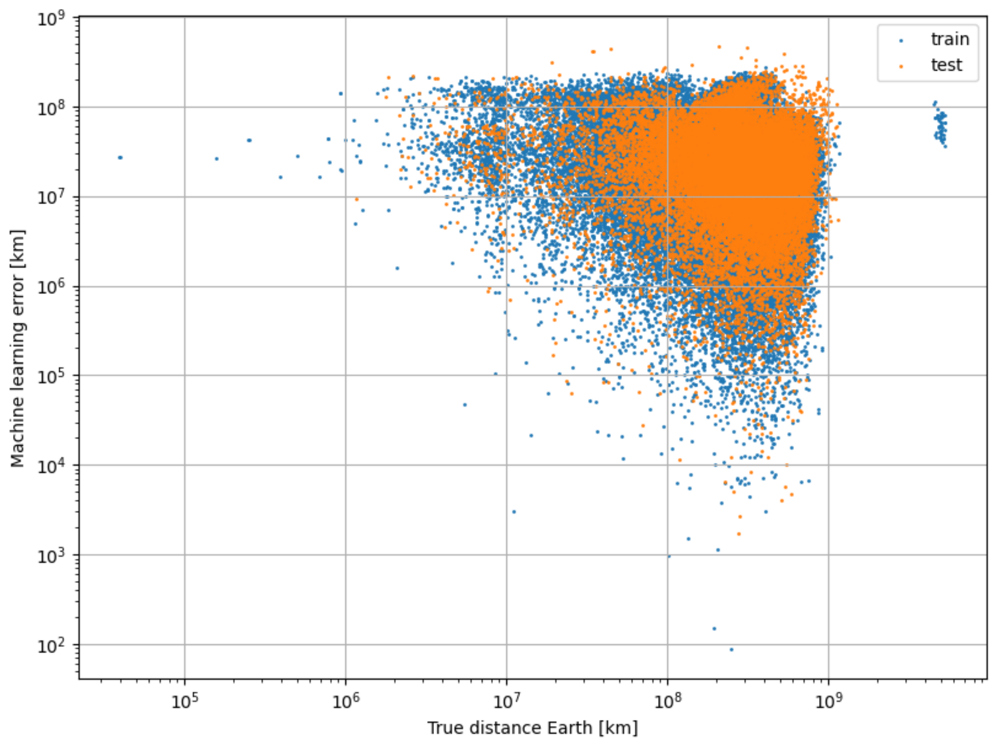


Figure 6.7: Error made by machine learning compared to the actual Earth distance for every data point for the trained and tested data set using separate outputs.

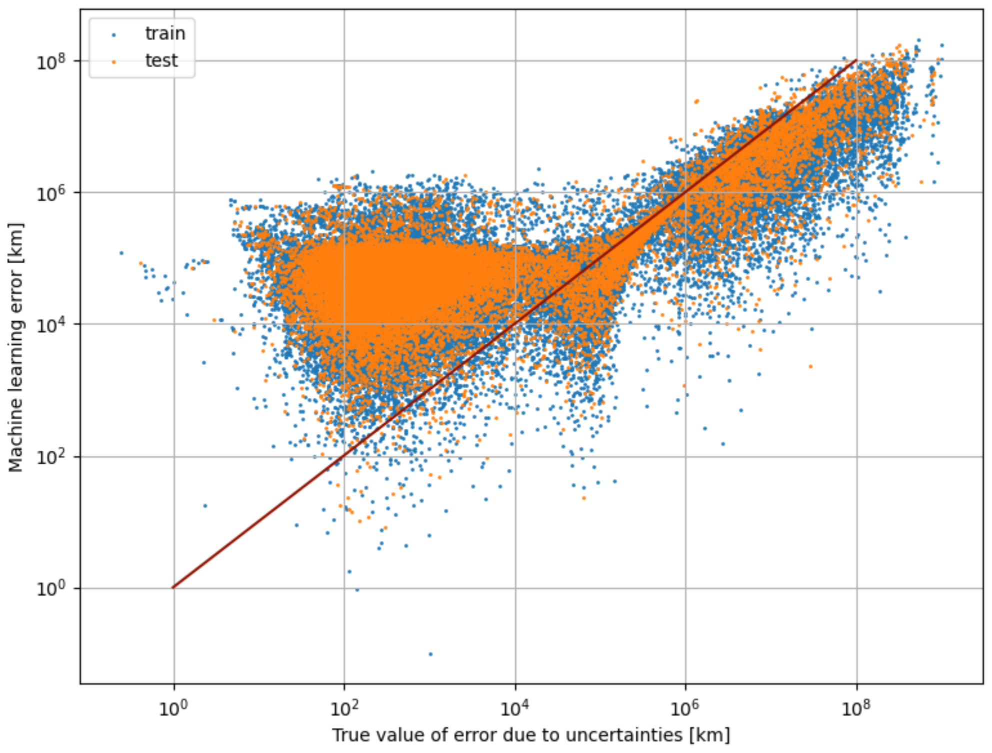


Figure 6.8: Error made by machine learning compared to the actual error caused by the uncertainties in the state and the dynamics model using one output. The red line shows the 100% error border.

6.7. Close approach data

It has not been tested yet if a better outcome can be obtained when only using a part of the data that is more interesting. These are the points where the Earth distance is relatively small. Two different slices are tested for this investigation, one using distances up to 10^8 km (15,000 data points) and one using distances up to $3 * 10^8$ km (65,000 data points). This is again tested for both outputs separately. Figures of results only shown in tables in this section can be found in Appendix E.

6.7.1. Data fraction - Earth distance

The results for the Earth distance are shown in Table 6.3 and it shows that the machine-learning error becomes larger for a smaller number of data points. This however does not necessarily mean that a worse output is obtained for the more interesting data points. This is checked visually, by looking at the graph of the case using 15,000 data points. This is shown in Figure 6.9, which shows that the more interesting points do not provide better results when comparing it to the case when all data points are used (Figure 6.7). The result for 65,000 points is shown in Appendix E and shows a similar result. It was already seen that at least a certain number of data points were required to obtain a reasonable solution. In this specific case there are unfortunately too few interesting data points in the data set. In order to perform a reasonable test, the data file needs to be regenerated and since this takes too much time this will not be considered for this research. Nevertheless, it would be a great recommendation to generate these interesting data points and see how well the neural network is able to predict these more interesting points. From the test performed in this research it can be concluded that using a specific fraction of the data does not result in a better outcome.

Table 6.3: Machine-learning error when using only a fraction of the data.

Data points	Machine-learning error - mean [km]
15,000	$4.29 * 10^8$ (148%)
65,000	$1.59 * 10^8$ (54.7%)
112,050	$2.76 * 10^7$ (9.50%)

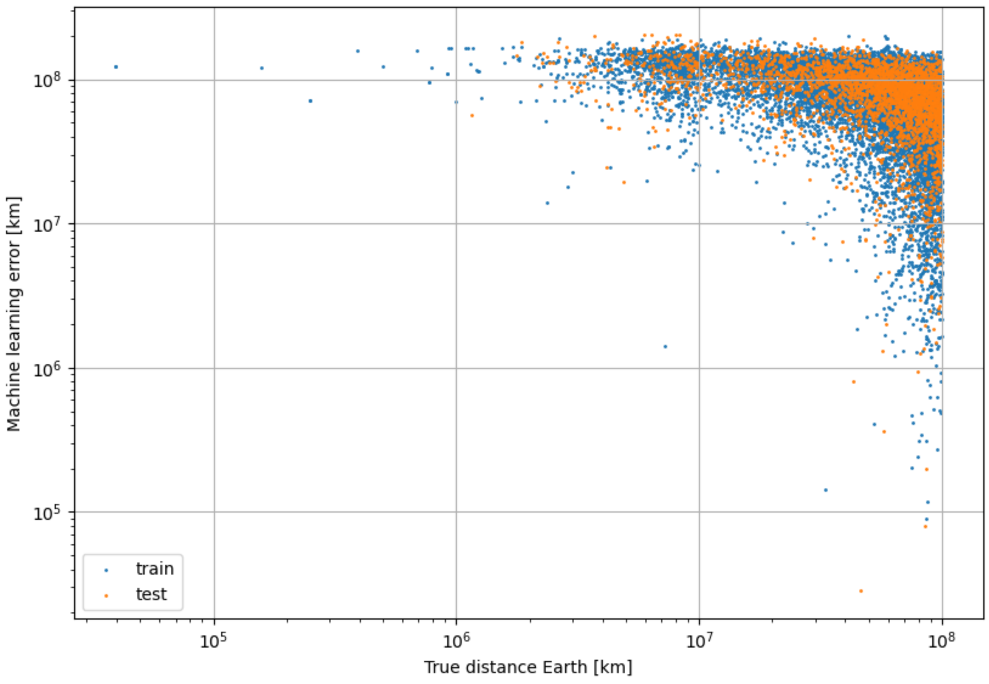


Figure 6.9: Error made by machine learning compared to the actual Earth distance for every data point for the trained and tested data set using 15,000 data points.

6.7.2. Data fraction - uncertainty

This test was performed after the test of the different scaling method, described in the next section. The different scaling method was also used here, and this section only focuses on the effect of using less data points, not on the effect of using a different scaler. The results for the uncertainty are shown in Table 6.4.

Table 6.4: Machine-learning error of the error caused by the uncertainties in the state and the dynamics model when using only a fraction of the data.

Data points	Machine-learning error [%]	Machine-learning error - random points [%]
15,000	47.8%	49.5%
65,000	48.2%	51.7%
112,050	30.0%	30.0%

The results are shown as percentages of the true error caused by the uncertainties in the state and the dynamics model. The table shows a small effect which is caused by the lack of data. This means that using more data is still worthwhile, and therefore all the data will be used for the upcoming tests. It should also be noted that the percentages are very close for a case when random data points are taken. This shows that the specific fraction of interesting points does not improve the performance significantly.

6.8. Different scaling method - uncertainty

A different scaling method can be used in order to improve the score of the prediction of the error caused by the uncertainties in the state and the dynamics model, one that is more applicable for an output with a substantial amount of variability. The log value will be taken on the output variable, whereas the input variables will still use the StandardScaler, which was used in the previous investigation. This leads to Figure 6.10, which is using the same settings of the neural network as before.

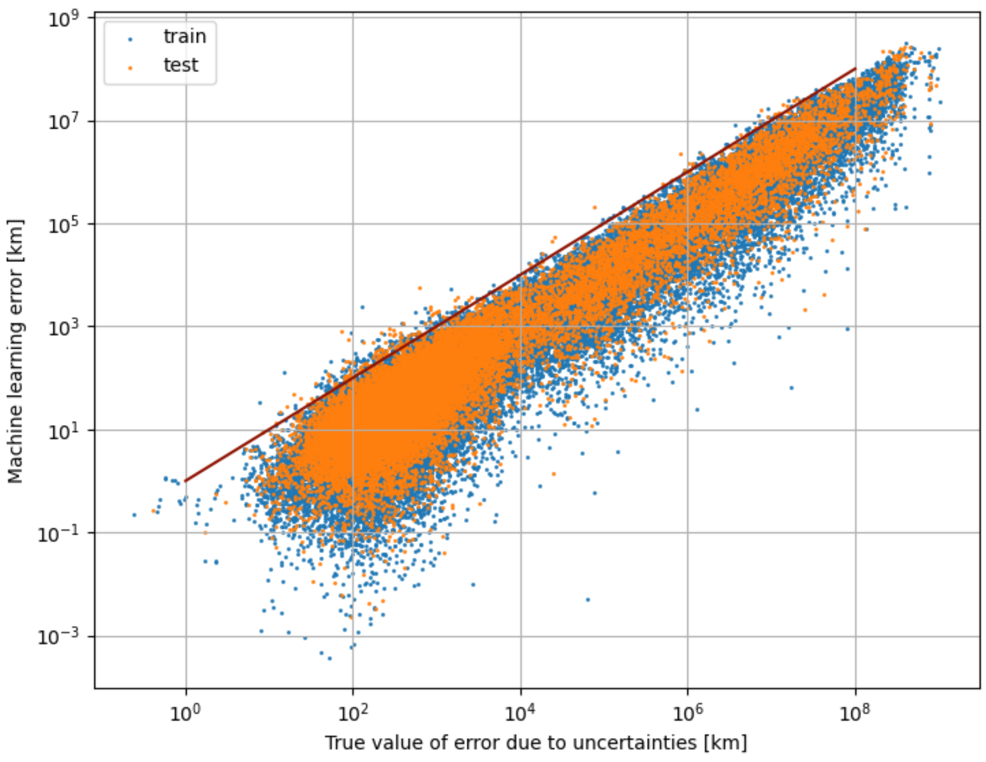


Figure 6.10: Error made by machine learning compared to the actual error caused by the uncertainties in the state and the dynamics model using one output and a log scaling on the output. The red line shows the 100% error border.

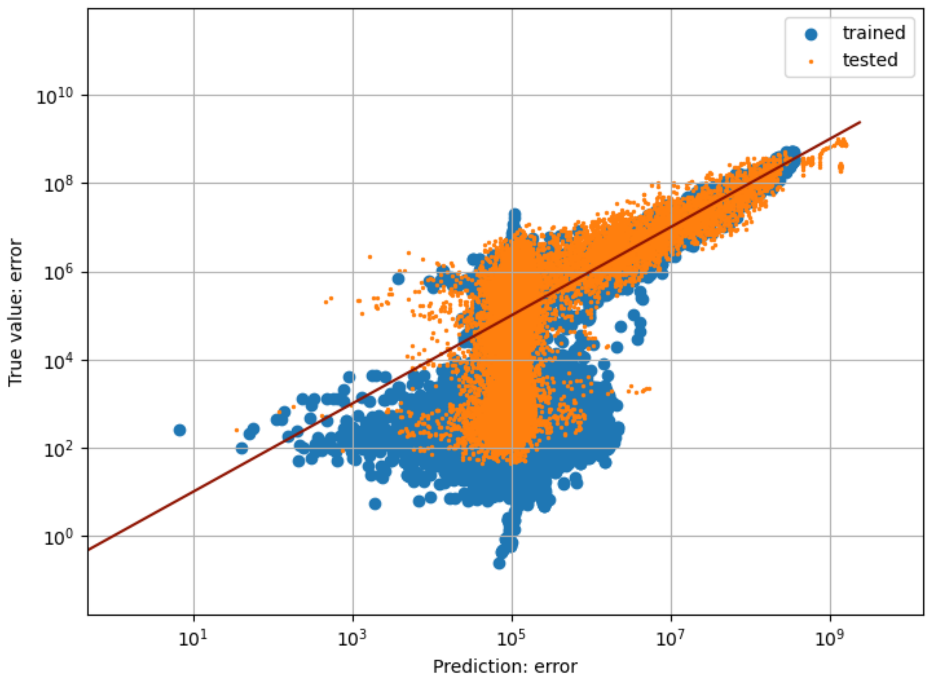


Figure 6.11: Predicted vs true values of the error caused by the uncertainties in the state and the dynamics model **using the StandardScaler** on the output variables.

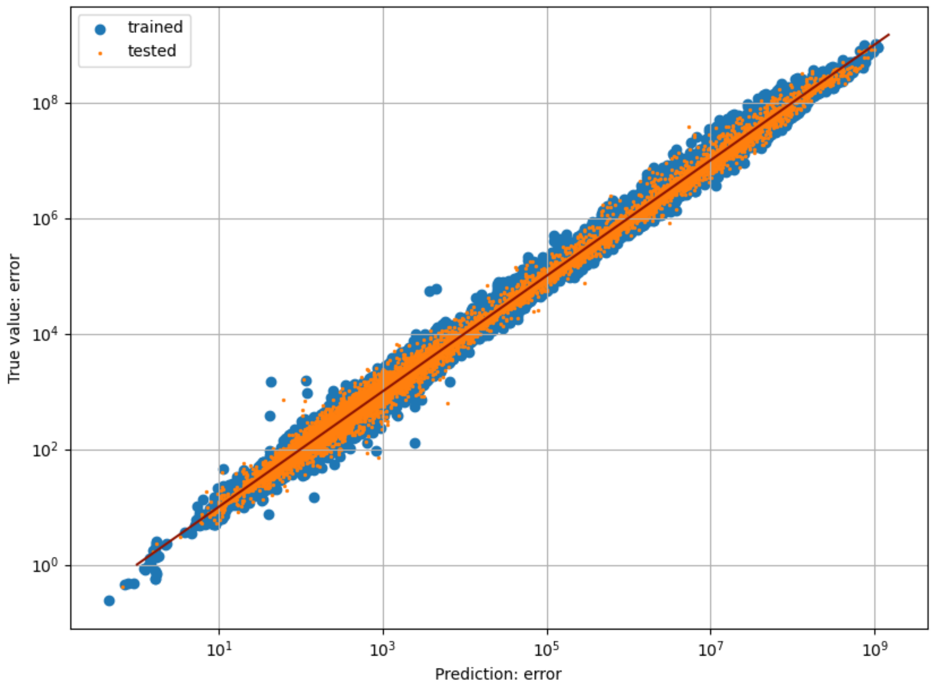


Figure 6.12: Predicted vs true values of the error caused by the uncertainties in the state and the dynamics model **using a log scale** on the output variables.

As can be seen from Figure 6.10, the result looks significantly better compared to the previous result. This can also be seen from the figures showing the deviation of the predicted values, as shown in Figures 6.11 and 6.12, where the StandardScaler is used in the first figure, which is compared to the method using the log scaling, as shown in the second figure.

The computed standard deviation is still large, but this is only due to the fact that the numbers are scaled now, this is thus not representative anymore. Therefore percentages are introduced, showing the error as a percentage of the true error caused by the uncertainties in the state and the dynamics model. In this case the mean error is 23.5% and the standard deviation 31.3%. This result shows that it is possible to estimate the error caused by the uncertainties in the state and the dynamics model better than the same order of magnitude.

6.9. Result bias

Until this point the results from the neural network are unbiased in a sense that the test data are correctly separated from the training data (random sampling). However there might exist some correlation between the training and the test data. The test data are picked randomly, and since 50 data points per asteroid are used, some data points for the same asteroid are part of the test data, and some are part of the training data. If the training data contains information about that asteroid, it is more likely to predict data points for that same asteroid better compared to a completely new asteroid. The expectation is that this difference is not large, but to be able to make sure that all bias is filtered out of the results, the same run will be performed using 80% of the PHAs for the training data and a completely independent 20% as the test data. The result is shown in Table 6.5.

Table 6.5: Result comparison between the case of using random data points (bias) and the case of using a specific number of asteroids for the training data (no bias).

Error	Bias	No bias
Mean [%]	23.5	30.0
Standard deviation [%]	31.3	62.4

The table shows a difference in both the mean and the standard deviation. It should be noted however that the standard deviation is very sensitive, as percentages are used. This will also be made more clear once the performance is discussed in Section 6.11. The mean is also different but already closer. It is therefore assumed that this result does not influence earlier found results or drawn conclusions. Bias will be removed from the best result from now on.

6.10. Best result tuning

The different scaling method showed a significant better result compared to earlier obtained results. The R^2 score of this result with the bias removed is 0.988, which is much higher than the 0.95 achieved before (R^2 0.95, 4 layers and 256 neurons per layer, see Table 5.14). This result was not used during the tuning process and it might mean that a different better set of hyperparameters exist that results in an even better optimum. This case is thus checked again for phases 1 and 2, similar to Section 5.4. A reasonable case from phase 1 is selected to test phase 2, which is 5 layers and 64 neurons per layer. The results are shown in Tables 6.6 and 6.7. The figures can again be found in Appendix D.

Table 6.6: R^2 scores for the best result using 90,000 data points for various numbers of neurons per layer and layers.

Neurons per layer	1 layer	2 layers	3 layers	4 layers	5 layers
4	0.976	0.977	0.977	0.977	0.981
8	0.982	0.980	0.978	0.984	0.978
16	0.981	0.983	0.973	0.977	0.982
32	0.982	0.980	0.983	0.980	0.981
64	0.977	0.981	0.987	0.987	0.988
128	0.978	0.978	0.989	0.989	0.985
256	0.976	0.984	0.990	0.989	0.990

Table 6.7: R^2 scores for the best result using 90,000 data points for various activation functions, learning rates and solvers.

Scaler	Activation function	Initial learning rate	Solver	Score
Standard	Logistic	0.001	ADAM	0.971
Standard	Logistic	0.001	SGD	0.982
Standard	Logistic	0.0005	ADAM	0.977
Standard	Logistic	0.0005	SGD	0.977
Standard	Tanh	0.001	ADAM	0.941
Standard	Tanh	0.001	SGD	0.971
Standard	Tanh	0.0005	ADAM	0.962
Standard	Tanh	0.0005	SGD	0.974
Standard	ReLu	0.001	ADAM	0.988
Standard	ReLu	0.001	SGD	0.972
Standard	ReLu	0.0005	ADAM	0.981
Standard	ReLu	0.0005	SGD	0.974
MinMax	Logistic	0.001	ADAM	0.983
MinMax	Logistic	0.001	SGD	0.976
MinMax	Logistic	0.0005	ADAM	0.983
MinMax	Logistic	0.0005	SGD	0.000
MinMax	Tanh	0.001	ADAM	0.962
MinMax	Tanh	0.001	SGD	0.984
MinMax	Tanh	0.0005	ADAM	0.968
MinMax	Tanh	0.0005	SGD	0.981
MinMax	ReLu	0.001	ADAM	0.985
MinMax	ReLu	0.001	SGD	0.982
MinMax	ReLu	0.0005	ADAM	0.983
MinMax	ReLu	0.0005	SGD	0.982

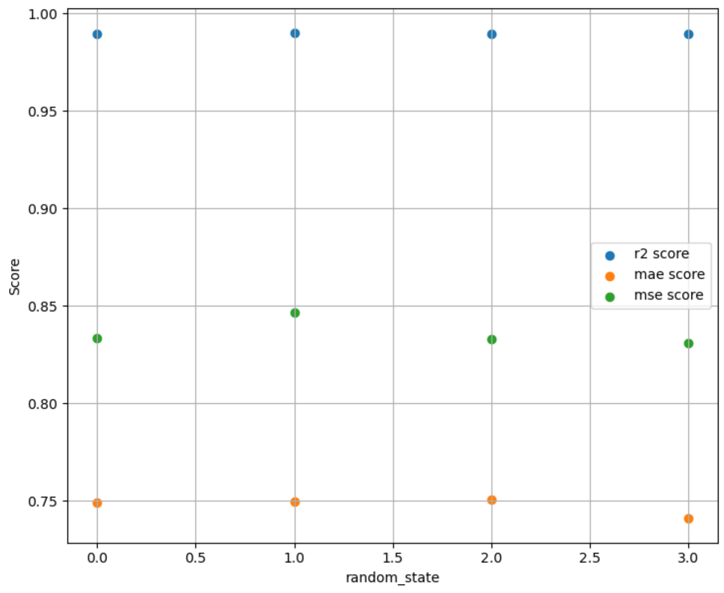


Figure 6.13: Cross validation of option using 3 layers and 256 neurons per layer.

Table 6.6 shows interesting scores. The best solution was obtained for the case with 3 layers and 256 neurons per layer, resulting in a mean error of 27.7% and a standard deviation of 51.7%. The R^2 score is 0.990, which is slightly higher than the 0.988 that was found before.

Table 6.7 shows that most of the hyperparameters do not result in a better model, and the combination between a logistic activation function and SGD as the solver for weight optimization is sometimes unstable, shown by the score of 0.000. This is similar to what was seen before.

The option using 3 layers and 256 neurons per layer is used for cross validation to see the robustness of the result. The results of this cross validation can be found in Figure 6.13. The R^2 values of the options in the cross validation are shown in Table 6.8.

Table 6.8: R^2 scores of options using a different random seed.

Seed	R^2 score
0	0.9894
1	0.9902
2	0.9894
3	0.9892

The table shows that the scores are very close to each other, meaning that the result is very robust and that randomness did not influence the tuning process.

6.11. Best result performance

The best result was found for estimating the error caused by the uncertainties in the state and the dynamics model. The error made by the neural network is shown for every data point compared to the true value in Figure 6.14, and the percentages are shown in Figure 6.15. The R^2 score is 0.9902 for the test data and 0.9972 for the training data set and this corresponds to a mean of 27.7% and a standard deviation of 51.7%. The hyperparameters of the neural network used can be found in Table 6.9.

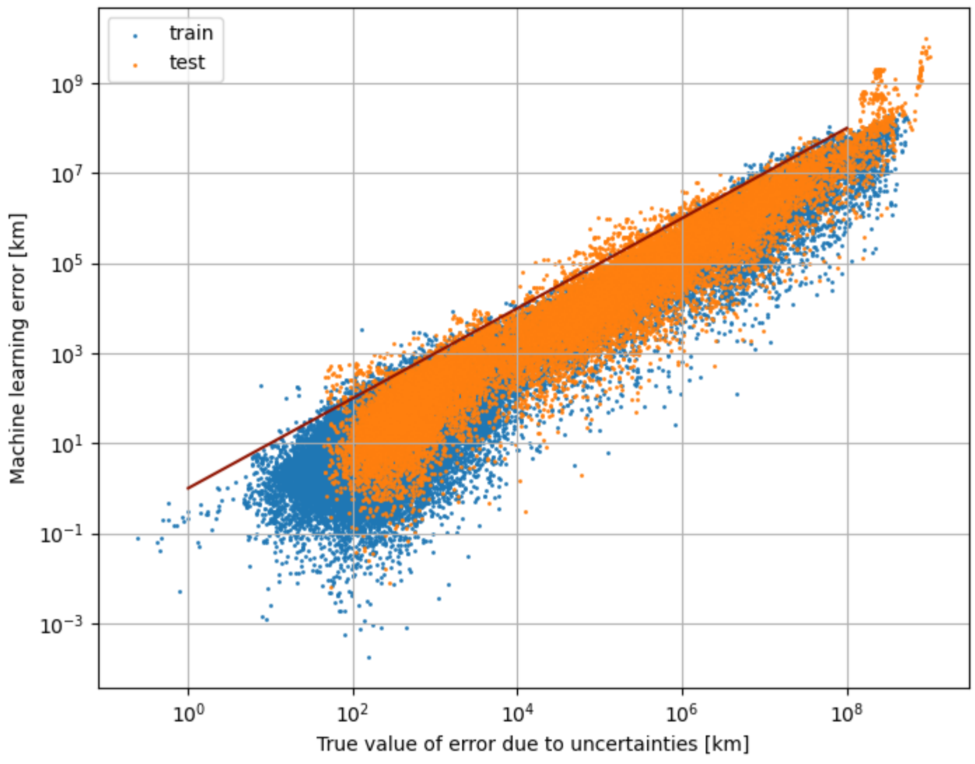


Figure 6.14: Machine-learning error compared to the true value of the error caused by the uncertainties in the state and the dynamics model for both the training and test data set for the best result.

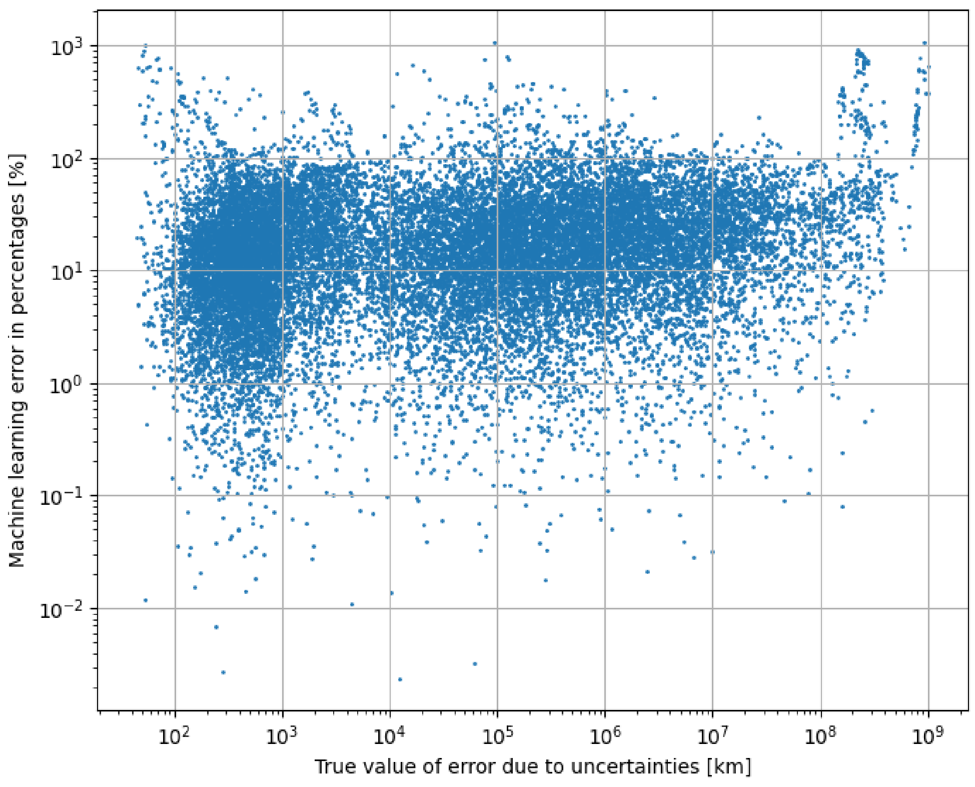


Figure 6.15: Machine-learning error in percentages compared to the true value of the **test** data.

Table 6.9: Hyperparameters of neural network predicting the error caused by the uncertainties in the state and the dynamics model.

Hyperparameter	Value
Scaler	Standard/Log
Learning rate type	Adaptive
Initial learning rate	0.001
Activation function	ReLu
Solver	ADAM
Neurons per layer	256
Layers	3
# of data points	112,500
Batch size	100
# Iterations without change	100
Maximum # of iterations	1500
Random seed	1

The mean and the standard deviation provide information on the usability of this result for the estimation of the error made by the uncertainty for PHAs. It should however be noted that it is difficult to know for which PHAs this result is reasonable, as it looks like some of the points do have a large percentage error. The test data contains three asteroids having an uncertainty in the eccentricity greater than 0.1. This already caused issues with computing the error, due to the fact that the eccentricity can not be smaller than 0 or larger than 1 to ensure eccentric orbits. The three asteroids are marked orange in Figure 6.16, which is also the right-hand side of Figure 6.15.

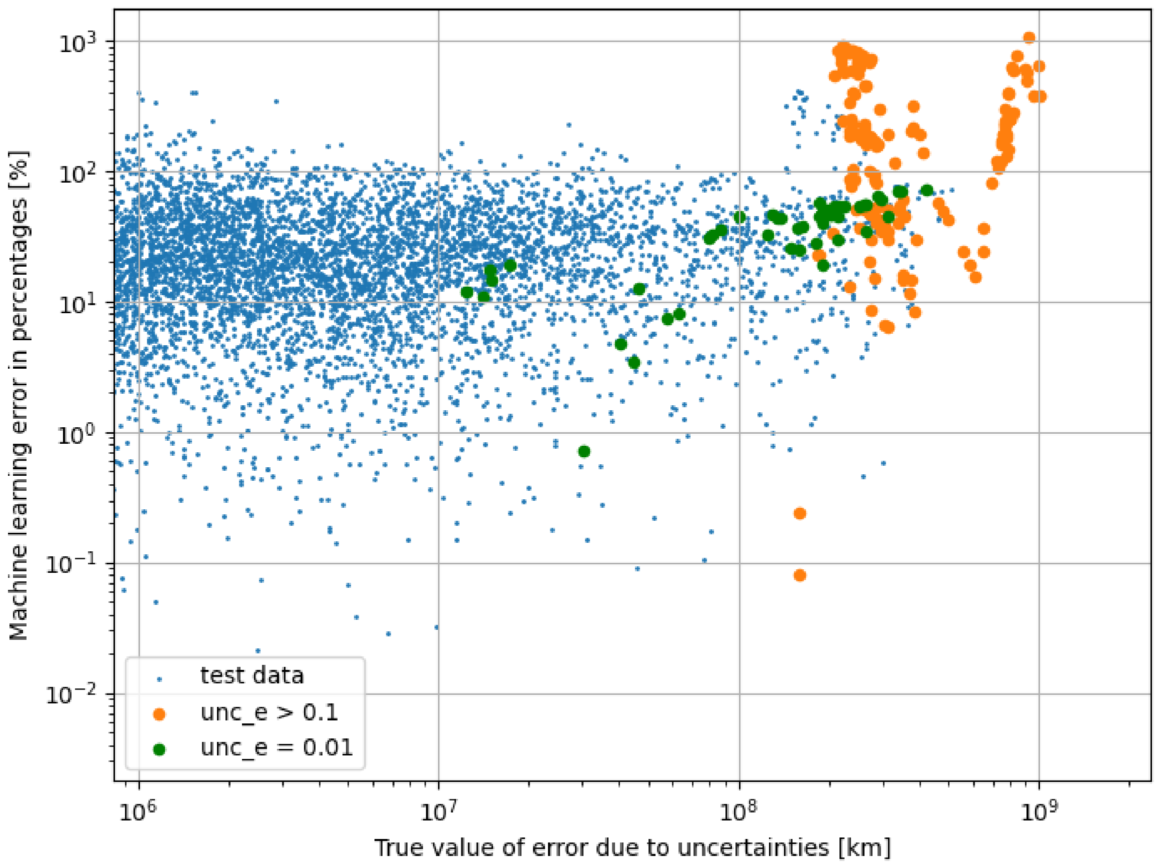


Figure 6.16: Machine-learning error in percentages compared to the true value of the **test** data. PHAs with an eccentricity uncertainty larger than 0.1 are shown in orange. One PHA with an eccentricity uncertainty of 0.01 is shown in green.

The figure shows that the three PHAs are outliers and also show that PHAs with a too high uncertainty value can not be estimated using this model. One PHA with an eccentricity uncertainty of around 0.01 was also tested, and is shown in green in Figure 6.16, and shows an average result. This means that this model can predict PHA trajectories as long as the uncertainty in the eccentricity is smaller than or equal to 0.01. It is assumed that a similar conclusion can be drawn for other orbital elements, as the uncertainties are usually within the same order of magnitude for the same PHA.

Removing the three outliers results in a mean of 26.1% and a standard deviation of 41.8%. This shows that the standard deviation is sensitive, as the three asteroids are only 3/448 = 0.67% of the test set. This also arises the question of the impact of the standard deviation, especially since percentages can not be negative, and for that a figure is created that shows the percentage of the data that is covered compared to the machine-learning error in percentages. For example, 70% of the test data has a machine-learning error smaller than 27%. This is also shown in Figure 6.17.

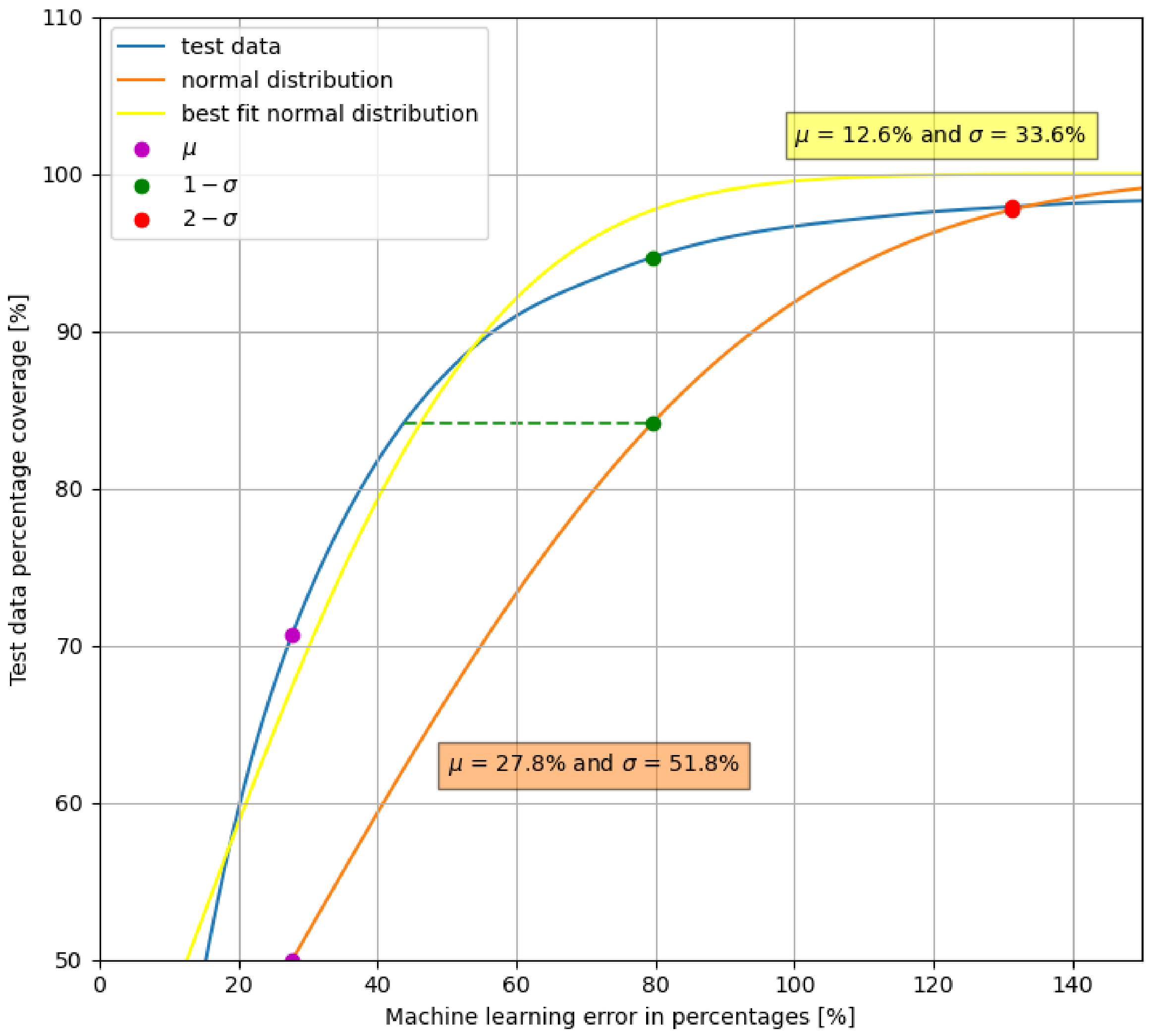


Figure 6.17: Mean and standard deviation of the machine-learning error in percentages compared to the coverage of these percentages. The test data are compared to a normal distribution determined from the mean and standard deviation of the test data. The standard deviation with the best fit is also shown.

The figure shows that 84% of the test data has a maximum error of 45%. This value is compared to a perfect normal distribution, which is created based on the estimated mean and standard deviation values. The one-sided $1-\sigma$ coverage for a perfect normal distribution is 84% and from this it can be computed that the maximum machine-learning error should be $27.8+51.8 = 79.6$%. This point is shown in green in the figure and the difference with the actual data is shown by the dotted green line. The test data thus performs significantly better than the estimated normal distribution. The mean and standard deviation values that are the most similar with the test data are found to be 12.6% and 33.6%, as shown in the figure in yellow. This values were found by fitting a normal distribution based on the plot, which ignores outliers and only focuses on the right-hand side of the normal distribution. That is the reason why the yellow line is different from the orange line. The yellow line is still not the truth, but shows that the performance of a model can not solely be extracted from the mean and standard deviation of the machine-learning error.

6.12. Summary

The main result from the previous chapter showed an R^2 score of 0.95. The fine-tuning processes have resulted in a significant improvement in the R2 value, from 0.95 to 0.990. This is mainly due to a different scaling method, namely a log scale on the output variables. This result holds for the prediction of the error caused by the uncertainties in the state and the dynamics model, resulting in a mean error of 27.8%. The Earth distance can not be determined from a neural network, as the result found was even worse compared to a Kepler orbit.

All in all, machine learning can be used for the estimation of uncertainty errors of PHAs. The nominal trajectory should therefore be propagated using a more classical propagation model. The nominal trajectory thus does not save time, but the uncertainty propagation is usually time-consuming and if an MC simulation is considered, the propagation needs to take place at least 50 times to have a reasonable estimate of the error caused by the uncertainties in the state and the dynamics model. CNEOS even mentions 'millions' or 100,000 runs to be able to propagate the uncertainties well enough [10], making this method even more attractive purely based on a significant potential time save. The neural network can be used to determine the uncertainty errors for several different propagation times and the Earth distance for these times can be extracted from the nominal trajectory. The estimation of the uncertainty error can be computed almost instantly, which means that this method is at least 50 up to 100,000 times faster than current methods. It should be noted that the accuracy is worse compared to using a propagation model, since an extra component is added to the total error, coming from the neural network. This is however not a problem as long as the initial state uncertainties are in the order of 10^{-2} or smaller. The average error is 27.8% and 84% of the estimates have an error smaller than 45%. The model is still limited to an average value of the error caused by the uncertainties in the state and the dynamics model, and does not provide information of the variation in this error if the initial uncertainties are varied, which is something that could be concluded from propagating the orbit 100,000 times.

7 Verification and validation

This chapter describes the verification and validation cases that are necessary to both verify and validate all the steps that were taken throughout this research. First the verification and validation cases of the propagation model are described in Section 7.1. The verification and validation cases concerning the PHA data are described in Section 7.2. The verification and validation of the neural network is described in Section 7.3.

7.1. Propagation model

As described in Chapter 3, a propagation model was created to propagate the position and velocity of an asteroid in time. This model requires verification and validation in order to make sure that the model behaves as expected and that the model is correct.

The integration consists of three main parts: the pre-processing, the main processing and the post-processing part. The pre-processing phase comprises the acquisition of the necessary inputs for the main process. These inputs are for example the position of all the planets in the Solar System, or the definition of all the different force models that were included in the integration. This information is acquired using the Tudat tool, which is assumed to be verified and validated extensively already, as also mentioned during the literature study [4]. The only other inputs for the integration model are the initial state and the uncertainties for a few asteroids, which was extracted from the JPL database. This information will be verified in the next section, which also includes measurement data and the relations between this measurement data and the states and uncertainties of all PHAs.

The main process is the propagation model itself and therefore requires verification and validation. The test cases presented in this section will also directly verify the post-processing part. The tests are thus more a combination of integration tests and model tests, rather than unit tests. It is assumed that these tests are sufficient to prove the validity of the propagation model.

7.1.1. Test case 1: Unperturbed orbit

The perturbations are ignored in the first test case, making the problem a two-body problem. This means that the analytical solution for a Kepler orbit should be more or less equal to the solution coming from the integration (not exactly, due to rounding errors or integration errors). This also directly verifies the used benchmark, as the benchmark should resemble the true orbit solution, and the exact same procedure is followed here. The orbit of Apophis is integrated backwards in time for 20 years from the provided state of 2022. The differences for all the Kepler elements can be found in Figure 7.1.

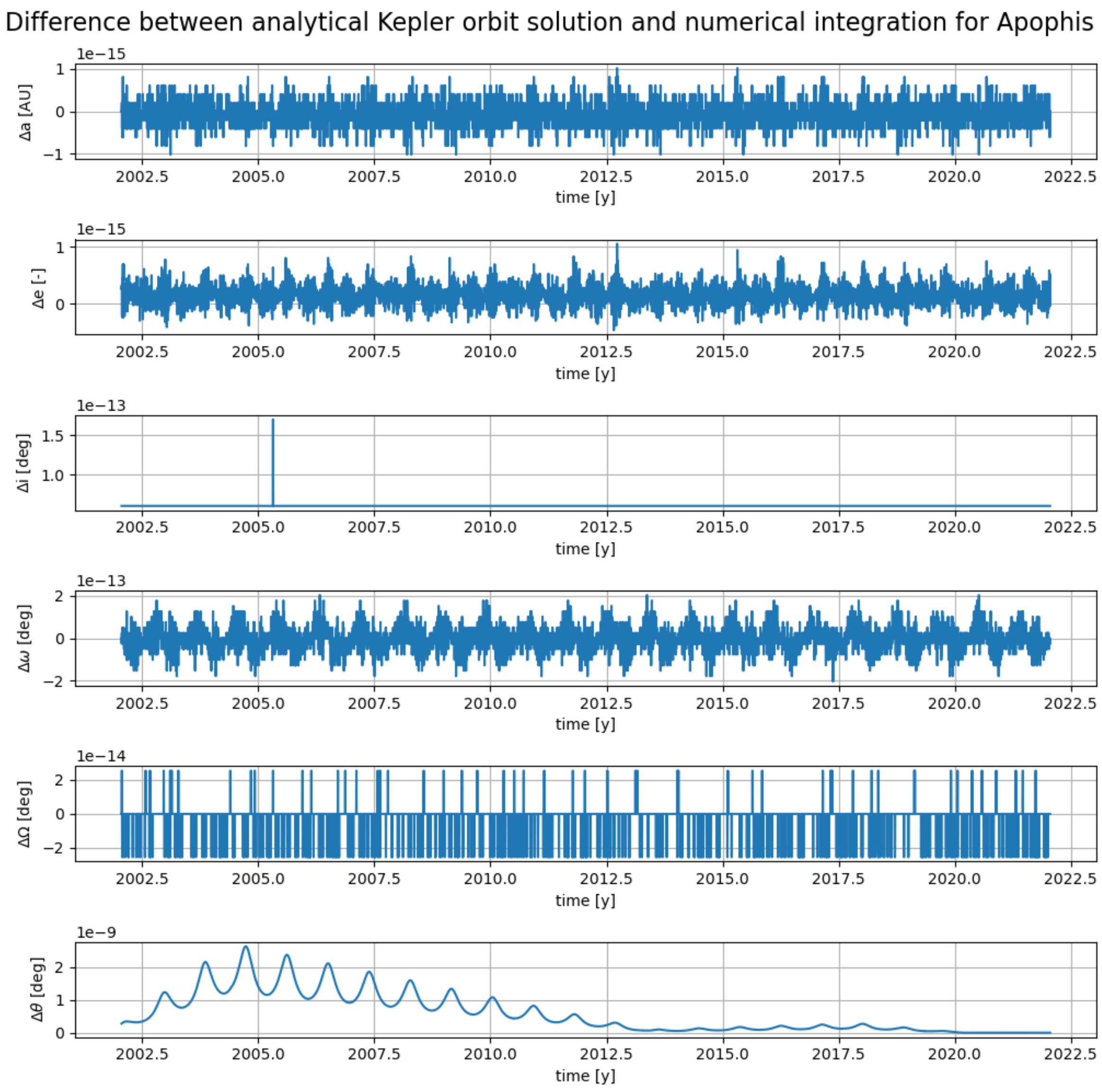


Figure 7.1: Difference between an analytical Kepler orbit and the result of the integration model for Apophis, integrated over a timespan of 20 years.

The figure shows that the differences are very small for all the orbital elements. Δa is only 10^{-15} AU, which comes down to 0.15 mm. This is much more accurate than any of the accuracy levels derived from the initial state uncertainties, which are 10^{-10} AU at the smallest [33]. Besides, the 10^{-15} in particular is close to machine precision (10^{-16}), making it very likely that the error is caused by rounding errors. This also holds for other elements, although the error is slightly larger, this is due to the fact that the initial state is converted to a Cartesian state, which is necessary for Tudat in order to work properly. This conversion adds to the error, as more computations are performed and the error from machine precision increases. Although there is an error component for all the orbital elements, it is very good that the error does not increase over time, and that the error levels are far from becoming a problem with respect to the uncertainty levels relevant in this thesis project. There is however one exception, as the error does increase for the true anomaly, but this component is known to be sensitive to a slight offset. This is also the reason why it is one of the factors to determine the uncertainty parameter U for minor planets [8]. It is therefore concluded that the differences with respect to a Kepler orbit are so small that the solution from the integration model is considered to be good. This test case is therefore now verified successfully.

7.1.2. Test case 2: Integration error

The second test case checks the magnitude of the integration error. This will be checked for the RK4 integrator, known to have an integration error related to the step size as shown in Equation 7.1 [49].

$$\epsilon = \mathcal{O}(\Delta t^4) \tag{7.1}$$

A few different step sizes were tested and the result can be found in Table 7.1.

Table 7.1: Final position error (Δr) as function of a number of step sizes (Δt). The factor shows how much the error increases every time the step size becomes twice as large.

Δt [s]	Δr [m]	factor
28800	$5.54 * 10^2$	-
57600	$1.00 * 10^4$	18.1
115200	$1.99 * 10^5$	19.8
230400	$4.67 * 10^6$	23.5
460800	$1.38 * 10^8$	29.4
921600	$4.35 * 10^9$	31.6
1843200	$1.33 * 10^{11}$	30.6

As can be seen, the error becomes between 18 till 31 times bigger when the step size increases by a factor of two. The expectation was that the error becomes more or less 2^4 = 16 times bigger, according to Equation 7.1. Although there is a difference, it can be safely assumed that the integrator behaves correctly. The factors are in between orders four and five (2^4 - 2^5), so within the same order of magnitude, which is also the definition of Equation 7.1. Other factors will also influence the error, making it not solely dependent on the step size. Nevertheless, since the factors end up in the correct order of magnitude, the test has passed successfully.

7.1.3. Test case 3: JPL Horizons validation model

The third and last test of the integration model considers the validation. An integration can be performed for asteroids using the JPL Horizons tool [32] and this will be done using the same settings for the integration model created in Tudat. The input settings that can be selected using the Horizon tool is shown in Table 7.2.

Table 7.2: Input settings of the JPL Horizons tool for asteroid propagation.

Target body	99942 Apophis
Coordinate center	Geocentric
Time specification	21-01-2000 – 21-01-2022
Time step	1 day

The same settings were used for the integration in Tudat and the position differences are calculated and the result is shown in Figure 7.2. As can be seen from the figure, the position difference adds up to around 400 km for a propagation time of 20 years. It is unclear what the exact propagation technique used in Horizons is, and it is probably different from what has been used in Tudat, which can cause a difference. Another possible explanation could be that different force models are taken into account, for example a solar radiation pressure model is not used in the Horizons tool [32]. It is therefore concluded that the integration model includes all the necessary components, and it should also be noted that the difference is still below the set accuracy requirement of 10^3 km. The next section will also provide more verification regarding this subject.

7.2. Data verification

In Chapter 2 it was investigated how much data are available for asteroids. The osculating orbital elements that are provided for one specific state can be verified and derived from the raw measurements. This verification of the data will be described here, where first the method is explained, after which the results are presented, followed by a discussion on the differences.

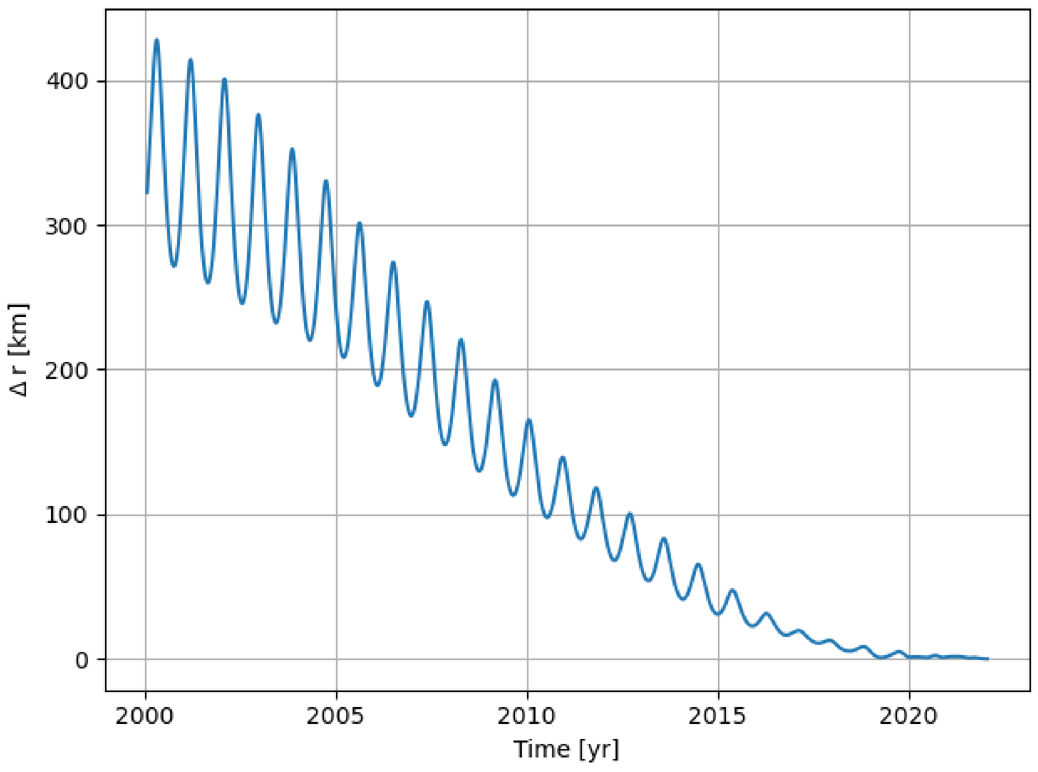


Figure 7.2: Position differences between Tudat and Horizons for the backward propagation of Apophis over a timespan of 20 years.

7.2.1. Method

The x, y and z position of the asteroid with respect to Earth is calculated using the propagation model in Tudat. The data are stored for every time instance, where the difference between two time instances is the step size. This is done in order to check the difference between the measurement data and the orbit solution from the integration model in Tudat. There are three different data types, and the right ascension (RA, α) and declination (DEC, δ) measurements will be investigated first. RA and DEC are the equivalences of longitude and latitude in 3D space. RA is measured from the vernal equinox and is ranging between 0 and 360 degrees. DEC is measured from the celestial equator and is ranging between -90 and +90 degrees. This is visualized in Figure 7.3.

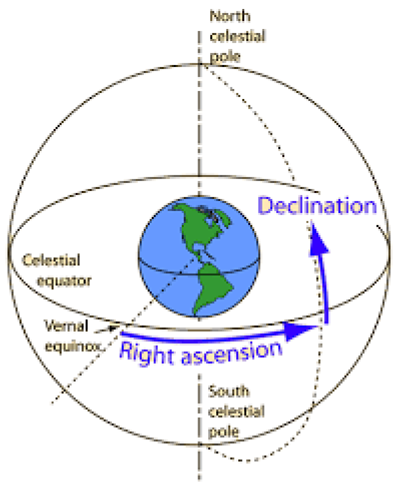


Figure 7.3: Declination and Right Ascension [35].

Every measurement data point of an asteroid consists of a time instance and corresponding RA and DEC values. The x,y and z-position of the asteroid is stored using the integration model for all the time instances that have a measurement. These positions are then transformed from the ecliptic reference frame to the equatorial reference frame via a frame transformation [47], see Equation 7.2. This is necessary since the measurement data are provided in a different reference frame.

$$\begin{bmatrix} x_{\text{equatorial}} \\ y_{\text{equatorial}} \\ z_{\text{equatorial}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varepsilon & -\sin\varepsilon \\ 0 & \sin\varepsilon & \cos\varepsilon \end{bmatrix} \cdot \begin{bmatrix} x_{\text{ecliptic}} \\ y_{\text{ecliptic}} \\ z_{\text{ecliptic}} \end{bmatrix} \tag{7.2}$$

In Equation 7.2 ε is the obliquity of the ecliptic, equal to 23.439292 degrees [47]. The RA and DEC values are then calculated from the asteroid position using the integration model with the following equations:

$$\begin{aligned} RA &= atan\left(\frac{y}{x}\right) \\ RA &+= \pi \quad if \quad x < 0 \\ RA &+= 2\pi \quad if \quad x > 0 \quad and \quad y < 0 \\ DEC &= atan\left(\frac{z}{\sqrt{x^2+y^2}}\right) \end{aligned} \tag{7.3}$$

RA is usually defined in hour angles, which is also the case for the measurements and therefore RA is divided by 15 to end up with hour angles for RA and degrees for DEC. The computed RA and DEC values are then compared to the measurement data and this is plotted for Apophis, which will be used as the main verification asteroid, as it has the most measurements (see Chapter 2).

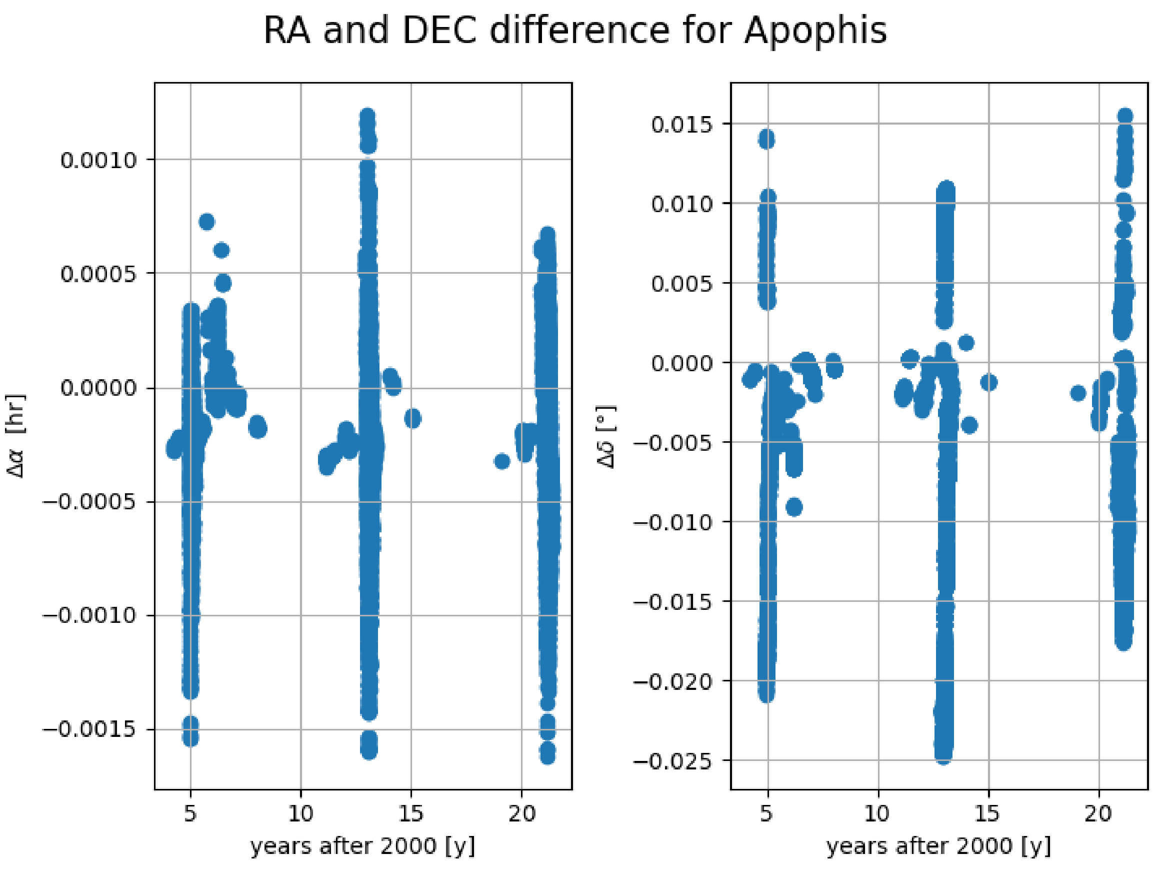


Figure 7.4: Difference in RA and DEC between the measurements and the integration model. RA is given in difference in hour angles and DEC in difference in degrees.

Figure 7.4 shows data that seems to be very concentrated about certain time periods (2005, 2013 and 2021). This is because measurements only have taken place when Apophis is relatively close to Earth, so no measurement data are available for the blank periods (e.g. 2009 - 2011). From the figure it can also be seen that the maximum differences are in the order of 10^{-3} hr for RA and 10^{-2} deg for DEC. The average differences are $3.6 * 10^{-4}$ hr and $1.1 * 10^{-2}$ deg for RA and DEC, respectively. This is only an error of 0.02% when comparing it to the true values (e.g. $\frac{1.1*10^{-2}}{45} * 100 = 0.02\%$). However it is more useful to look at the effect that these differences have on the difference in position. It was already shown in Chapter 4 that the 1-sigma uncertainties can lead to a position difference of 100 km for Apophis, and it is expected that these uncertainties are related to the differences found in this section, since the uncertainty in the initial state contributes the most to the total error.

The position was tweaked and by trial and error it was found that a position difference of about 3000 km would induce a similar average difference obtained from the results from the previous paragraph. 3000 km is about 30 times larger than the expected 100 km. Before any conclusions will be drawn on these results, other data types will also be looked at to see if similar or completely different results are obtained for these data types. Data was also provided as time delays and Doppler frequencies, also known as radar astrometry data. About 20 time delay measurements and 30 Doppler frequencies are available for Apophis. From a time delay a distance can be computed:

$$d_E = \frac{c}{2} \cdot t_d + R_E \tag{7.4}$$

where d_E is the distance between the asteroid and the center of Earth, R_E is the equatorial radius of Earth (6378.136 km [49]), c the speed of light ($2.99792458 * 10^8$ m/s [49]) and t_d the time delay. From the Doppler frequency the radial velocity can be determined. From the position and the velocity of the asteroid the radial velocity can also be determined, which is derived in Equation 7.6.

$$V_r(measurements) = \frac{c}{2} \cdot \frac{\Delta f}{f_0} \tag{7.5}$$

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2} \\ \frac{d\mathbf{r}}{dt} &= \frac{dr}{dx}\frac{dx}{dt} + \frac{dr}{dy}\frac{dy}{dt} + \frac{dr}{dz}\frac{dz}{dt} \\ \frac{d\mathbf{r}}{dt} &= \frac{dr}{dx}V_x + \frac{dr}{dy}V_y + \frac{dr}{dz}V_z \\ V_r &= \frac{x}{r}V_x + \frac{y}{r}V_y + \frac{z}{r}V_z \\ V_r(integration\ model) &= \frac{xV_x + yV_y + zV_z}{r} \end{aligned} \tag{7.6}$$

The difference for both the distances and the radial velocities are plotted in Figures 7.5 and 7.6. As can be seen from the figures, the difference in distance is about 4,000 km at its maximum. This is the same order of magnitude from what was seen for the RA and DEC. For the radial velocity difference it is a bit more difficult to quantify the error. The velocity however can also be estimated between two delay measurements, using Equation 7.7.

$$V_r = \frac{\Delta d}{\Delta t} \tag{7.7}$$

where Δt is the time between two radar time delay measurements, and Δd the difference between two ranges computed from the same two radar time delay measurements. For example for the measurements in 2013 it is known that the frequency of the measurements are on average one per day. During this period the distance differed about 3,000 km (observed from the time delay and also based on the RA and DEC). An extra 3,000 km could potentially result in an extra velocity of: $V = \frac{3{,}000{,}000}{86{,}400} = 34.7$ m/s. This is a rough estimate but it can be seen that this is indeed similar to the results observed in Figure 7.6 (same order of magnitude). So although the difference in distance is not entirely what is expected, it is at least similar for all three measurement principles (RA/DEC, radar time-delay and radar Doppler). This test was also performed for Icarus and 2001XQ and showed a similar result. The figures for these asteroids can be found in Appendix A.

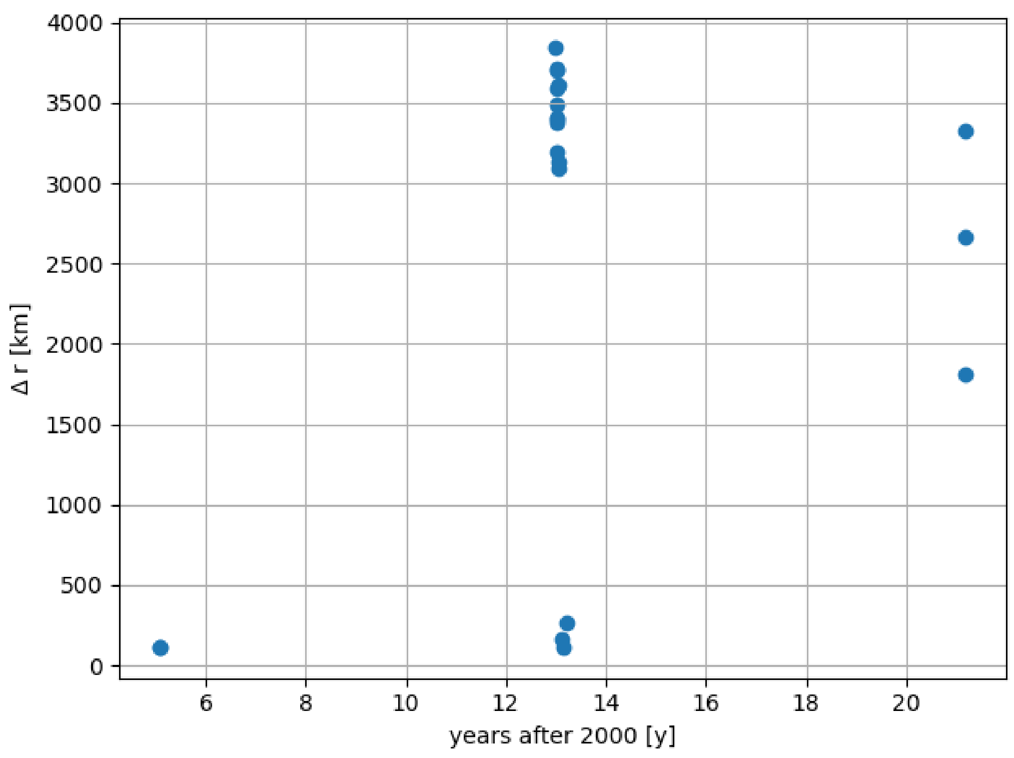


Figure 7.5: Differences between integration model and radar time-delay measurements for Apophis.

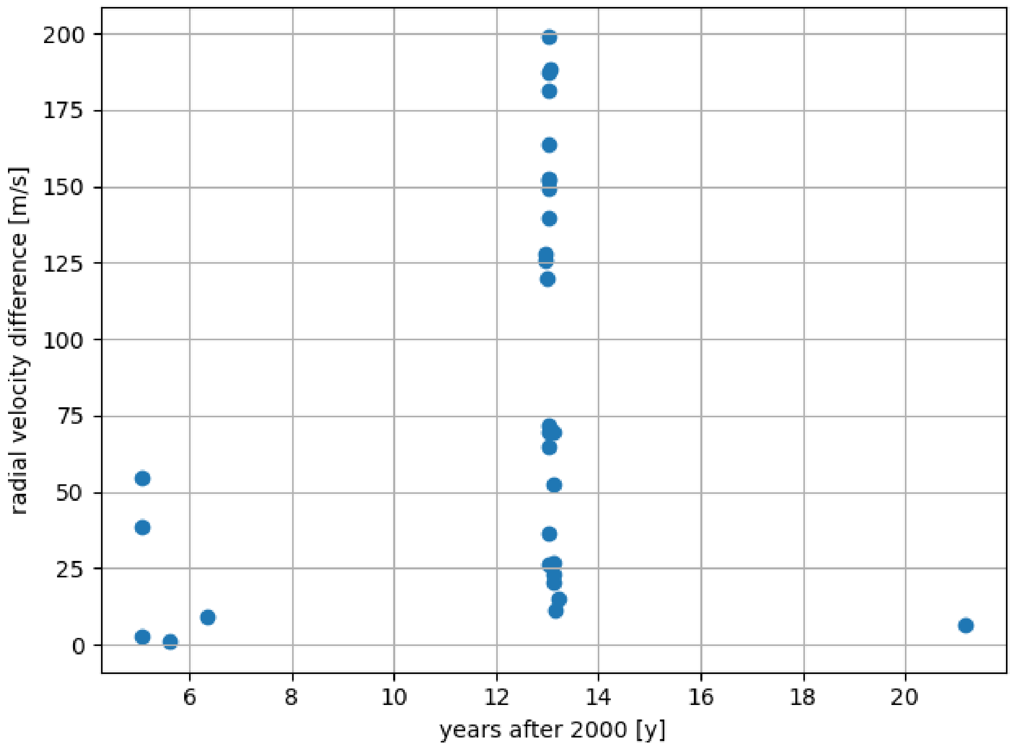


Figure 7.6: Differences between integration model and radar Doppler measurements for Apophis.

7.2.2. Discussion on observed differences

In Section 7.1 it was found that the model from Horizons compared to the model created in Tudat caused a difference of 400 km. That model has also been used to see if it behaves significantly different that the model created in Tudat. It turned out that the differences for all types of measurements were similar to what was presented in this section. On top of that, it was even a bit worse. This means that the model created in Tudat can not cause the issue of the results being not entirely perfect. The only remark that can be made regarding this is that the Horizons model does exclude some of the forces that might be necessary for the propagation of Apophis. This however has already been investigated in detail for Apophis [17]. The propagation time was set to 23 years (2006 to 2029), because in 2029 the first close encounter with Earth will occur. The Standard Dynamical Model (SDM) was used, which includes point mass gravity models of the Sun, the Moon, all the planets in the Solar System, main relativistic effects and the three largest asteroids (Ceres, Pallas and Vesta). These forces were not taken into account in the model created in Tudat, but is not causing any significant difference (only 0.6 km). Georgini [17] also mentions the accuracy level of the SDM and this resulted in residuals of around 1 arcsec, which turned out to be almost equal to the expected 100 km difference.

A reason for the obtained differences between the expected error and the computed error could be that fitting an orbit through points is a different method than back-tracking (using the known asteroid position and velocity to compute and compare a measurement data point), which was done for this validation test case. It should also be noted that although there is a difference, it is not large compared to the true RA and DEC values for example. As mentioned before, a difference of 0.02% was obtained, which is fairly close to the true value. The result found for this test case is therefore considered as acceptable and means that the integration model is now validated.

7.3. Neural network

The neural network described in Chapter 5 also needs verification and validation. It will first be validated by comparing outputs of a problem with a straightforward and understandable output using a neural network, after which the problem is used to perform analyses on verification.

7.3.1. Neural network validation

The neural network built for this project makes use of the scikit-learn Python package [38], and the example model used the Tensorflow Python package. The same model is also built in scikit-learn, to be able to validate the model created in scikit-learn. The expectation is that both methods are resulting in similar accuracy predictions, when selecting the same hyperparameters. More information on hyper-parameters and neural networks can be found in Chapter 5.

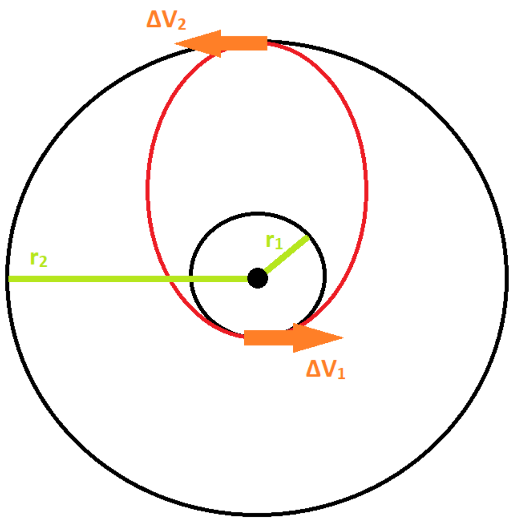


Figure 7.7: Hohmann transfer between two circular orbits.

The example problem is the determination of the required ΔV for Hohmann transfers [3]. A Hohmann transfer is the most efficient transfer between two circular orbits (see Figure 7.7). The radii of these orbits are provided as r_1 and r_2. The total ΔV is defined by the sum of ΔV_1 and ΔV_2, and both these numbers are determined using the following formulas: [48]

$$\begin{aligned} \Delta V_1 &= \sqrt{\frac{\mu}{r_1}}\left(\sqrt{\frac{2r_2}{r_1+r_2}}-1\right) \\ \Delta V_2 &= \sqrt{\frac{\mu}{r_2}}\left(1-\sqrt{\frac{2r_1}{r_1+r_2}}\right) \end{aligned} \tag{7.8}$$

where μ is the gravitational parameter of the central body, which is the Sun in this example. Besides ΔV, the Time Of Flight (TOF) can also be determined from only r_1 and r_2: [48]

$$TOF = \pi\sqrt{\frac{(r_1+r_2)^3}{8}} \tag{7.9}$$

The input file consists of 10^6 data points, ranging from 0.1 AU to 100 AU. The neural network is then trained using two inputs and three outputs. The hyperparameters are listed below: [3]

- **Hidden layer sizes:** (64, 64)
- **Activation:** ReLu
- **Solver:** ADAM
- **Batch size:** 100
- **Learning rate:** 0.001 (constant)
- **Max iter:** 100
- **Val fraction:** 0.2
- **# iters to stop if no improvement is found in val loss:** 20
- **Training/test data split:** 0.05 (=> 50,000 data points)

The only hyperparameter that is different in the two models is the metric that is used for training. Scikit-learn uses MSE by default, whereas this is MAE in Tensorflow. This neural network is trained using both the scikit-learn and the Tensorflow packages and the result can be found in Figures 7.8-7.11. It should be noted that the hyperparameters used were only selected to end up with a similar model performance and the model is not necessarily tuned. This means that the expectation is that the models have a similar performance, but not necessarily perfect (predicted values equal to the true value).

The figures show that the ΔV is more difficult to predict and not perfect, as expected earlier due to the selection of hyperparameters. The figures also show that the results are almost equal visually. When comparing actual values this can also be concluded, since the standard deviation of the machine-learning error is more or less equal (see Table 7.3). Scikit-learn seems to be slightly better but this could be the case due to the fact that a different scoring method was used during fitting, as said earlier. It is therefore assumed that these differences are small enough to be able to conclude that the neural network is fully validated.

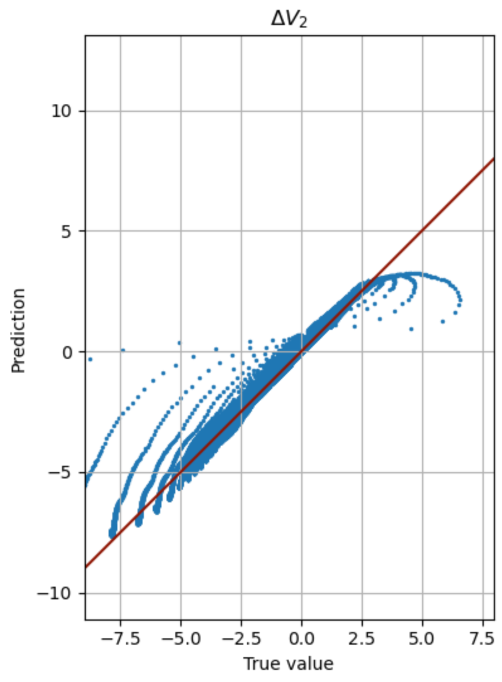


Figure 7.8: Predicted vs true values of ΔV_2 of the standard Hohmann transfer using the **scikit-learn** package.

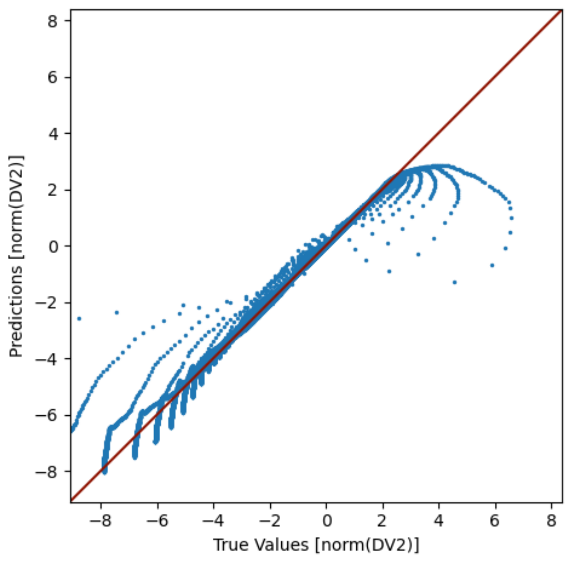


Figure 7.9: Predicted vs true values of ΔV_2 of the standard Hohmann transfer using the **Tensorflow** package [3].

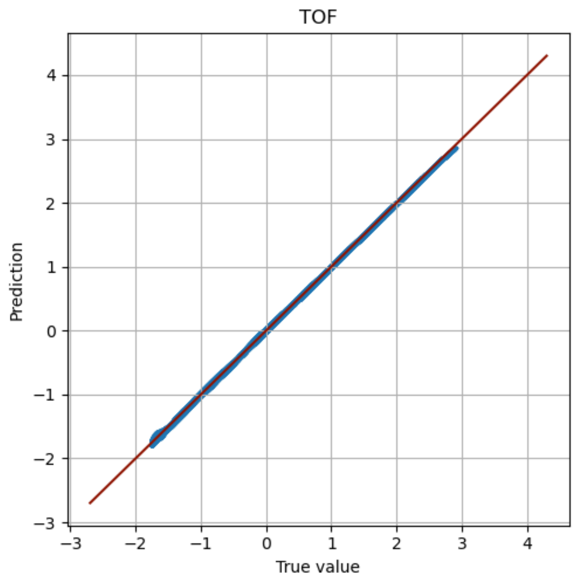


Figure 7.10: Predicted vs true values of TOF of the standard Hohmann transfer using the **scikit-learn** package.

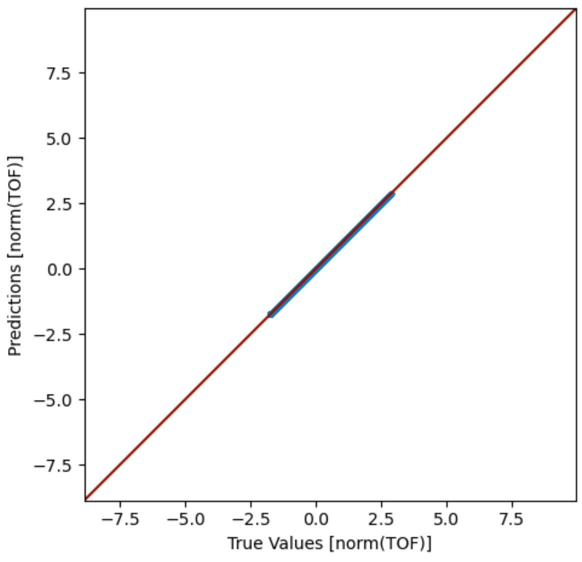


Figure 7.11: Predicted vs true values of TOF of the standard Hohmann transfer using the **Tensorflow** package [3].

Table 7.3: Difference between Tensorflow and scikit-learn when comparing standard deviations.

	Tensorflow [3]	Scikit-learn	Difference
1-σ ΔV_1	0.18	0.17	0.01 (5.5%)
1-σ ΔV_2	0.22	0.19	0.03 (14%)
1-σ TOF	0.033	0.018	0.015 (45%)

7.3.2. Neural network verification

The validation in the previous subsection only showed that the results are reliable. However, verification still is required to test extreme cases, in order to investigate whether the behaviour of the neural network is as expected.

The first test case is a test case with a very limited number of data points. The expectation is that the neural network performs significantly worse, as there is not enough data to capture all the aspects of the behaviour and overfitting can occur. Figures 7.12 and 7.13 show the results of the first test case.

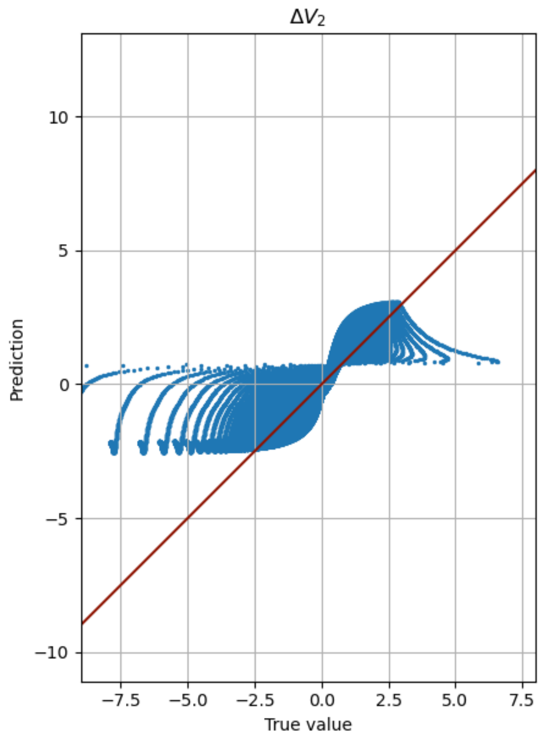


Figure 7.12: Predicted vs true values of ΔV_2 of the standard Hohmann transfer using a sample size of 50.

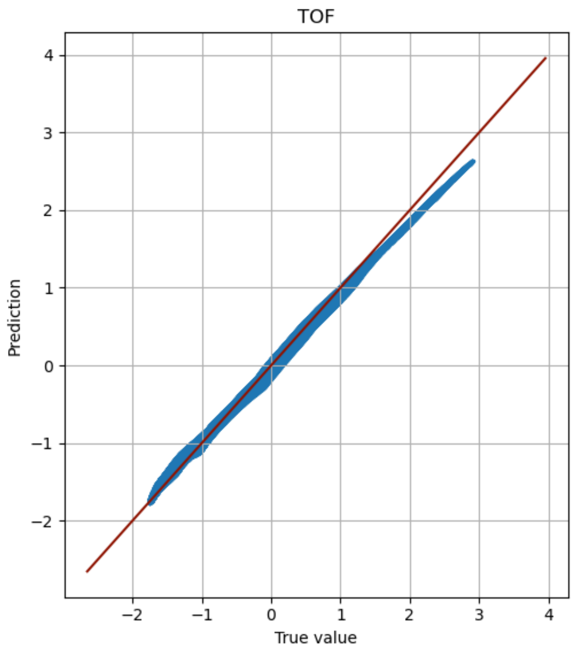


Figure 7.13: Predicted vs true values of TOF of the standard Hohmann transfer using a sample size of 50.

From the figures it can be concluded that the result is definitely worse, and this can also be concluded from the normalized standard deviations (σ ΔV_2 = 0.67 (+ 0.48) and σ TOF = 0.073 (+0.055)), which are significantly larger.

A second test considers the impact of the learning rate, seen as one of the most important hyperparameters of a neural network [18]. A learning rate that is too large may cause the neural network to converge to a sub-optimum. A learning rate that is too small takes first of all much longer and can also get stuck at not the most optimal point [6], but this does not necessarily has to be the case. To test this behaviour first the learning rate is increased to 0.1 and after to 0.00001. The result of these tests can be found in Appendix C. The standard deviations are as follows: (σ ΔV_2 = 0.41 (+ 0.22) and TOF = 0.13 (+0.11)) for a learning rate of 0.1 and (σ ΔV_2 = 0.39 (+ 0.20) and σ TOF = 0.011 (-0.007)) for a learning rate of 0.00001, and this shows that again as expected worse results on average are found. A smaller learning rate is not necessarily worse, as also already mentioned earlier.

This concludes the verification part of the neural network, showing that the neural network and its implementation perform according to the expectations. Other extreme cases are already taken care of by scikit-learn, for example when a batch size larger than the sample size is chosen, the code throws an error. These kind of cases are all assumed to be already verified and validated by scikit-learn.

8 Conclusions and recommendations

The conclusions and recommendations of this thesis project are addressed in this chapter.

8.1. Conclusions

The goal of the thesis project was to find a more efficient way of orbit and uncertainty propagation for PHAs. Machine learning was considered to obtain this goal. First of all asteroid data was acquired from two databases (JPL and MPC), which included asteroid states, uncertainties and physical properties. It also included measurement data, which was used to validate the asteroid state and uncertainties. The database data was used to propagate PHAs using a propagation model created in Tudat. Tudat provided the force models and relevant information of planets, such as the ephemerides. PHA data was stored in a file, containing the initial state, the radius and uncertainties of each PHA, as well as the integration time as inputs. The outputs were also stored in the same file, consisting of the x,y and z-position of the PHA after the set integration time and the error caused by the uncertainties in the state and the dynamics model. This leads to 15 inputs and 4 outputs in total.

The data file coming from the propagation was used as an input to the neural network, created using the scikit-learn package in Python. Many different options were considered for the estimation of the asteroid position and the error caused by the uncertainties in the state and the dynamics model. First of all a small neural network was setup using only a fraction of the total number of PHAs (100 out of the total of 2241). Hyperparameters were extracted from similar astrodynamics problems. The results of this small network were used to set the values of the hyperparameters of a neural network containing the information of all PHAs. The hyperparameters of this network were also tuned for three different cases. A selection of different input and output variables defined these cases. The first case used Kepler elements as the input for machine learning, whereas these were Cartesian elements in the second case. The third case adds input variables containing information of the initial position of Earth, making it possible to reduce the number of outputs to two (PHA - Earth distance and the error caused by the uncertainties in the state and the dynamics model). Case 1 performed the best out of the three, resulting in an R^2 score of 0.96.

The score was then put in perspective, by calculating the actual error caused by the neural network, for both the Earth distance and the error caused by the uncertainties in the state and the dynamics model. Ephemeris data of Earth was used to calculate the Earth distance from the PHA position, since this was one of the outputs of the neural network. The results show a machine-learning error in Earth distance with a mean of $3.42*10^7$ km and a standard deviation of $3.83*10^7$ km. The machine-learning error in the error caused by the uncertainties in the state and the dynamics model has a mean of $1.17*10^6$ km and a standard deviation of $4.65*10^6$ km. The standard deviations were larger than the mean values and also from visualisations it was concluded that this result can not be used to make reasonable estimations. The result was investigated for many other tests, such as the investigation of the batch size, the effect of data points per asteroid, the current effect of under- or overfitting, the importance of the input variables, separate outputs, a test with more focus on close approach data and a different scaling method. Most of these investigations did not lead to a significantly better result, except for the different scaling method.

The different scaling method used a log scale on the output for the estimation of the error caused by the uncertainties in the state and the dynamics model, as a separate output. Estimating the error caused by the uncertainties in the state and the dynamics model separately was also necessary to make sure that the performance of the model was not influenced by the prediction of the Earth distance. This result was significantly better compared to the first result and the hyperparameters of this result were therefore tuned again and resulted in a mean error of 27.7% and a standard deviation of 51.7%. These percentages were introduced due to the log scale and are defined by dividing the machine-learning error by the true value of the error caused by the uncertainties in the state and the dynamics model. The standard deviation was found to be sensitive to outliers, since it decreased to 41.8% when 3 out of 448 PHAs were removed from the test set. Furthermore, 84% of the test data has a machine-learning error of 45% or lower, which is much better than a perfect normal distribution (80% for 84% of the test data). The test data was also fitted by a normal distribution and this showed a standard deviation of 33.6%. These results show that the calculated standard deviation of 51.7% does not fully represent the error distribution and that the result is actually better. The result is also summarized in Table 8.1.

Table 8.1: Model result.

Data points	112,050
Hidden layer sizes	(128, 128, 128)
Learning rate	0.001 (Adaptive)
Solver	ADAM
Activation function	ReLU
Scaler	Standard / Log
Error - mean [%]	27.7
Error - standard deviation [%]	51.7

It is thus concluded that machine learning can be used for the estimation of the error caused by the uncertainties in the state and the dynamics model of PHAs. The PHA position could not be estimated with a realistic accuracy, and therefore the nominal trajectory still requires a propagation model. This however still saves a significant amount of time, since a Monte-Carlo simulation is typically used for the calculation of the error caused by the uncertainties in the state and the dynamics model, which required 50 propagations. Thus, the machine-learning model reduces the computational speed by a factor of 50, and this can even be more since sometimes more propagations are required to estimate the uncertainty error. The accuracy is of course worse, but still acceptable to be able to draw reasonable conclusions on the probability of an Earth impact.

The research question of this thesis project is: To what extent is machine learning able to improve orbit and uncertainty propagation of asteroids, in terms of both accuracy and computational speed? The drawn conclusions can be used to answer the research question: machine learning is suitable and can indeed be used to improve uncertainty propagation. The machine-learning technique showed a significant decrease in computational expensiveness, whereas the accuracy was still reasonable. This is better compared to current techniques, as these always require more computation time. Numerical propagation is required to propagate PHAs with a reasonable accuracy and analytical techniques only showed a purpose in the verification of the propagation model. A Kepler orbit showed to have a higher accuracy compared to a prediction made for the PHA position, showing that the nominal trajectory can not be estimated with machine learning in this way. This might be doable if the machine-learning technique is used as part as a hybrid propagation technique, but due to time constraints this was not investigated. Optimization techniques were considered for the tuning process of the neural network, but were discarded since some parameters were categorical, making it more difficult to implement such technique.

8.2. Recommendations

The result contains a model that is capable of making proper estimations on the error caused by the uncertainties in the state and the dynamics model. There are however alternatives that could be investigated in order to improve the current result. One of these options is to use the measurement data of PHAs directly in the neural network. This increases the complexity of the model, but if the neural network is capable of finding a reasonable solution, it would result in a model with a better representation of reality, as the measurement data directly reflects reality and the orbit fit is only a representation of reality.

Another option is to gather more data about close approaches of PHAs. For this it is necessary to propagate each PHA just before a close approach to be able to store this more relevant data. This would take significantly more time, since this differs for every PHA, might cause issues for not having the same starting time and the Tudat software is not capable of propagating orbits after 2040, meaning that a different software should be used to propagate all PHAs. This data can be the solution to predict the PHA position for close approaches with more accuracy.

A different option would be to investigate the usefulness of machine learning as part of a hybrid propagation technique. This needs to be carefully executed, since only a portion is covered by the neural network, meaning that the computational speed is already worse compared to a stand-alone model, by definition. It should thus only be used in cases where a stand-alone model is not accurate enough, which is the case for the prediction of the nominal trajectory and therefore would be interesting if this is investigated as well.

The obtained result can also serve as an introduction for different purposes, such as the uncertainty propagation of an orbit around Earth. Knowing that the method can work makes it more worthwhile investigating similar astrodynamics problems. The recommendations are also summarized in Table 8.2.

Table 8.2: Recommendations.

Recommendation	Purpose
Use measurement data in neural network	A more representable solution
More close approach data	Better predictions for the PHA position in close approach scenarios
Machine learning as part of a hybrid propagation technique for the estimation of the nominal trajectory	A more accurate estimation of the nominal trajectory (with a slight increase in computational speed)
Machine learning applied to similar purposes	Obtaining a faster propagation technique

References

[1] 3Blue1Brown. *But what is a neural network? | Chapter 1, Deep learning*. 2017. URL: https://www.youtube.com/watch?v=aircAruvnKk (visited on 09/12/2021).

[2] 3Blue1Brown. *Gradient descent, how neural networks learn | Chapter 2, Deep learning*. 2017. URL: https://www.youtube.com/watch?v=IHZwWFHWa-w (visited on 09/19/2022).

[3] J. Achterberg et al. “Hohmann transfer trajectory design based on artificial neural networks; Special Topics in Astrodynamics (AE4889) - Phase 1”. Delft University of Technology. Mar. 2021.

[4] J. Achterberg. “Orbit and uncertainty propagation methods: Applicability of methods on different orbit types.” Delft University of Technology. 2021.

[5] P. Baheti. *Activation Functions in Neural Networks [12 Types & Use Cases]*. 2022. URL: https://www.v7labs.com/blog/neural-networks-activation-functions#:~:text=Try%5C%20V7%5C%20Now-,What%5C%20is%5C%20a%5C%20Neural%5C%20Network%5C%20Activation%5C%20Function%5C%3F,prediction%5C%20using%5C%20simpler%5C%20mathematical%5C%20operations. (visited on 06/30/2022).

[6] J. Brownlee. *Understand the Impact of Learning Rate on Neural Network Performance*. 2019. URL: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/ (visited on 06/16/2022).

[7] B. Carry. “Density of asteroids”. In: *Planetary and Space Science* 73 (2012), pp. 98–118.

[8] Minor Planet Center. *Uncertainty Parameter U*. 2022. URL: https://minorplanetcenter.net//iau/info/UValue.html (visited on 06/18/2022).

[9] Lei Chen et al. “Orbital Prediction Error Propagation of Space Objects”. In: *Orbital Data Applications for Space Objects: Conjunction Assessment and Situation Analysis*. Singapore: Springer Singapore, 2017, pp. 23–75. ISBN: 978-981-10-2963-9. DOI: 10.1007/978-981-10-2963-9_2. URL: https://doi.org/10.1007/978-981-10-2963-9_2.

[10] P. Chodas. *Sentry: Earth Impact Monitoring*. 2022. URL: https://cneos.jpl.nasa.gov/sentry/intro.html (visited on 09/08/2022).

[11] Dr. ir. D. Dirkx and Dr. ir. E. Mooij. *Propagation and Optimisation in Astrodynamics (AE4866)*. Lecture slides. Delft University of Technology, 2021.

[12] Delft University of Technology. *Tudat Space: Documentation*. 2020. URL: https://tudat-space.readthedocs.io/en/latest/index.html (visited on 07/02/2021).

[13] C. Englert et al. “Optical orbital debris spotter”. In: *Acta Astronautica* 104 (Nov. 2014), pp. 99–105. DOI: 10.1016/j.actaastro.2014.07.031.

[14] ESA. *A 2021 Guide to improving CNNs-Optimizers: Adam vs SGD*. 2022. URL: https://www.esa.int/About_Us/ESAC/Space_Situational_Awareness_-_SSA (visited on 07/05/2022).

[15] J.A. Fernández et al. “Assessing the physical nature of near-Earth asteroid sthrough their dynamical histories”. In: *Earth and Planetary Astrophysics* 1 (2014), pp. 1–28.

[16] W. M. Folkner et al. “The Planetary and Lunar Ephemerides DE430 and DE431”. In: *IPN Progress Report, JPL, NASA* 1 (2014), pp. 42–196.

[17] J.D. Giorgini et al. “Predicting the Earth encounters of (99942) Apophis”. In: *Icarus* 193 (2008), pp. 1–19.

[18] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. 1st ed. Cambridge, United States: Massachusetts Institute of Technology (MIT), 2016.

[19] N.W. Harris and D.W. Hughes. “Asteroid-Earth collision velocities”. In: *Planetary and Space Science* 42.4 (1994). Special Issue: Asteroids, Comets and Meteors 1993-11, pp. 285–289. ISSN: 0032-0633. DOI: https://doi.org/10.1016/0032-0633(94)90098-1. URL: https://www.sciencedirect.com/science/article/pii/0032063394900981.

[20] B. Iannota. *A 2021 Guide to improving CNNs-Optimizers: Adam vs SGD*. 2009. URL: https://www.space.com/5542-satellite-destroyed-space-collision.html (visited on 07/05/2022).

[21] D. Izzo, C. Sprague, and D. Tailor. “Machine learning and evolutionary techniques in interplanetary trajectory design”. In: *CoRR* abs/1802.00180 (Sept. 2018), pp. 1–19.

[22] A. Khutar. *What’s the Optimal Batch Size to Train a Neural Network?* 2022. URL: https://wandb.ai/ayush-thakur/dl-question-bank/reports/What-s-the-Optimal-Batch-Size-to-Train-a-Neural-Network---VmlldzoyMDkyNDU (visited on 08/15/2022).

[23] A. Kumar. *Mean Squared Error or R-Squared – Which one to use?* 2022. URL: https://vitalflux.com/mean-square-error-r-squared-which-one-to-use/#:~:text=value%5C%20of%5C%20R2.-,Difference%5C%20between%5C%20Mean%5C%20Square%5C%20Error%5C%20%5C%26%5C%20R%5C%2DSquared,data%5C%20is%5C%20scaled%5C%20or%5C%20not. (visited on 06/13/2022).

[24] A. Kumar. *MinMaxScaler vs StandardScaler – Python Examples*. 2020. URL: https://vitalflux.com/minmaxscaler-standardscaler-python-examples/#:~:text=The%5C%20MinMaxscaler%5C%20is%5C%20a%5C%20type,range%5C%20from%5C%20min%5C%20to%5C%20max. (visited on 06/30/2022).

[25] S. Lei et al. “How training data affect the accuracy and robustness of neural networks for image classification”. In: *ICLR* (2018).

[26] E.G. Lemoine et al. “An improved solution of the gravity field of Mars (GMM-2B) from Mars Global Surveyor”. In: *Journal of Geophysical Research* 106 (2001), pp. 23359–23376.

[27] Z. Li and M. Ziebart. “Uncertainty analysis on direct solar radiation pressure modelling for GPS IIR and Galileo FOC satellites”. In: *University College London* 66 (2020), pp. 963–973.

[28] J.J. Lissauer and I. de Pater. *Fundamental Planetary Science - Physics, Chemistry and Habitability*. 2nd ed. Cambridge, United Kingdom: Cambridge Univiersity Press, 2019.

[29] D.C Liu and J. Nocedal. “On the limited memory BFGS method for large scale optimization”. In: *Mathematical Programming* 45 (1989), pp. 503–528.

[30] Minor Planet Center. *Small-Body Database Radar Observations*. 2022. URL: http://mpcweb1.cfa.harvard.edu/iau/ECS/MPCAT-OBS/MPCAT-OBS.html (visited on 02/26/2022).

[31] T.G. Müller et al. “Thermal infrared observations of asteroid (99942) Apophis with Herschel□”. In: *Astronomy and Astrophysics* 566 (2014), pp. 1–10.

[32] NASA. *Horizons Web-Interface*. 2021. URL: https://ssd.jpl.nasa.gov/horizons.cgi (visited on 07/13/2021).

[33] NASA. *Small-Body Database Query*. 2022. URL: https://ssd.jpl.nasa.gov/tools/sbdb_query.html (visited on 02/26/2022).

[34] NASA. *Small-Body Database Radar Observations*. 2022. URL: https://ssd.jpl.nasa.gov/sb/radar.html (visited on 02/26/2022).

[35] R. Nave. *Right Ascension and Declination*. 2000. URL: http://hyperphysics.phy-astr.gsu.edu/hbase/eclip.html (visited on 04/19/2022).

[36] M. Paine and B. Peiser. “The Frequency and Consequences of Cosmic Impacts Since the Demise of the Dinosaurs”. In: *Bioastronomy 2002: Life Among the Stars* (July 2002), pp. 1–13.

[37] S. Park. *A 2021 Guide to improving CNNs-Optimizers: Adam vs SGD*. 2021. URL: https://medium.com/geekculture/a-2021-guide-to-improving-cnns-optimizers-adam-vs-sgd-495848ac6008 (visited on 06/30/2022).

[38] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[39] F. Pedregosa et al. *Scikit-learn: Machine Learning in Python; 3.3. Metrics and scoring: quantifying the quality of predictions*. 2022. URL: https://scikit-learn.org/stable/modules/model_evaluation.html#regression-metrics (visited on 06/13/2022).

[40] Pixabay. *Asteroid flybys are common, and we usually can see them coming.* 2019. URL: https://pixabay.com/illustrations/asteroid-land-planet-space-4145080/ (visited on 05/12/2022).

[41] C.M. Rumpf, H.G. Lewis, and P.M. Atkinson. “Asteroid impact effects and their immediate hazards for human populations”. In: *Geophysical Research Letters* 44.8 (Apr. 2017), pp. 3433–3440. DOI: 10.1002/2017gl073191. URL: https://doi.org/10.1002%2F2017gl073191.

[42] J.F. San-Juan et al. “Hybrid SGP4 orbit propagator”. In: *Acta Astronautica* 137 (2017), pp. 254–260.

[43] J.F. San-Juan et al. “Hybrid SGP4 propagator based on machine-learning techniques applied to GALILEO-type orbits”. In: *International Astronautical Federation* 1.2 (2018), pp. 1–11.

[44] Carlos Sánchez-Sánchez and Dario Izzo. “Real-Time Optimal Control via Deep Neural Networks: Study on Landing Problems”. In: *Journal of Guidance, Control, and Dynamics* 41 (Oct. 2016), pp. 1–35. DOI: 10.2514/1.G002357.

[45] R. Silipo. *Machine Learning Algorithms and The Art of Hyperparameter Selection; A review of four optimization strategies.* 2020. URL: https://towardsdatascience.com/machine-learning-algorithms-and-the-art-of-hyperparameter-selection-279d3b04c281 (visited on 08/30/2022).

[46] V. Trevisan. *Comparing Robustness of MAE, MSE and RMSE*. 2022. URL: https://towardsdatascience.com/comparing-robustness-of-mae-mse-and-rmse-6d69da870828 (visited on 06/13/2022).

[47] S.E. Urban and K Seidelmann. *Explanatory Supplement to the Astronomical Almanac.* 3rd ed. California, United States: University Science Books., 1992.

[48] K.F. Wakker. *Fundamentals of Astrodynamics*. 1st ed. Delft, The Netherlands: Institutional Repository Library Delft University of Technology, 2015.

[49] J.R. Wertz et al. *Orbit & Constellation Design & Management*. 2nd ed. Hawthorne, California and New York, United States: Microcosm Press & Springer, Space Technology Library, 2001.

[50] D.R. Williams. *Asteroid Fact Sheet: Information on Selected Asteroids*. 2019. URL: https://nssdc.gsfc.nasa.gov/planetary/factsheet/asteroidfact.html (visited on 03/30/2022).

[51] P.R. Winters. “Forecasting sales by exponentially weighted moving averages”. In: *Management Science* 6 (1960), pp. 324–343.

[52] S. Wu. *3 Best metrics to evaluate Regression Model?* 2020. URL: https://towardsdatascience.com/what-are-the-best-metrics-to-evaluate-your-regression-model-418ca481755b (visited on 06/24/2022).

A
Extended data verification

Data of object Apophis was verified in Chapter 7. Other asteroids were also tested using the same inputs to see whether the results obtained were a coincidence for that asteroid or not. The expectation and desired result is that the verification results are similar for each asteroid. 1566 Icarus and 2008 HS3 were tested and the difference in all three observation types are shown in the figures below:

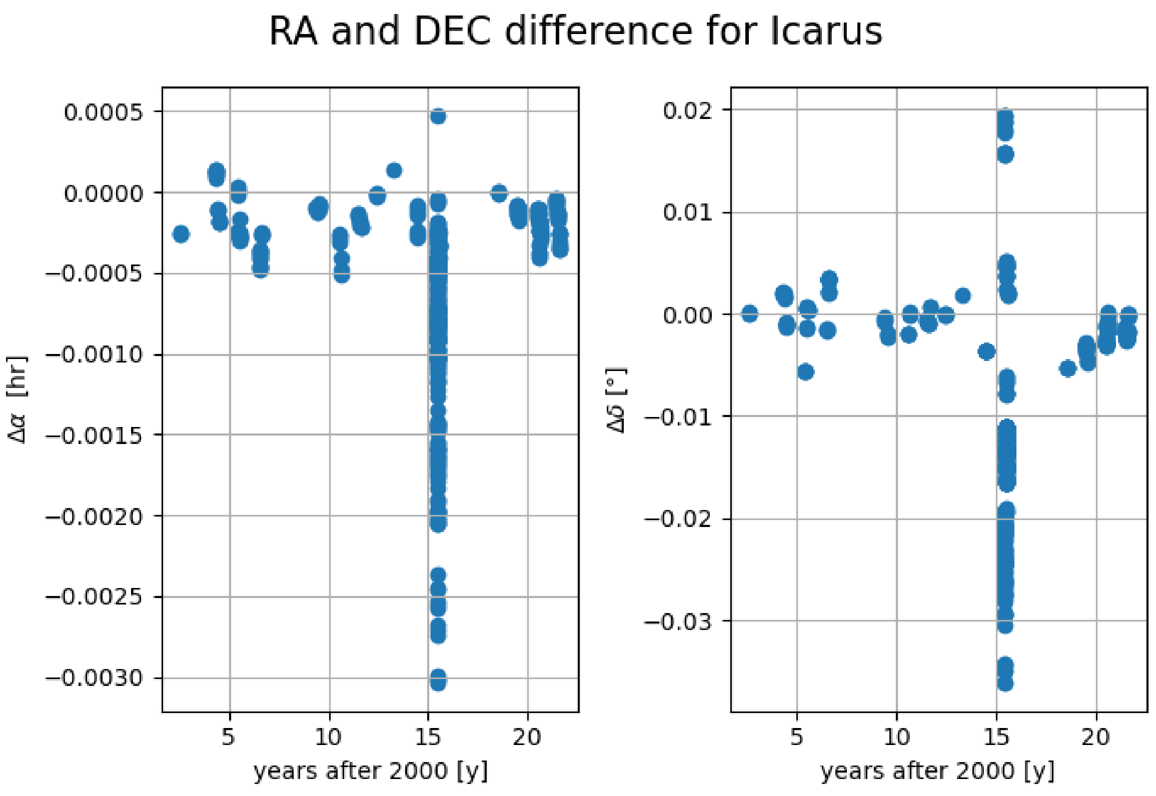


Figure A.1: ICARUS: Difference in RA and DEC between the measurements and the integration model. RA is given in difference in hour angles and DEC in difference in degrees.

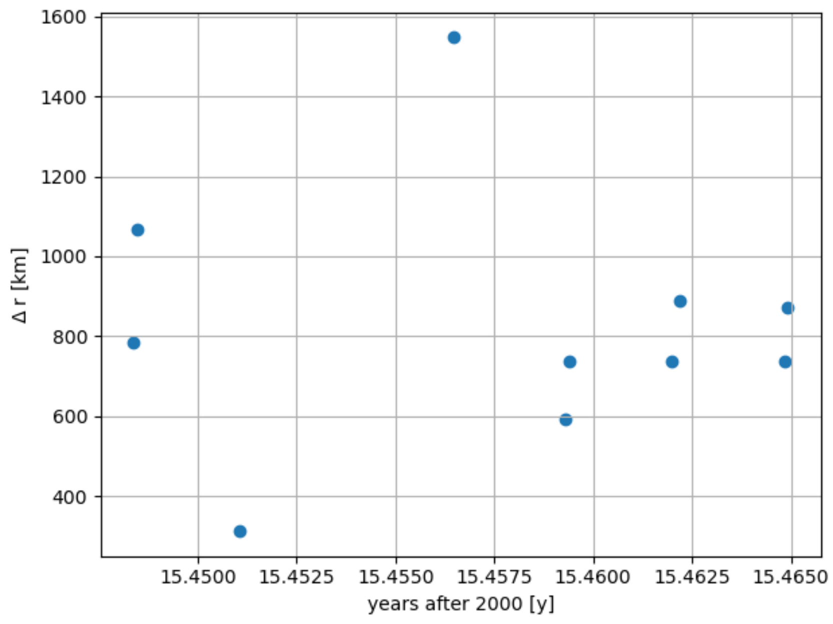


Figure A.2: ICARUS: Differences between integration model and radar time-delay measurements.

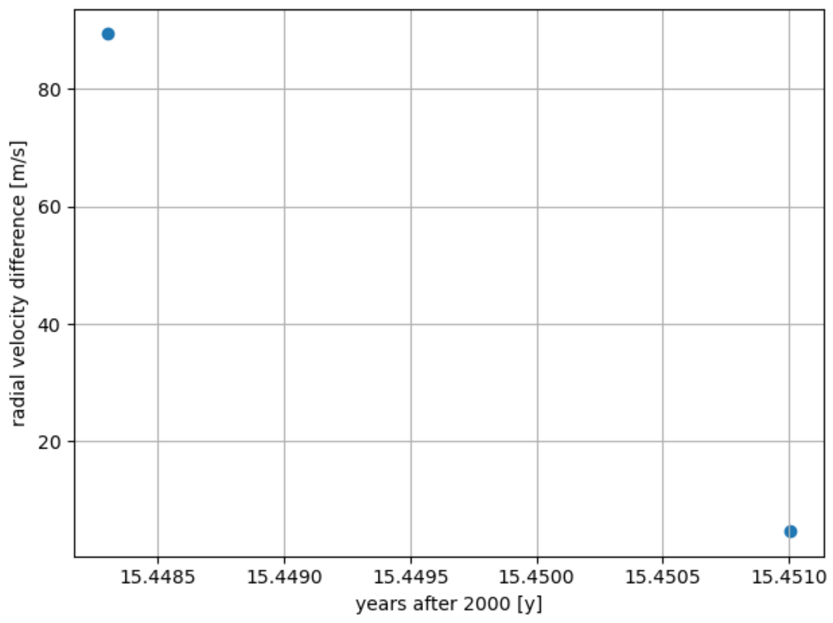


Figure A.3: ICARUS: Differences between integration model and radar Doppler measurements.

For Icarus it can be seen that the error is in the same order of magnitude as what was observed for Apophis (see Chapter 7 for the results for Apophis). It is therefore concluded that similar results are obtained for Icarus.

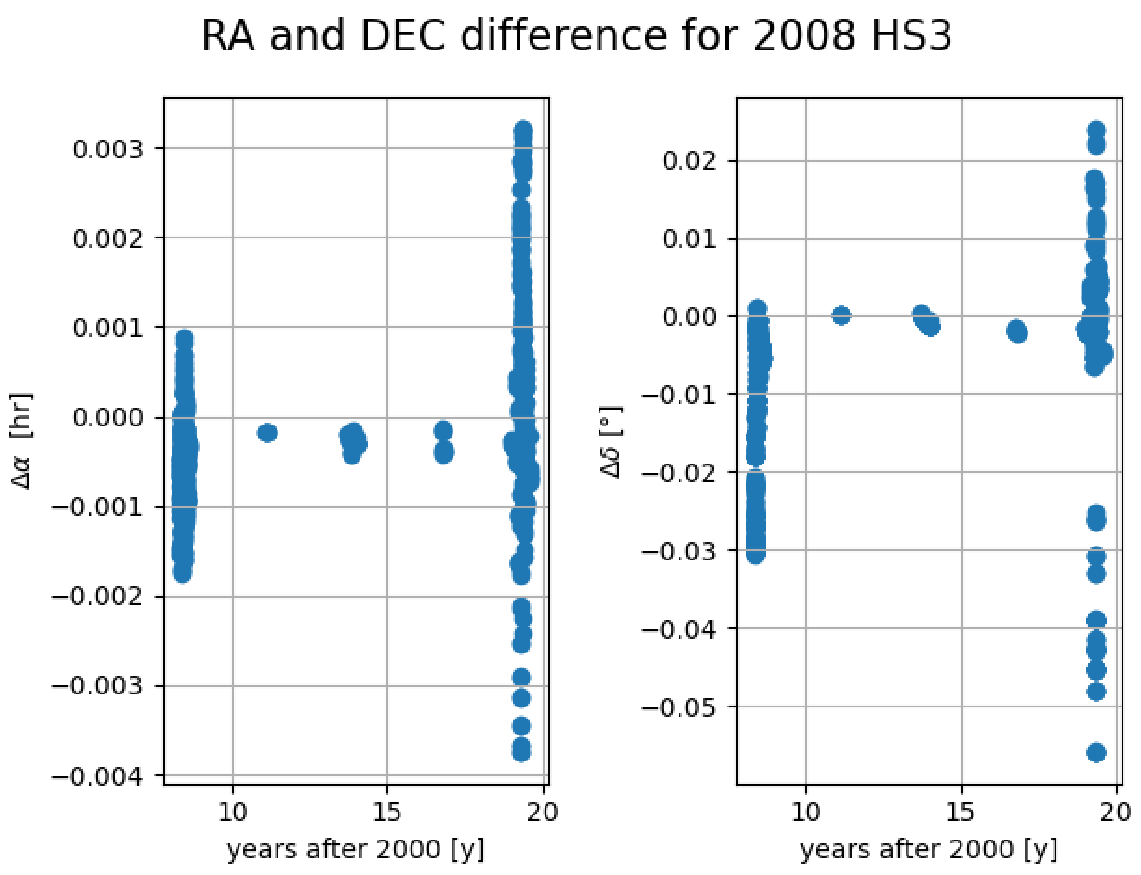


Figure A.4: 2008HS3: Difference in RA and DEC between the measurements and the integration model. RA is given in difference in hour angles and DEC in difference in degrees.

For (2008 HS3), an unnamed asteroid, no radar Doppler measurements are available and only one radar time-delay measurement, providing a distance difference of 345 km. The reason of choosing such an asteroid is to see if similar results are also obtained for asteroids with lacking data. Figure A.4 shows that this is indeed the case for the RA and DEC measurements. It is therefore concluded that similar results are indeed obtained for all asteroids.

B
Linearity initial state uncertainties

The initial state uncertainties related to the maximum position error are shown in the figures below. It can be seen that the effect is almost linear and that no strange behaviour occurs, showing that a MC analysis of these parameters is a valid approach.

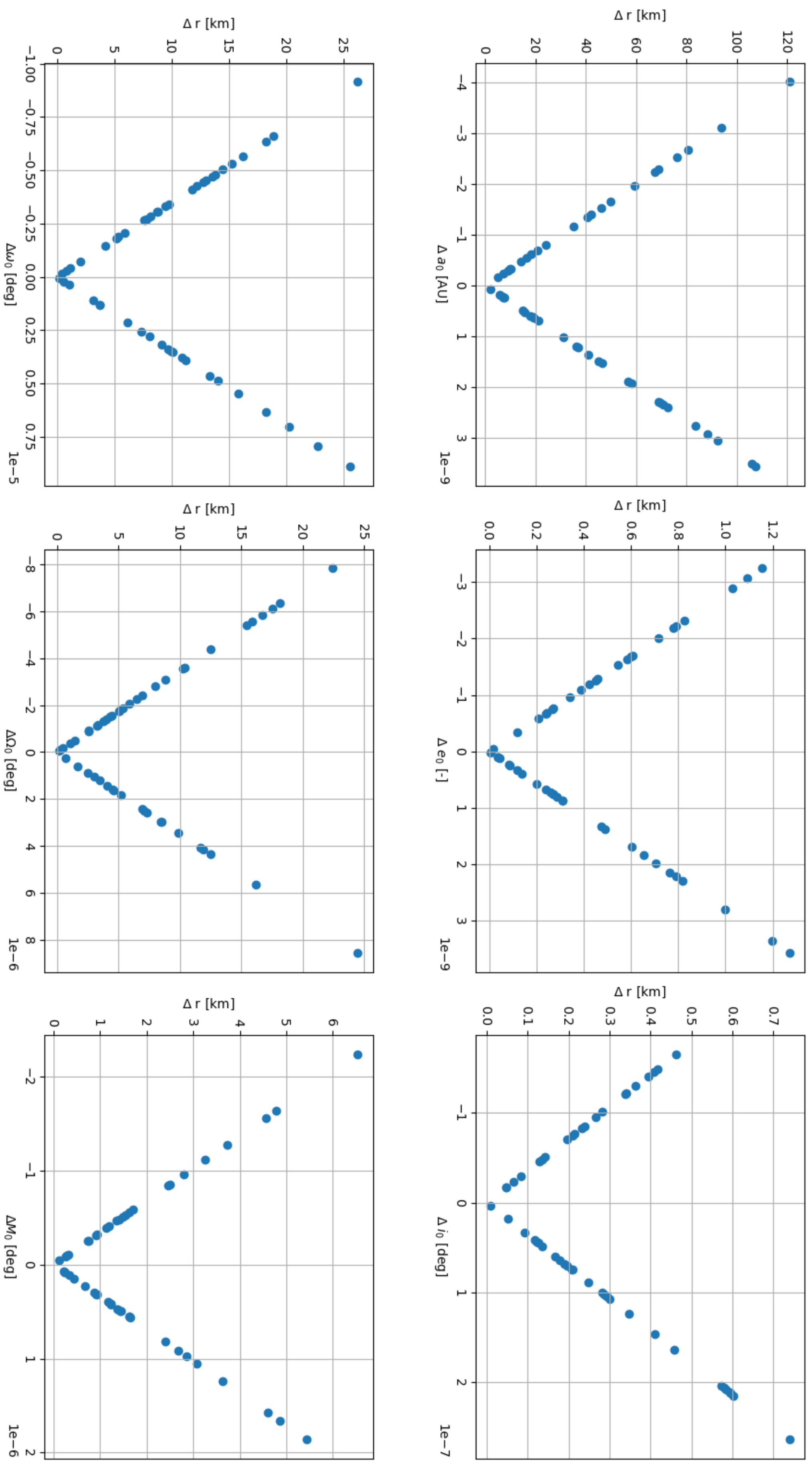


Figure B.1: Effect of initial state parameters of Apophis. Every subfigure shows solely the effect of one parameter, e.g. the first subfigure shows what the effect a difference in semi-major axis (a) has on the nominal trajectory (max Δr).

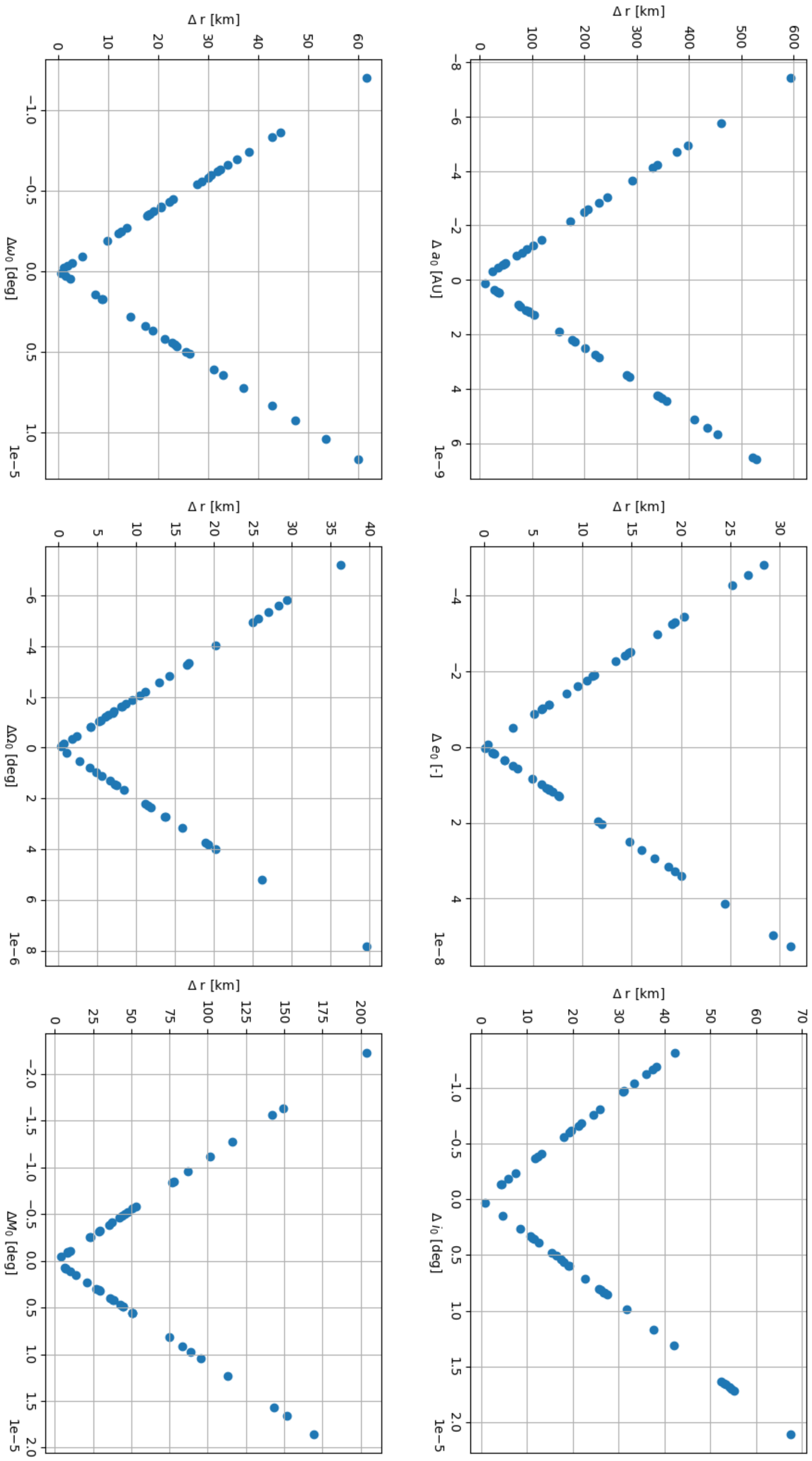


Figure B.2: Effect of initial state parameters of Icarus. Every subfigure shows solely the effect of one parameter, e.g. the first subfigure shows what the effect a difference in semi-major axis (a) has on the nominal trajectory (max Δr).

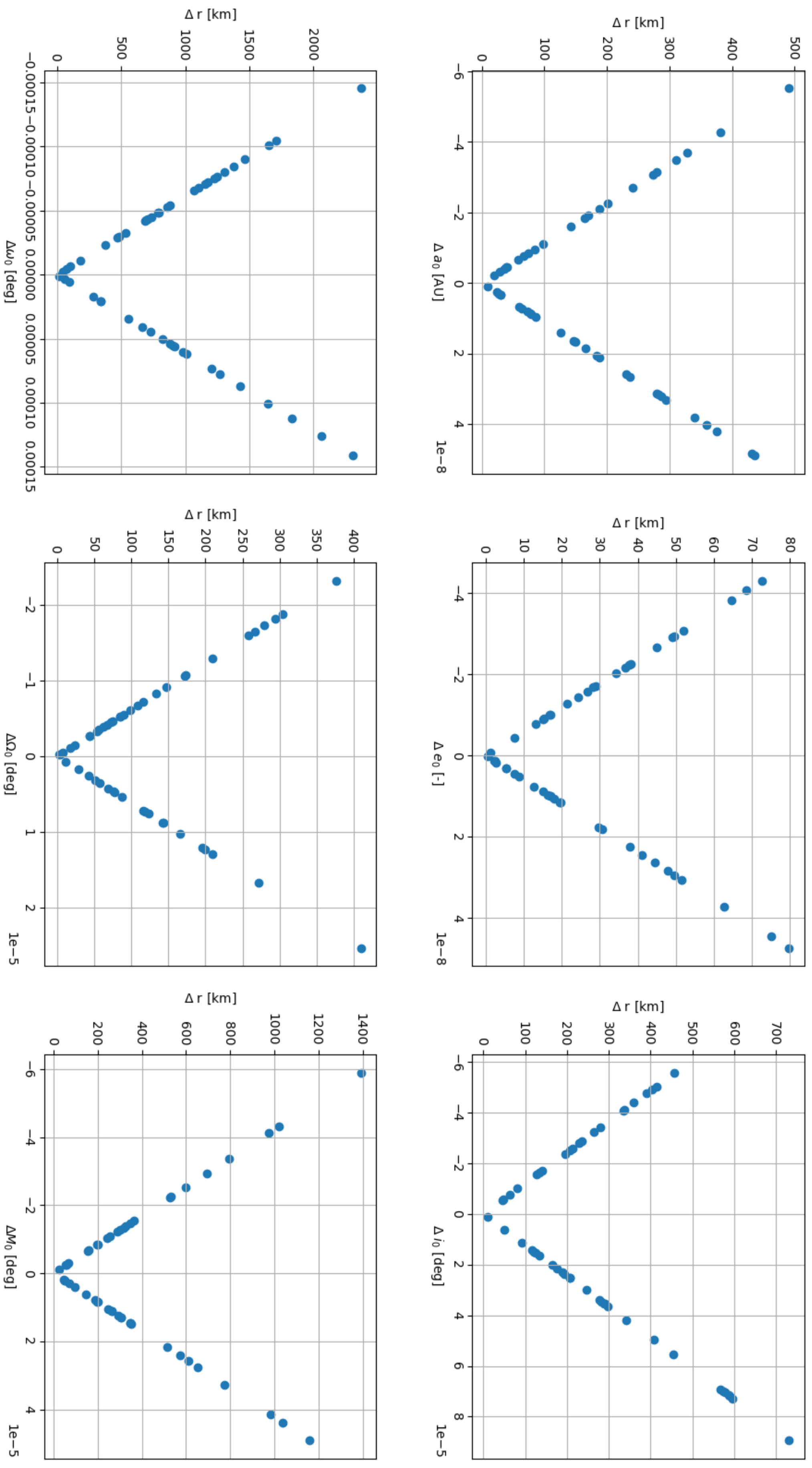


Figure B.3: Effect of initial state parameters of (2001 XQ). Every subfigure shows solely the effect of one parameter, e.g. the first subfigure shows what the effect a difference in semi-major axis (a) has on the nominal trajectory (max Δr).

C
Extended neural network verification

The neural network was verified and validated in Chapter 7. The results for some specific verfication cases are shown here, with first of all the result of the increase and decrease of the learning rate:

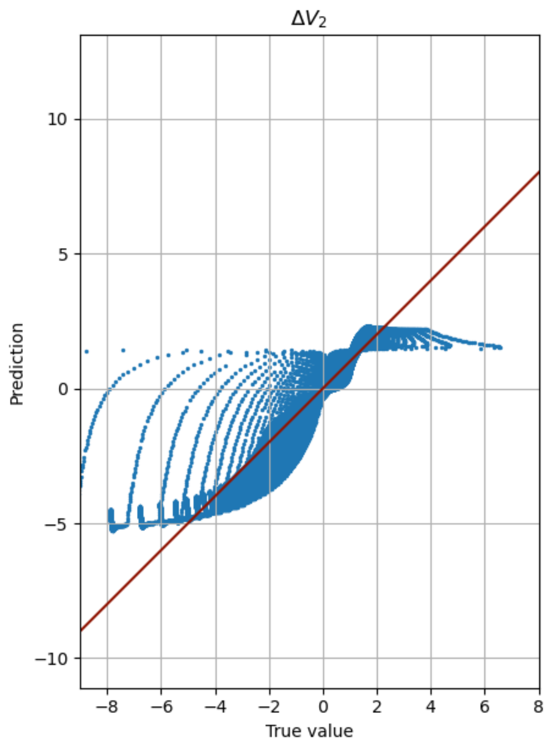


Figure C.1: Predicted vs true values of ΔV_2 of the standard Hohmann transfer using a learning rate of 0.1.

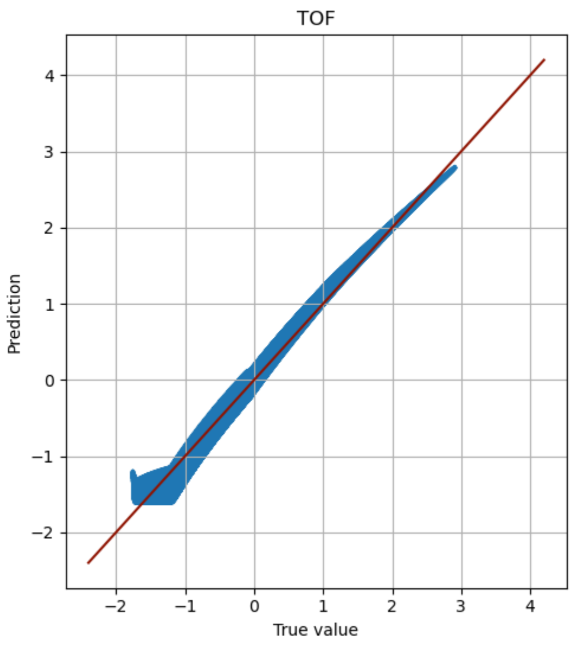


Figure C.2: Predicted vs true values of TOF of the standard Hohmann transfer using a learning rate of 0.1.

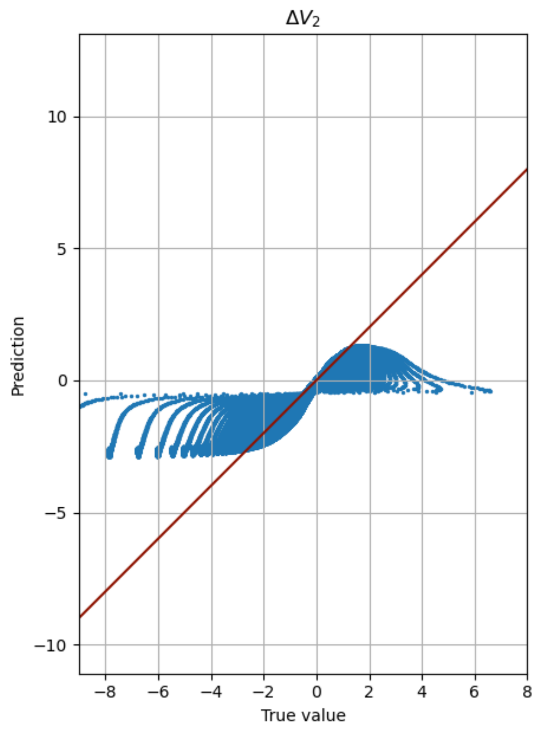


Figure C.3: Predicted vs true values of ΔV_2 of the standard Hohmann transfer using a learning rate of 0.00001.

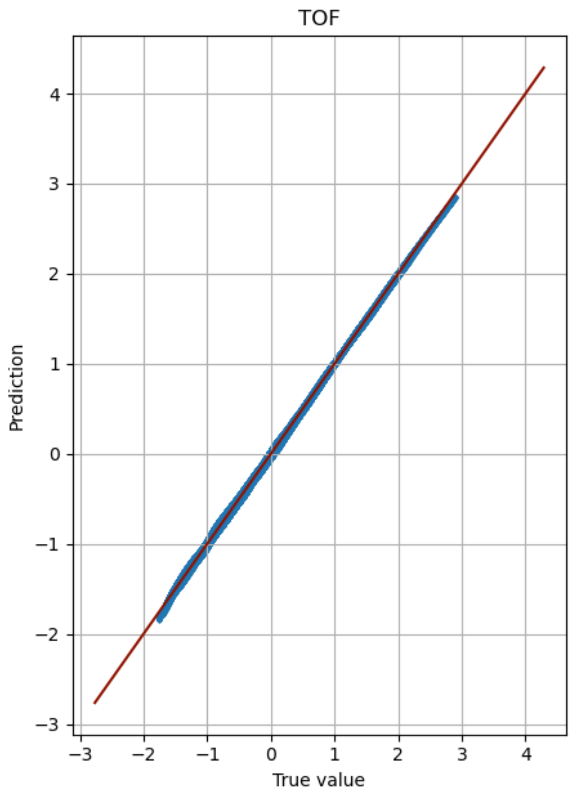


Figure C.4: Predicted vs true values of TOF of the standard Hohmann transfer using a learning rate of 0.00001.

These figures show, as also mentioned in the report, a worse performance compared to the nominal case, as expected.

D

Score results neural network

The score results for all the options, of which the figure is not shown in Chapter 5.

D.1. Case 1, phase 1

The results of different sample sizes are shown below:

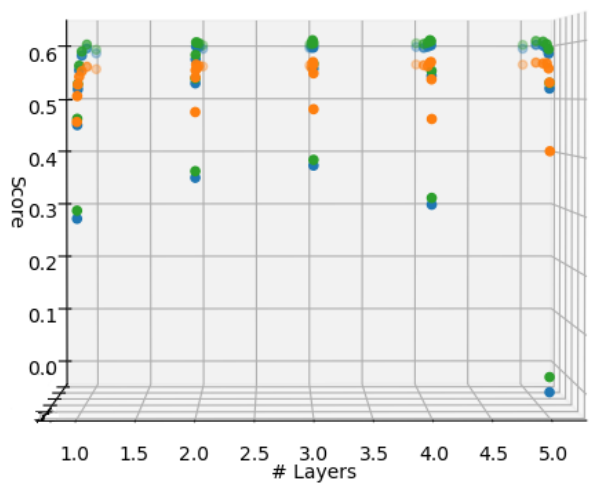


Figure D.1: Number of layers for a sample size of 25,000 - Case 1.

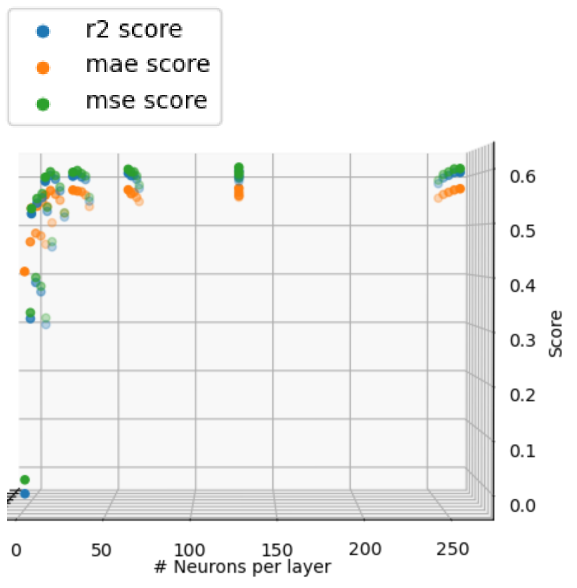


Figure D.2: Number of neurons per layer for a sample size of 25,000 - Case 1.

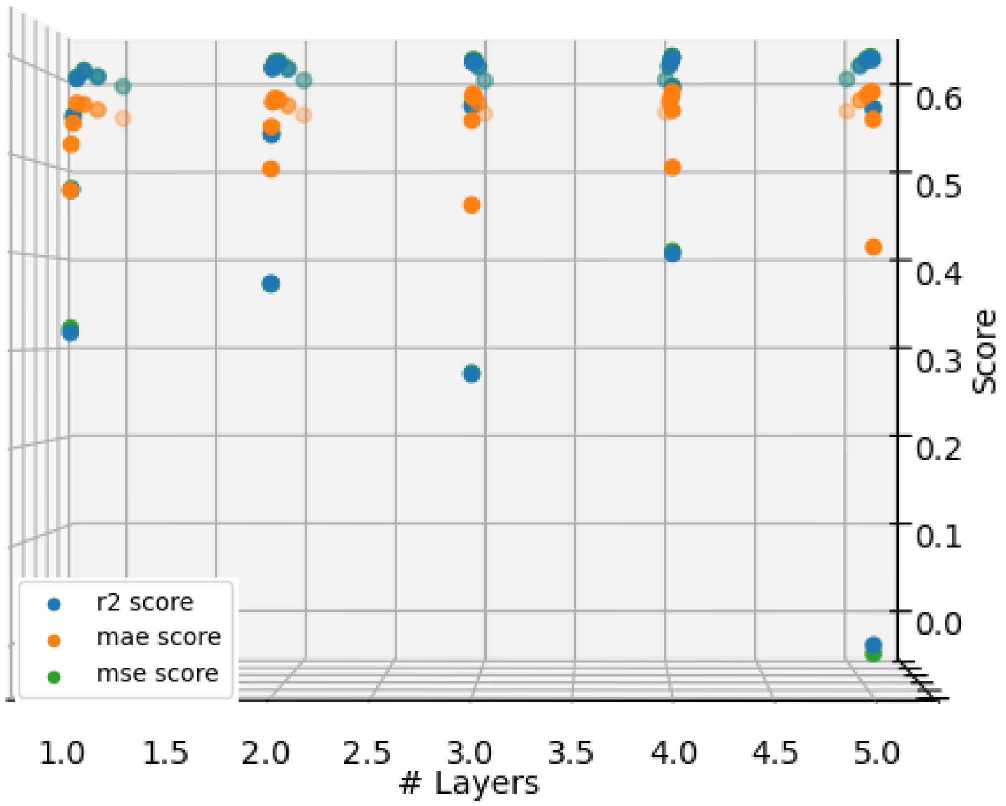


Figure D.3: Number of layers for a sample size of 45,000 - Case 1.

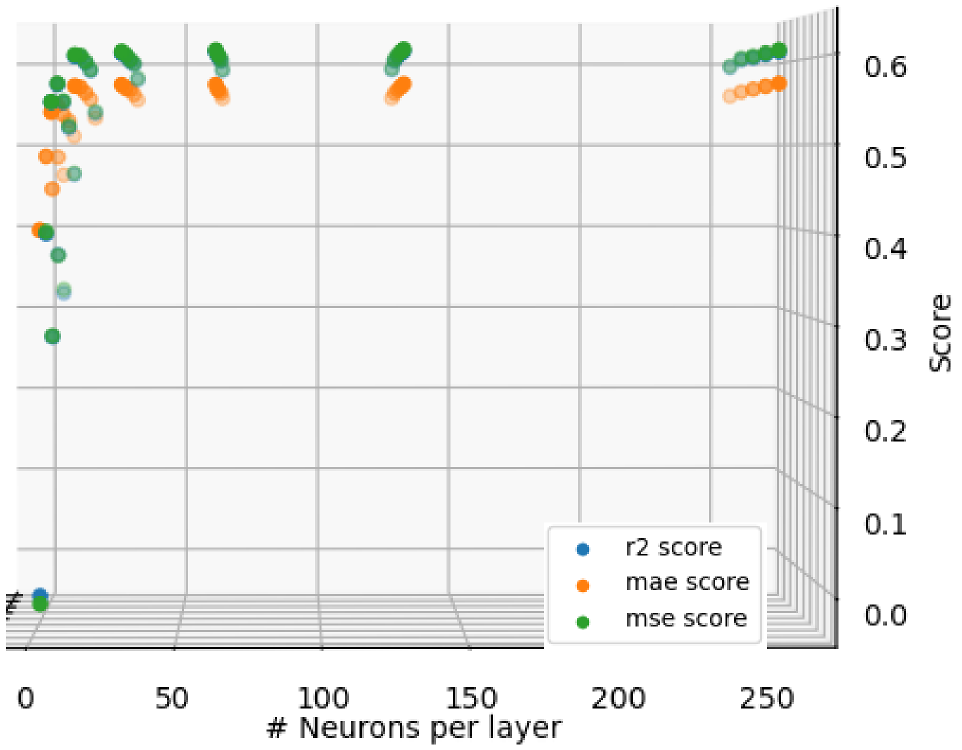


Figure D.4: Number of neurons per layer for a sample size of 45,000 - Case 1.

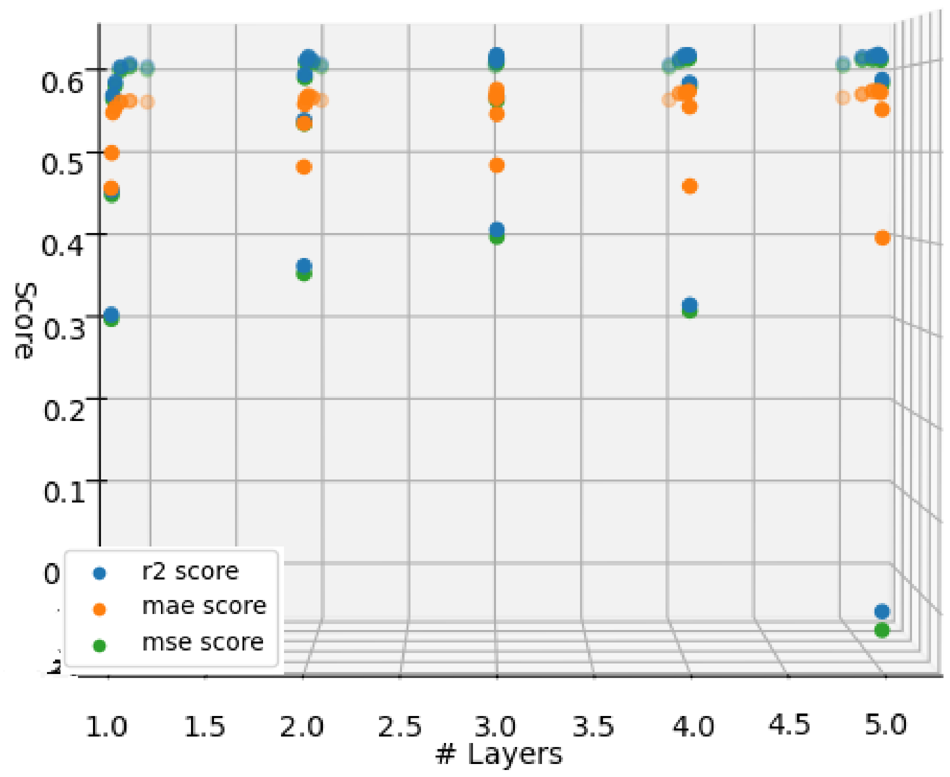


Figure D.5: Number of layers for a sample size of 65,000 - Case 1.

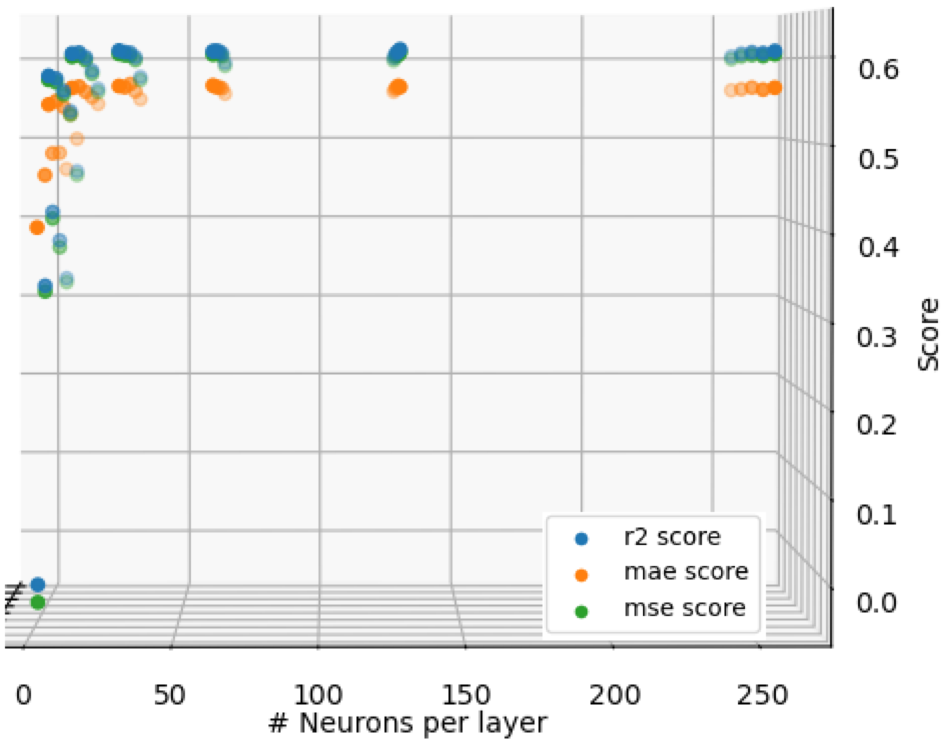


Figure D.6: Number of neurons per layer for a sample size of 65,000 - Case 1.

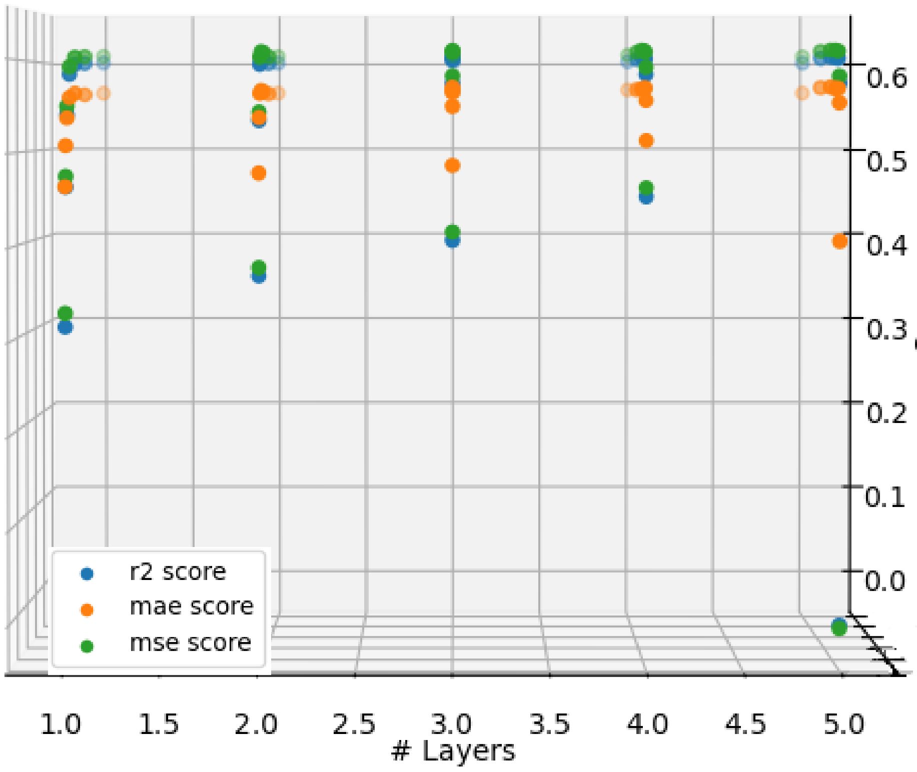


Figure D.7: Number of layers for a sample size of 90,000 - Case 1.

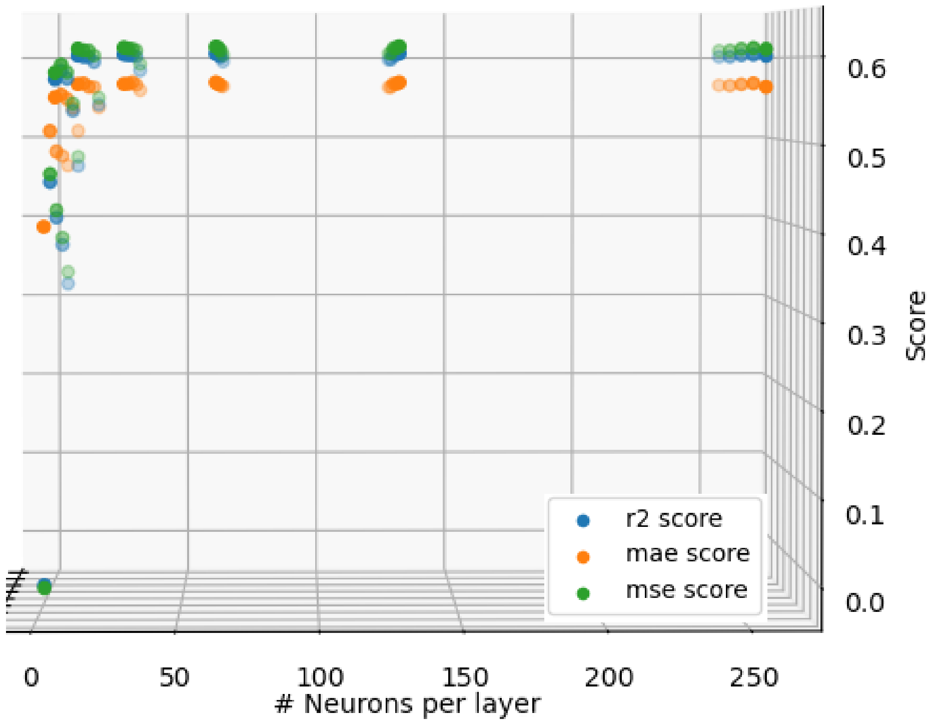


Figure D.8: Number of neurons per layer for a sample size of 90,000 - Case 1.

D.2. Case 1, phase 2

The results for other scalers are shown in the figures below:

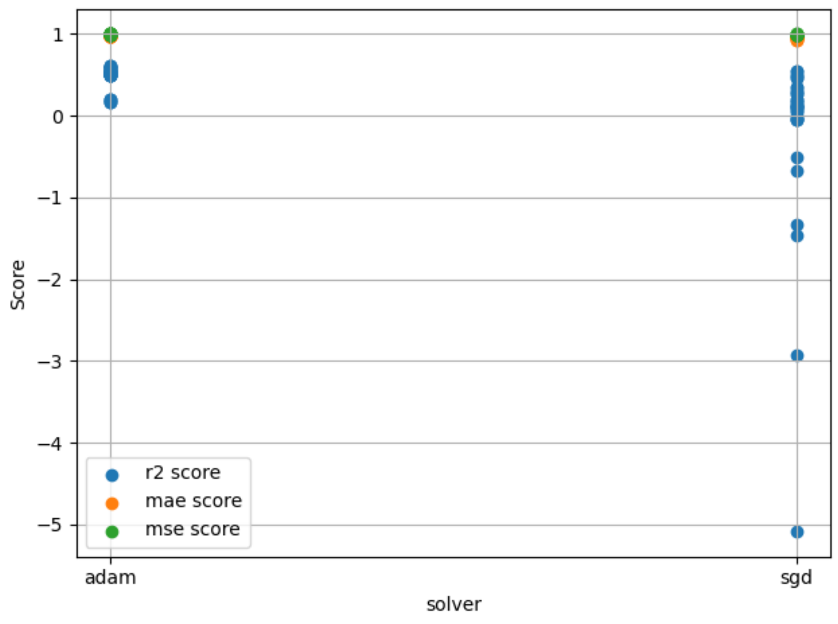


Figure D.9: Scores for different solvers using the MinMaxScaler for case 1.

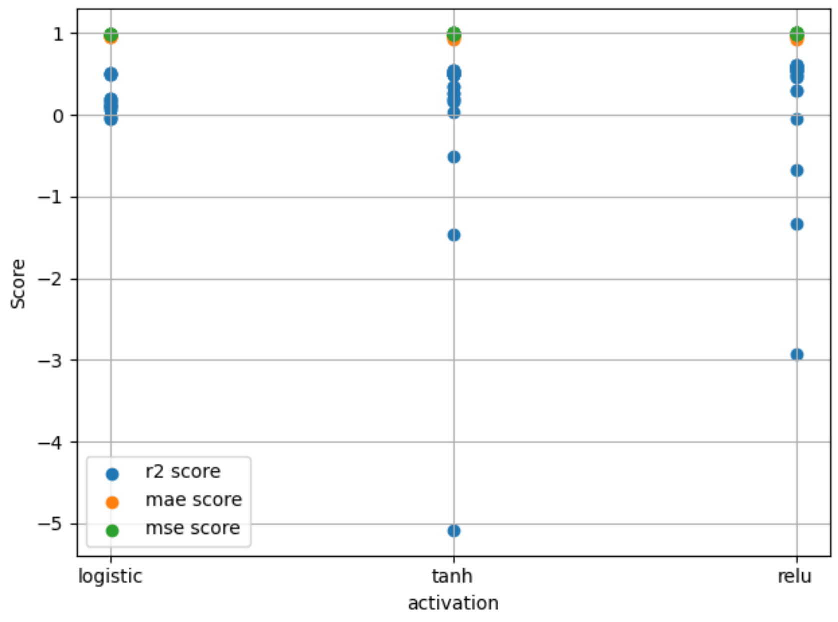


Figure D.10: Scores for different activation functions using the MinMaxScaler for case 1.

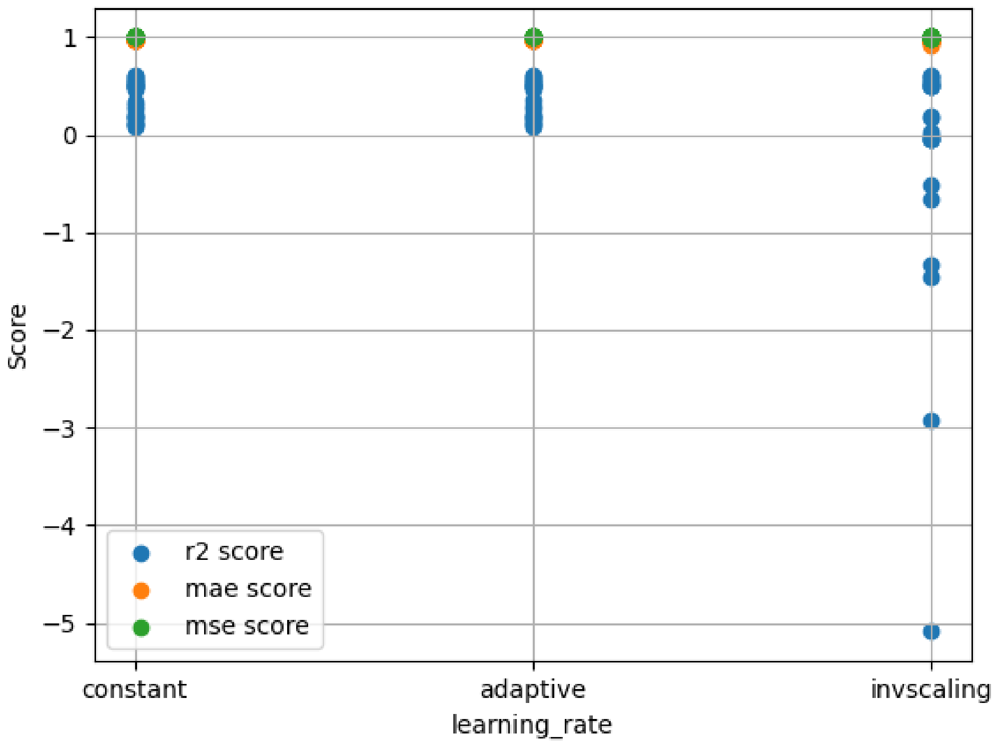


Figure D.11: Scores for different learning rate types using the MinMaxScaler for case 1.

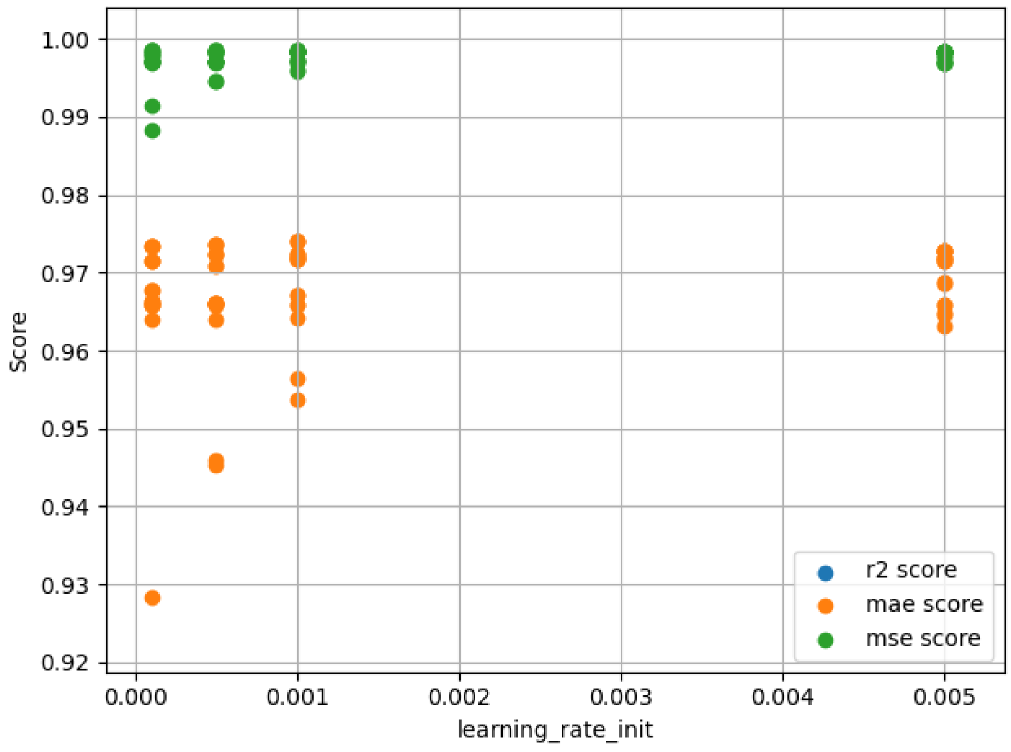


Figure D.12: Scores for different learning rates using the MinMaxScaler for case 1.

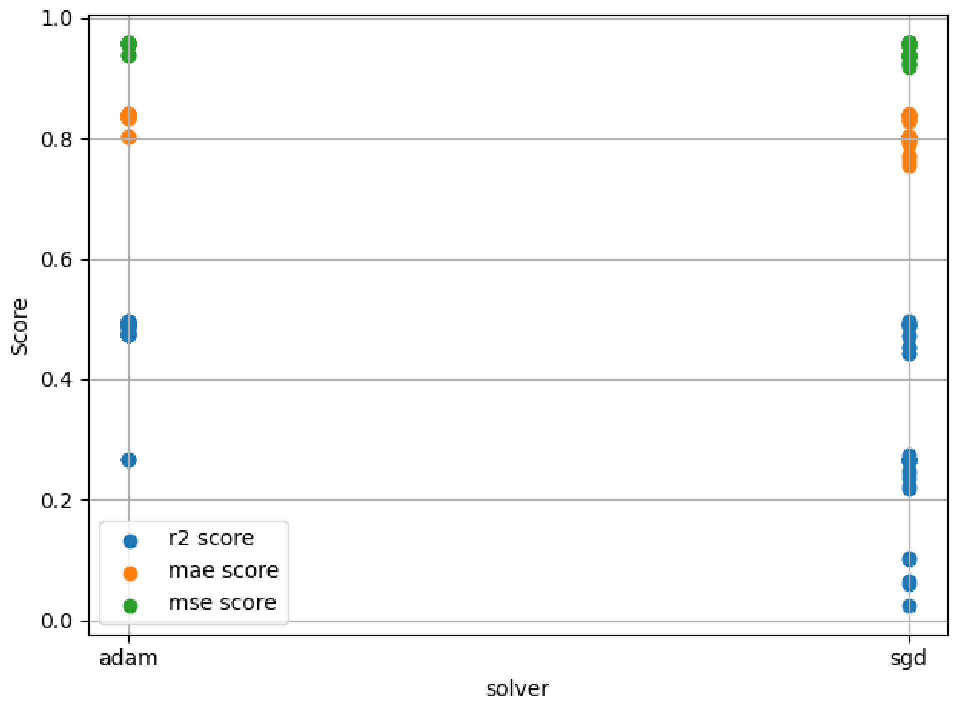


Figure D.13: Scores for different solvers using the QuantileTransformer for case 1.

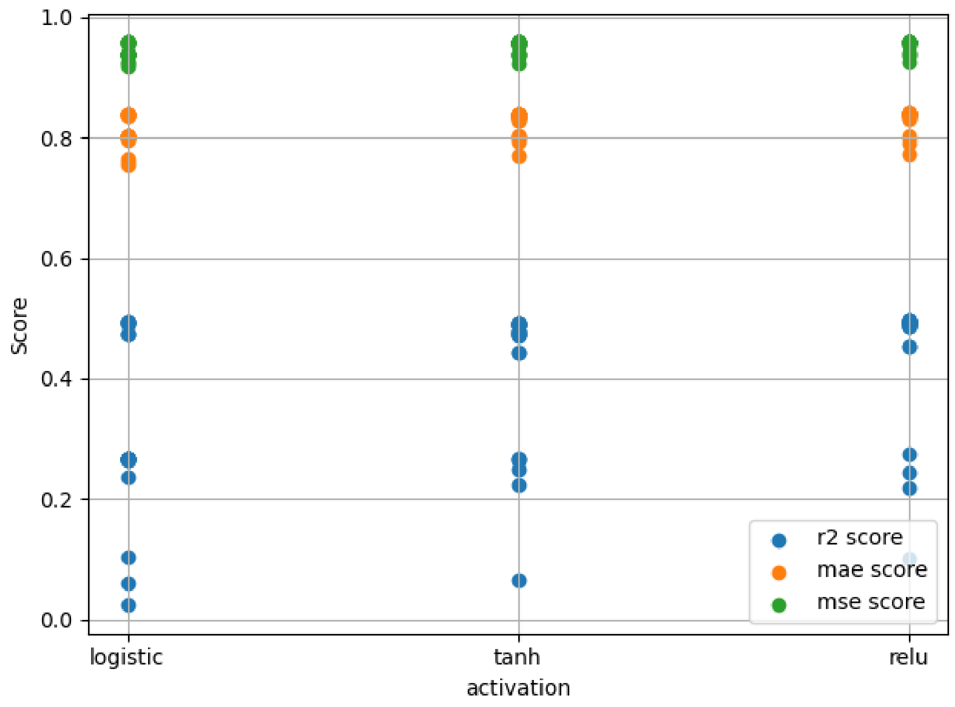


Figure D.14: Scores for different activation functions using the QuantileTransformer for case 1.

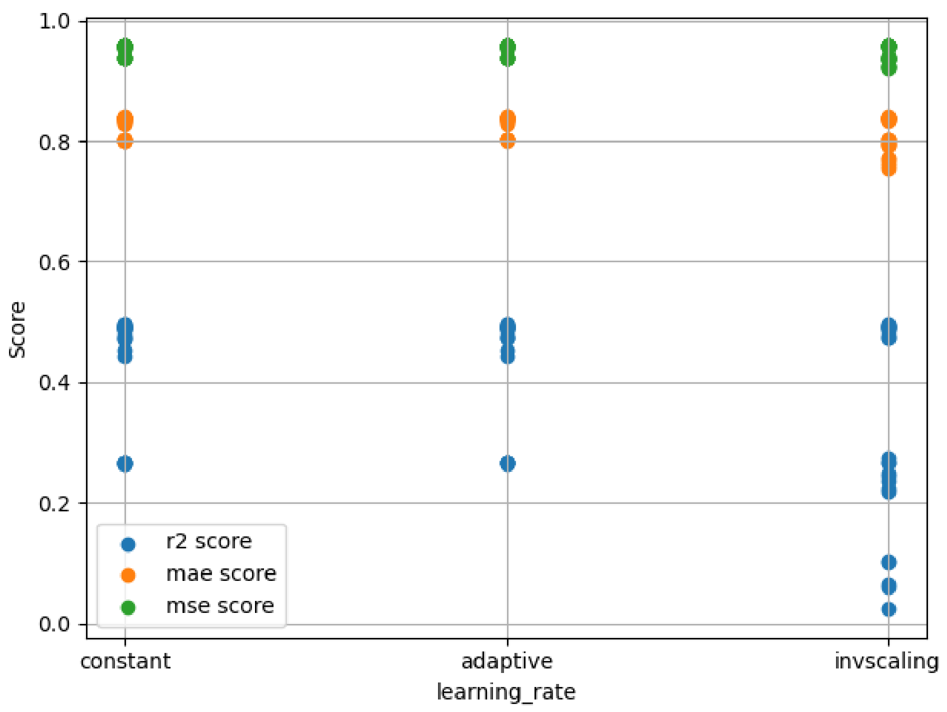


Figure D.15: Scores for different learning rate types using the QuantileTransformer for case 1.

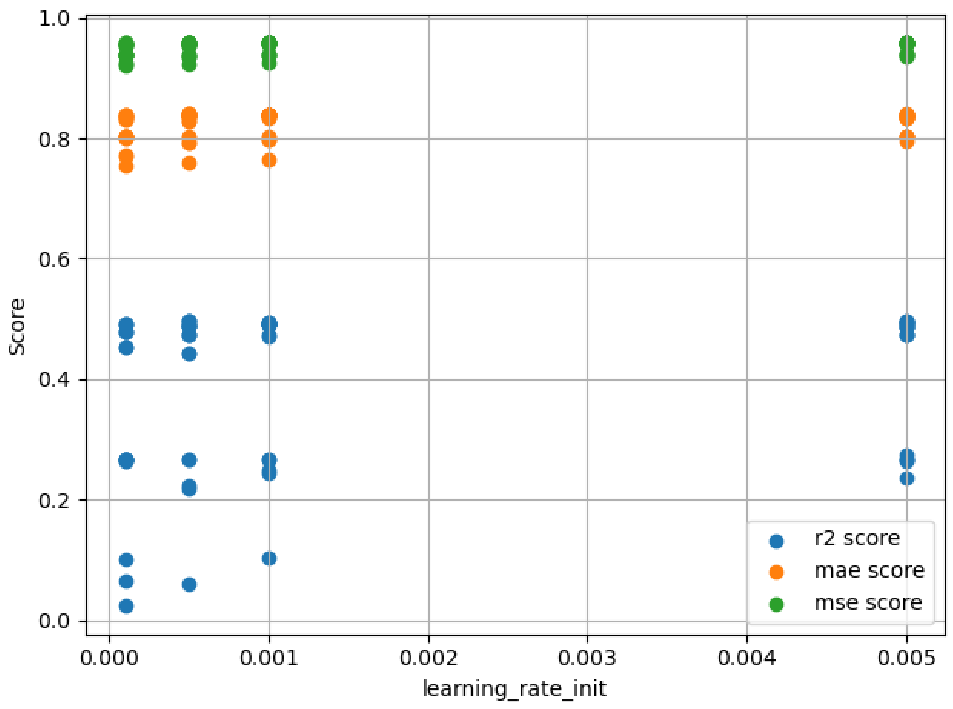


Figure D.16: Scores for different learning rates using the QuantileTransformer for case 1.

D.3. Case 2, phase 1

The results of case 2, phase 1 are shown below:

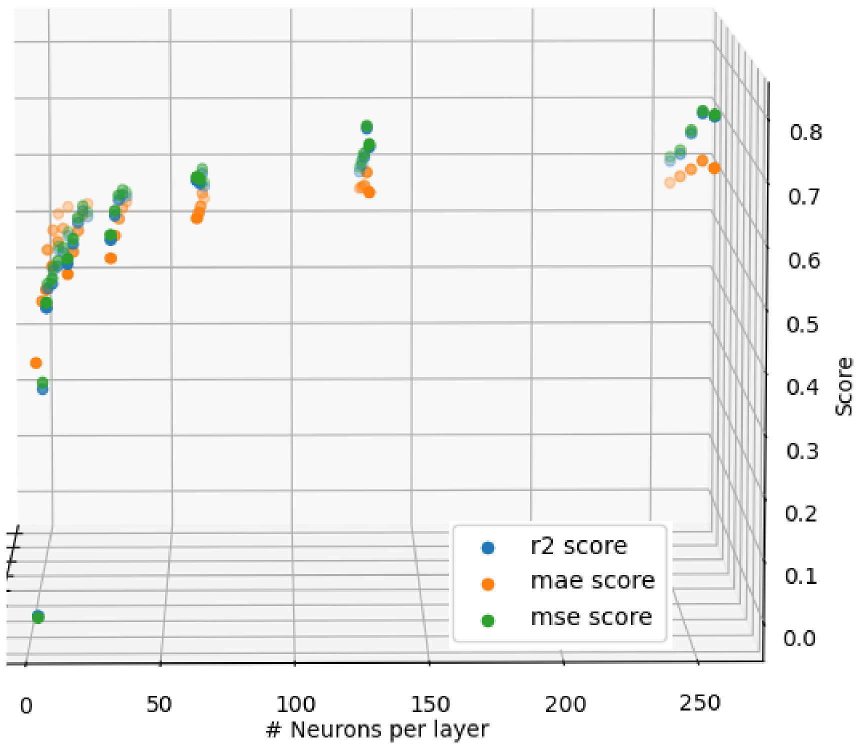


Figure D.17: Neurons per layer for a sample size of 90,000 for case 2.

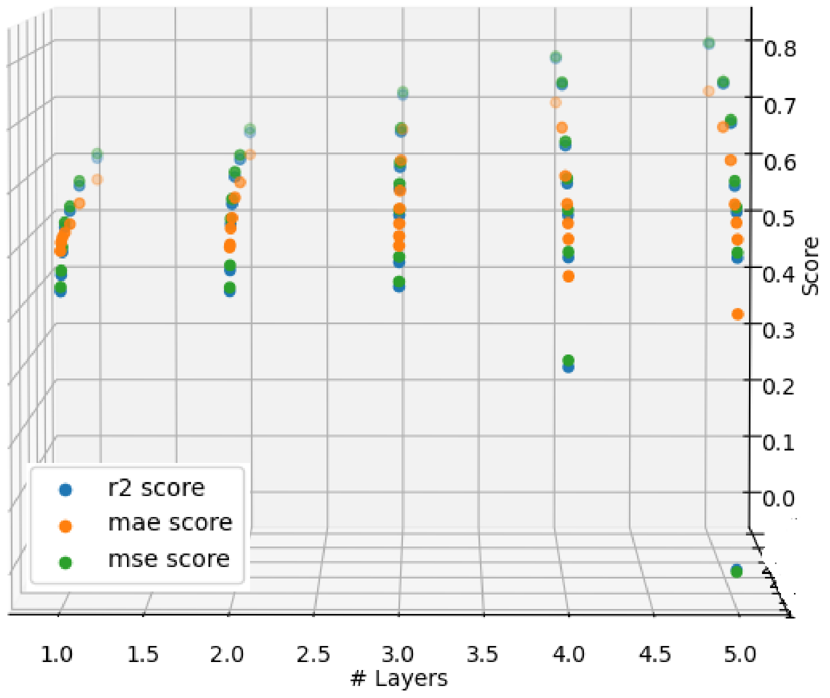


Figure D.18: Layers for a sample size of 90,000 for case 2.

D.4. Case 2, phase 2

The results of SS is shown below:

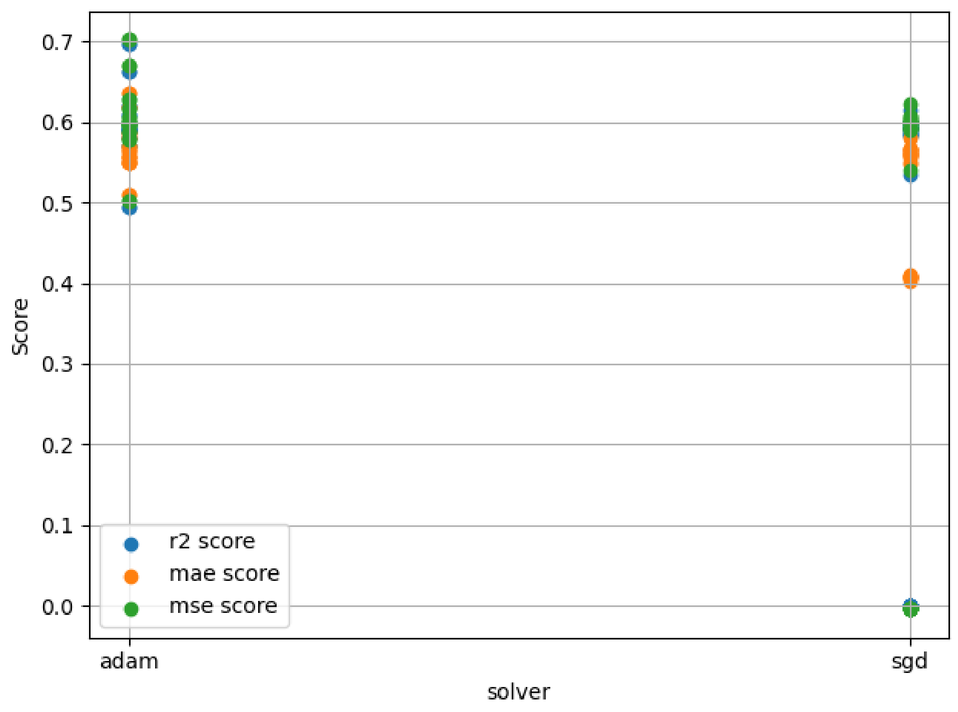


Figure D.19: Scores for different solvers using the StandardScaler for case 2.

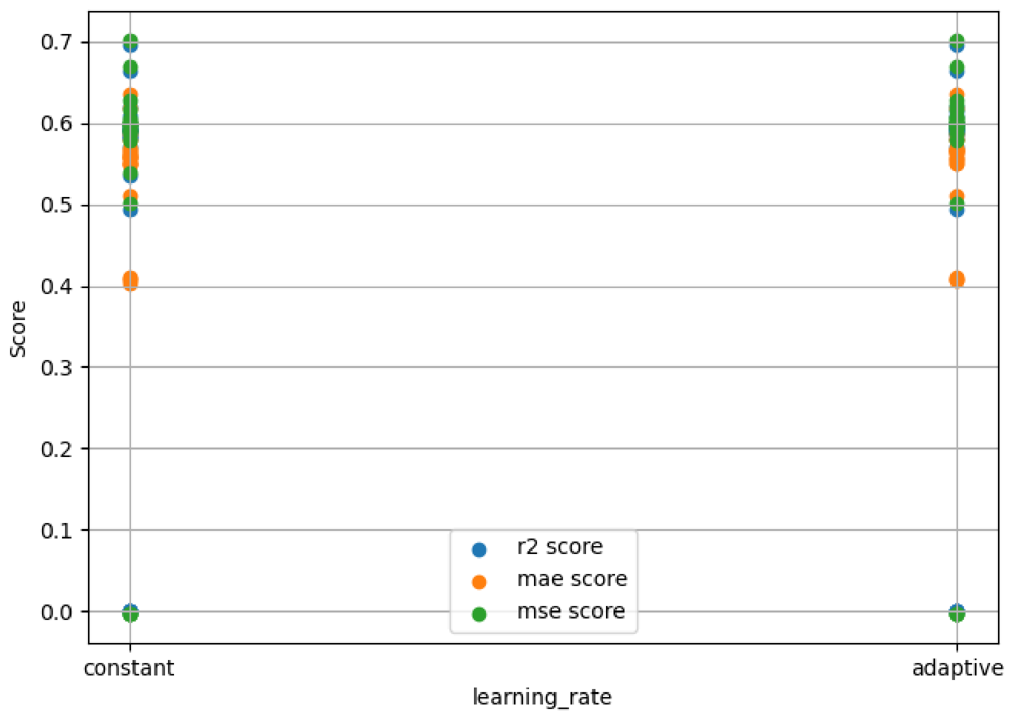


Figure D.20: Scores for different learning rate types using the StandardScaler for case 2.

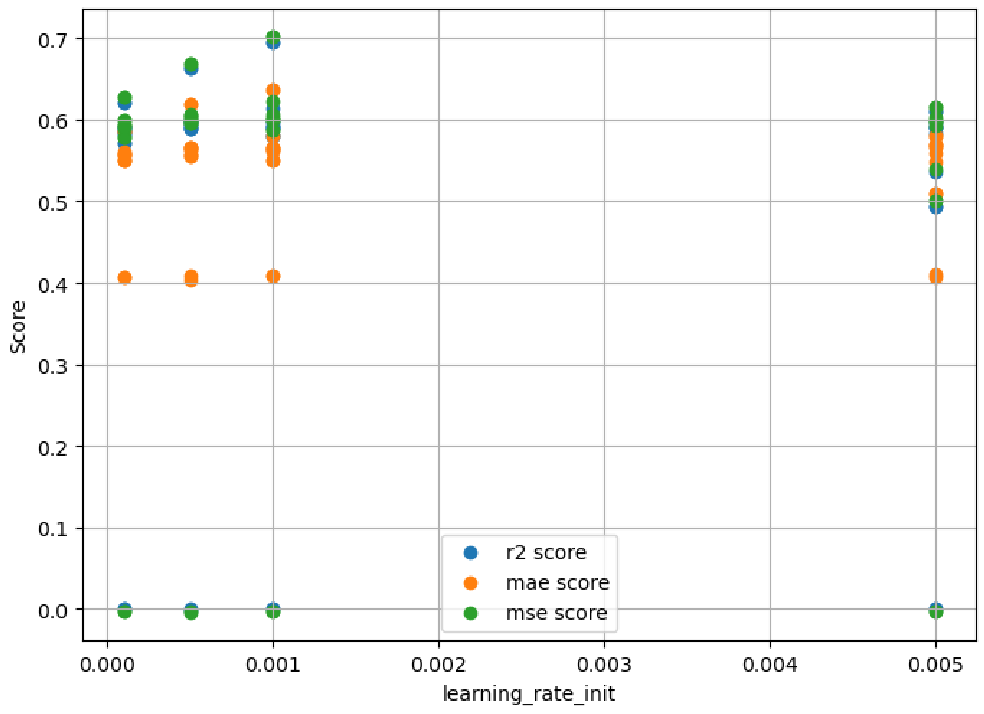


Figure D.21: Scores for different learning rates using the StandardScaler for case 2.

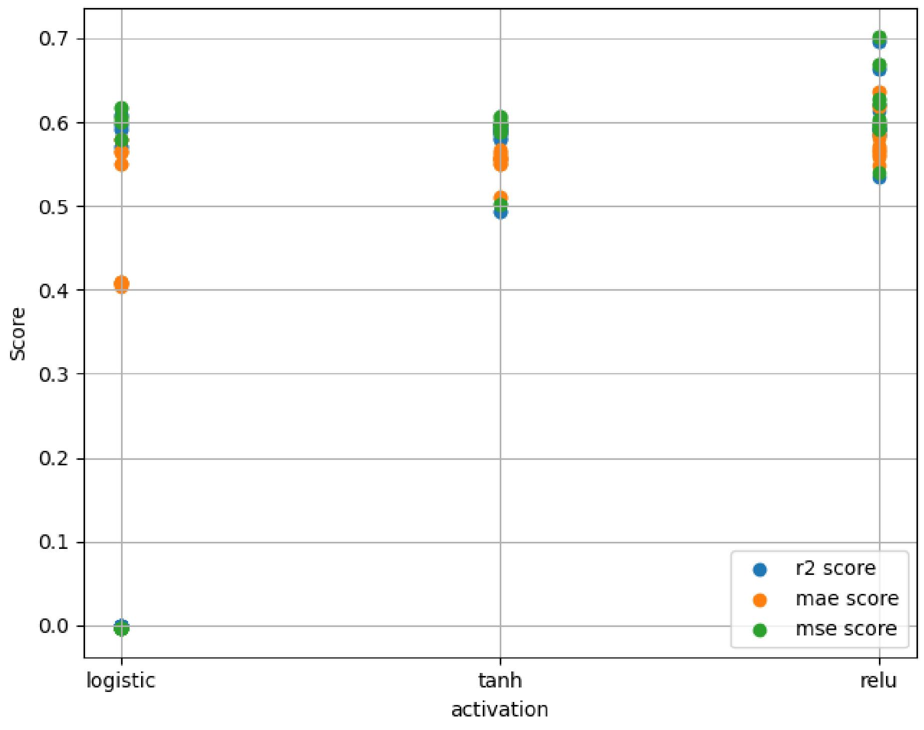


Figure D.22: Scores for different activation functions using the StandardScaler for case 2.

The MMS is only varied for solvers and activation functions, as the learning rate did not show significant differences for SS.

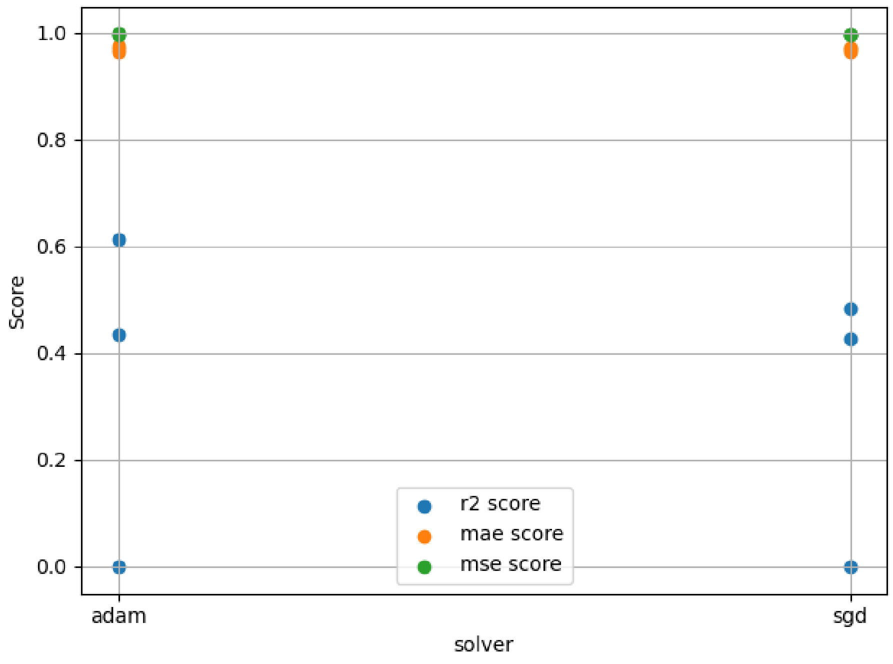


Figure D.23: Scores for different solvers using the MinMaxScaler for case 2.

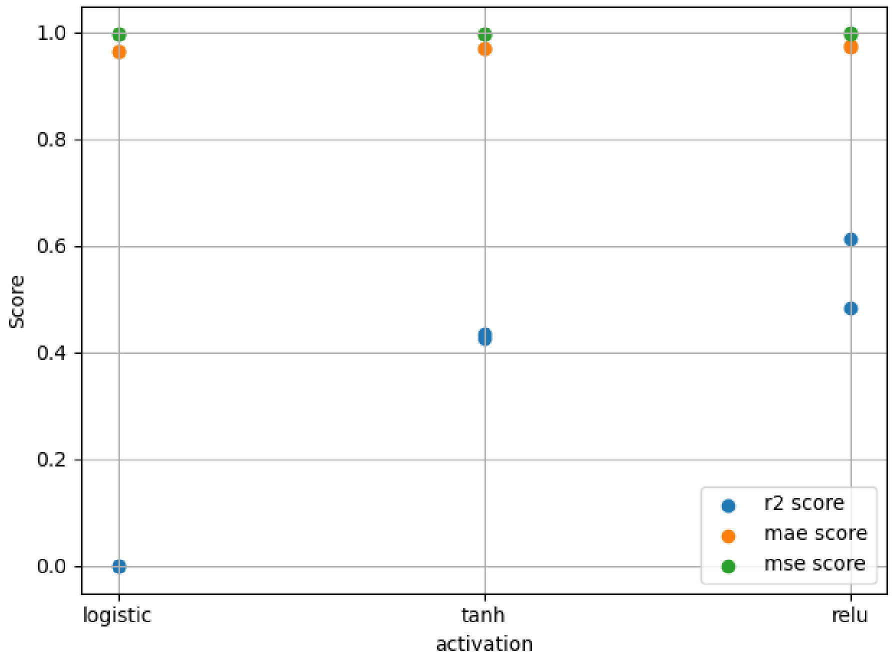


Figure D.24: Scores for different activation functions using the MinMaxScaler for case 2.

D.5. Case 3, phase 1

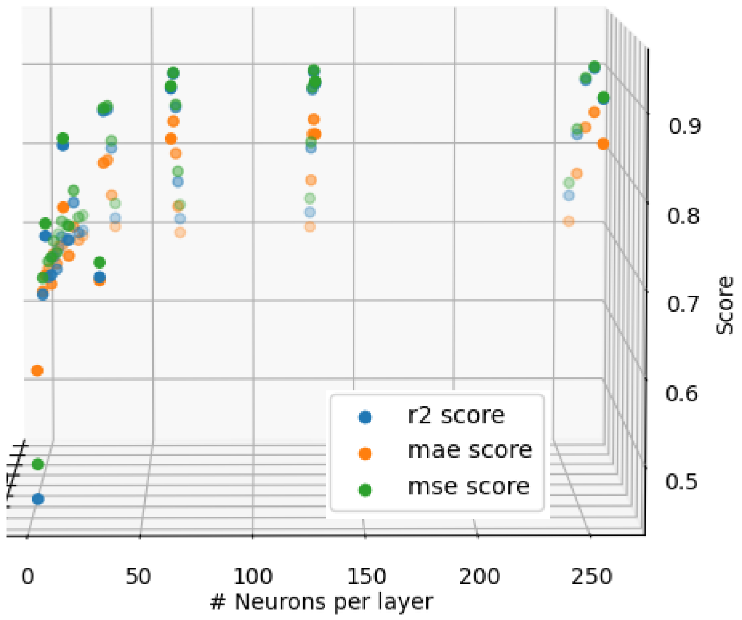


Figure D.25: Neurons per layer for a sample size of 90,000 for case 3.

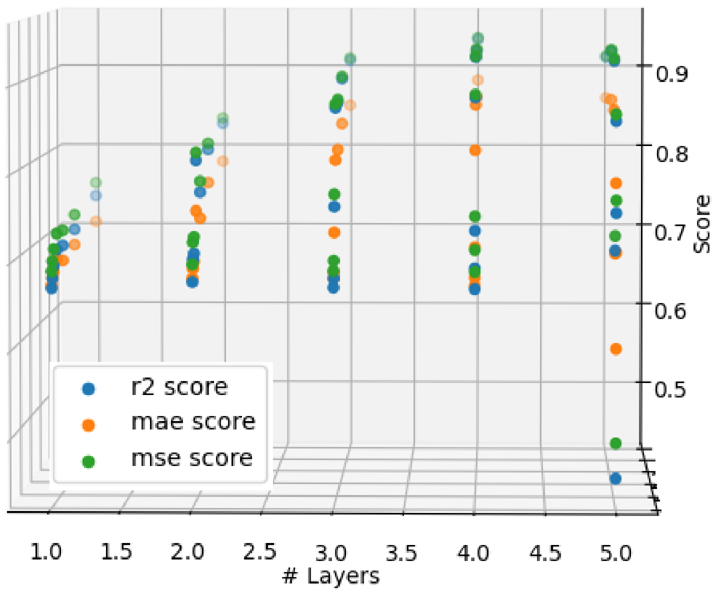


Figure D.26: Layers for a sample size of 90,000 for case 3.

D.6. Case 3, phase 2

The results of using SS are shown below:

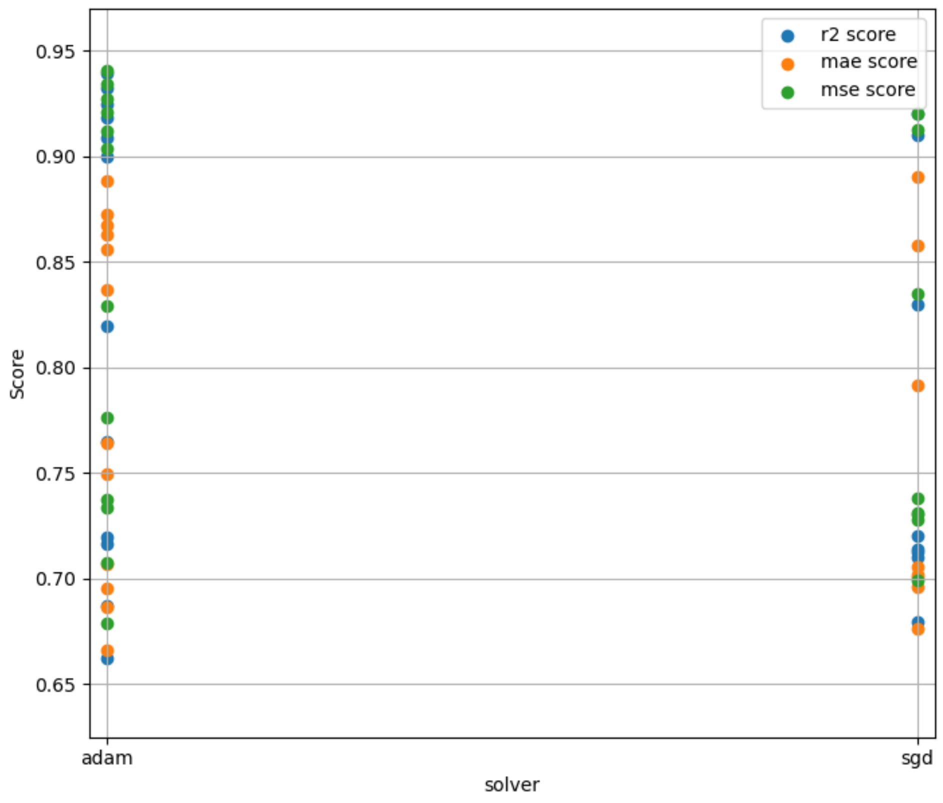


Figure D.27: Scores for different solvers using the StandardScaler for case 3.

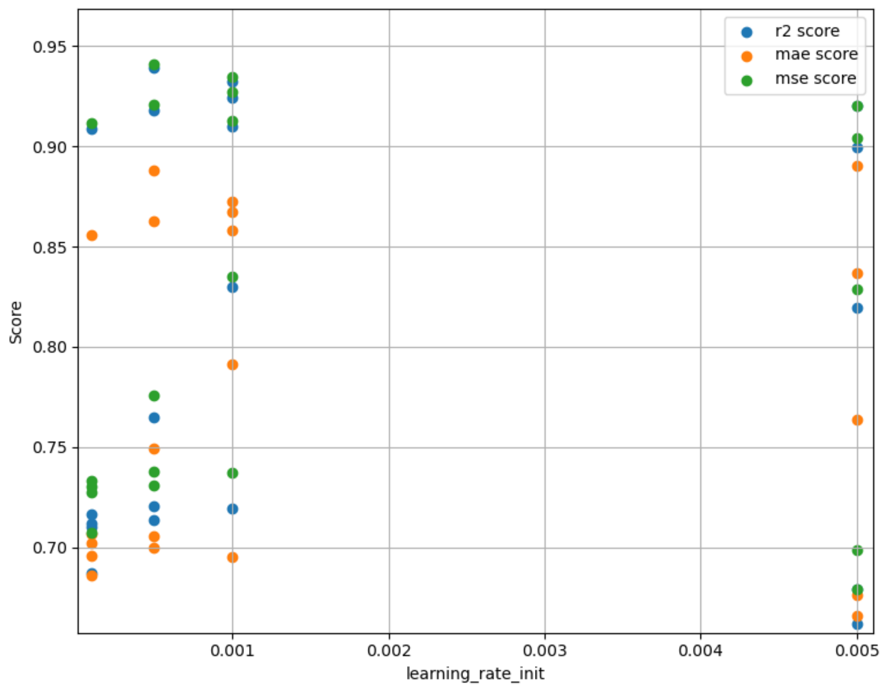


Figure D.28: Scores for different learning rates using the StandardScaler for case 3.

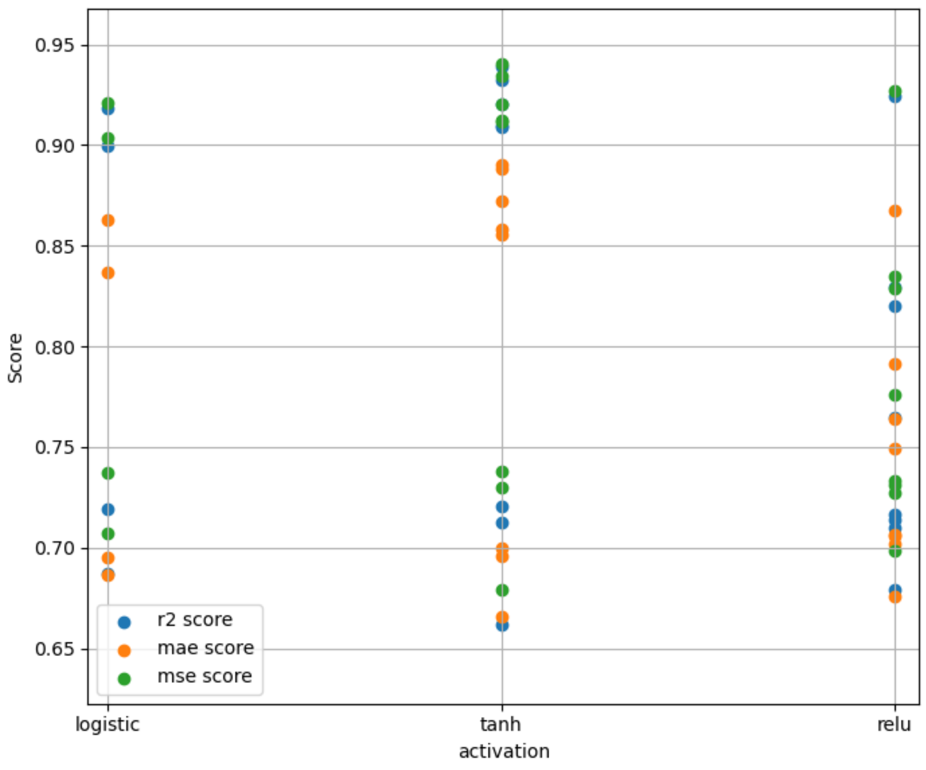


Figure D.29: Scores for different activation functions using the StandardScaler for case 3.

The results of using MMS are shown below:

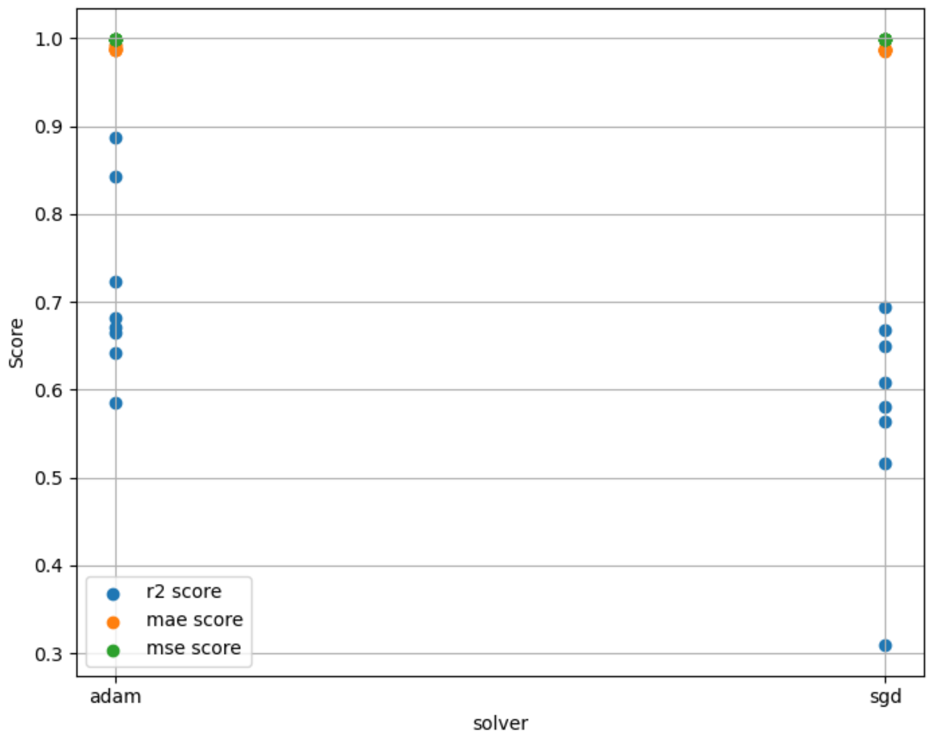


Figure D.30: Scores for different solvers using the MinMaxScaler for case 3.

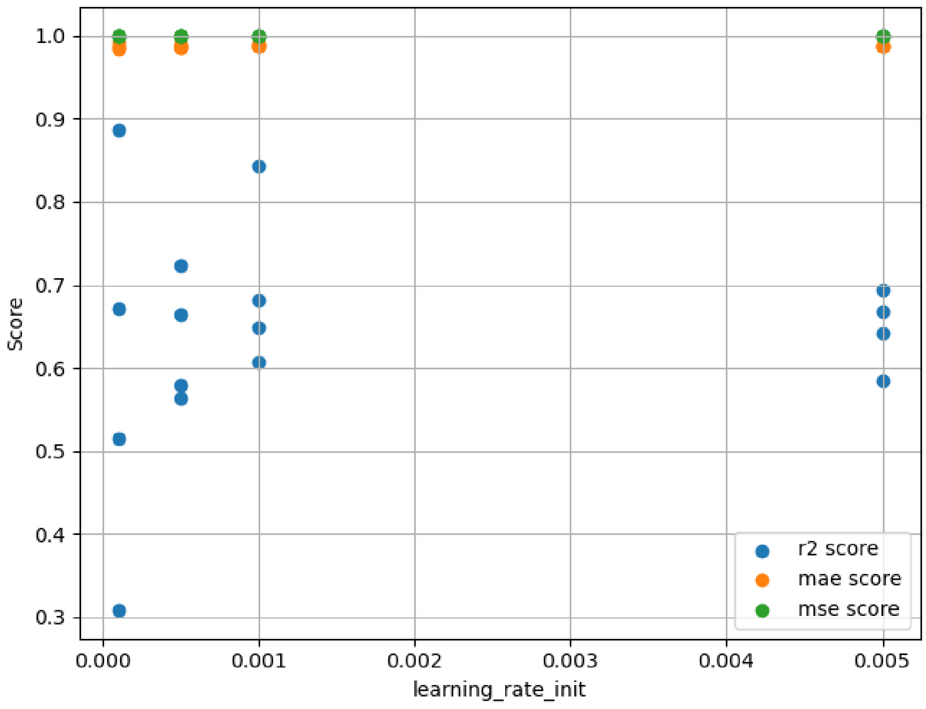


Figure D.31: Scores for different learning rates using the MinMaxScaler for case 3.

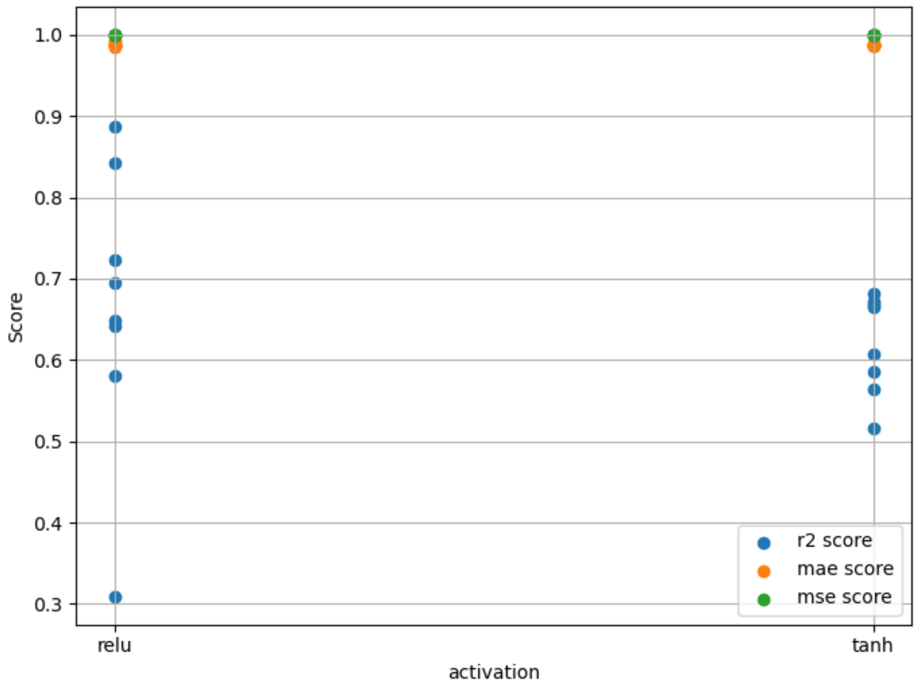


Figure D.32: Scores for different activation functions using the MinMaxScaler for case 3.

D.7. Best result tuning

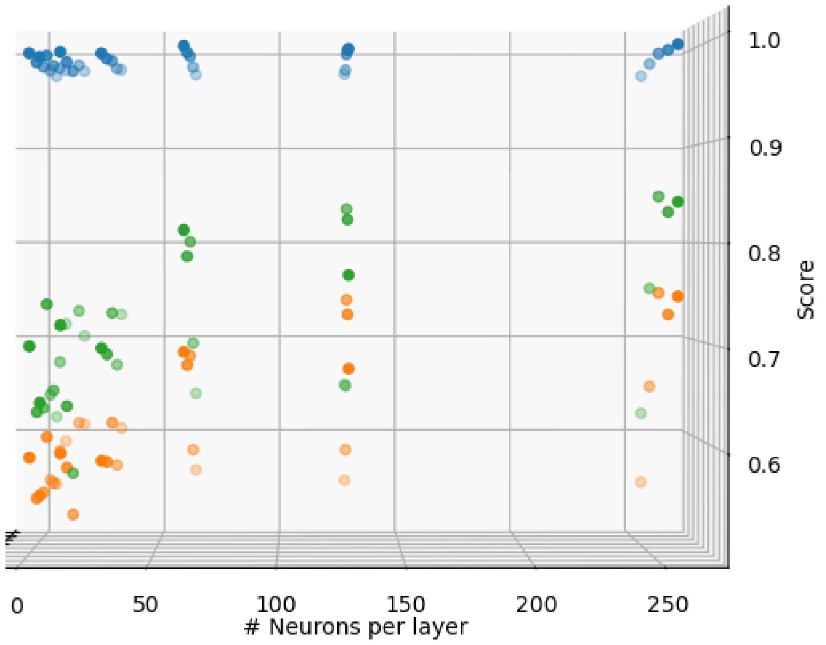


Figure D.33: Neurons per layer for a sample size of 90,000 for the current best case.

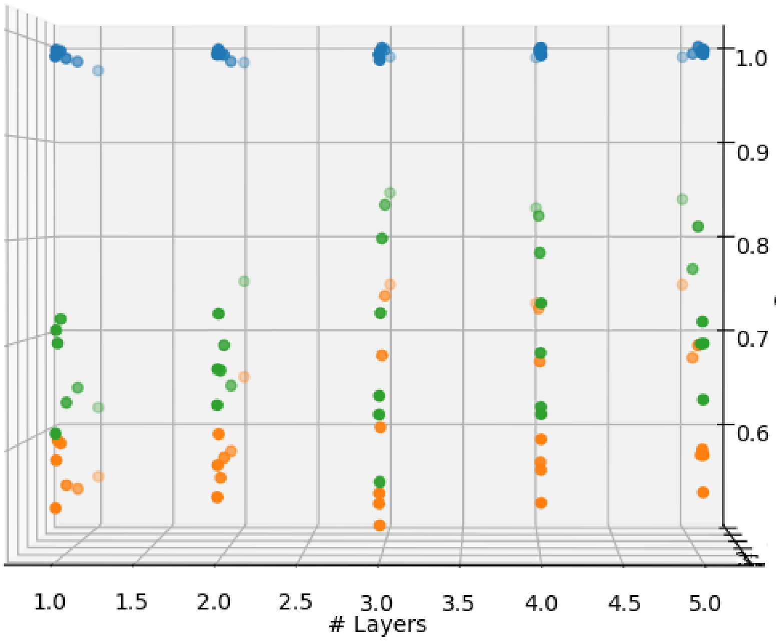


Figure D.34: Layers for a sample size of 90,000 for the current best case.

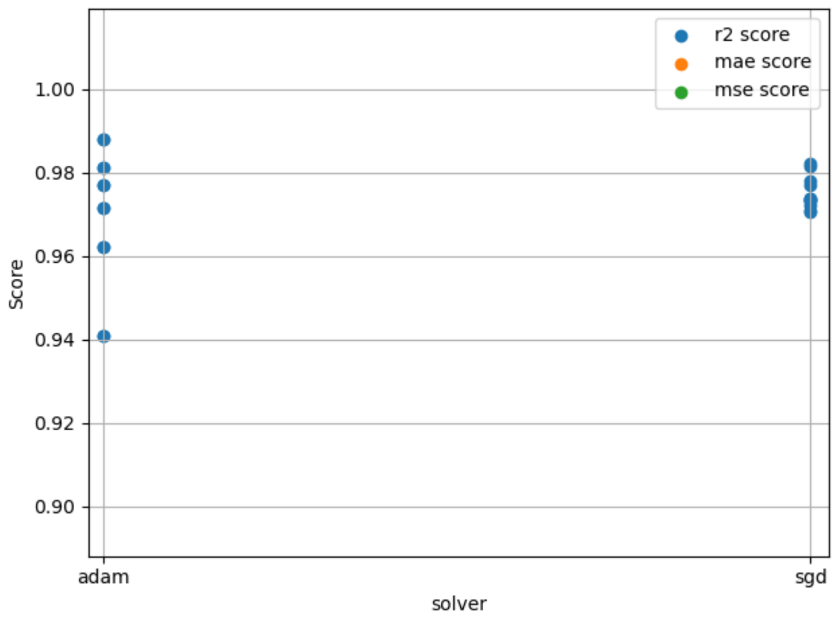


Figure D.35: Scores for different solvers using the StandardScaler for the best case. Only R^2 scores are shown for better visualization and comparison.

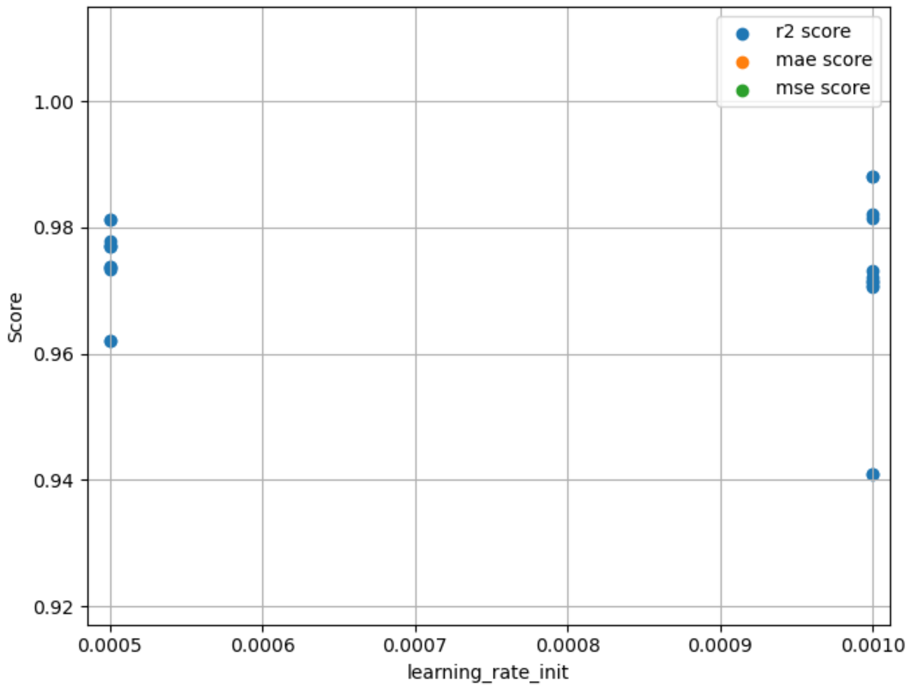


Figure D.36: Scores for different learning rates using the StandardScaler for the best case. Only R^2 scores are shown for better visualization and comparison.

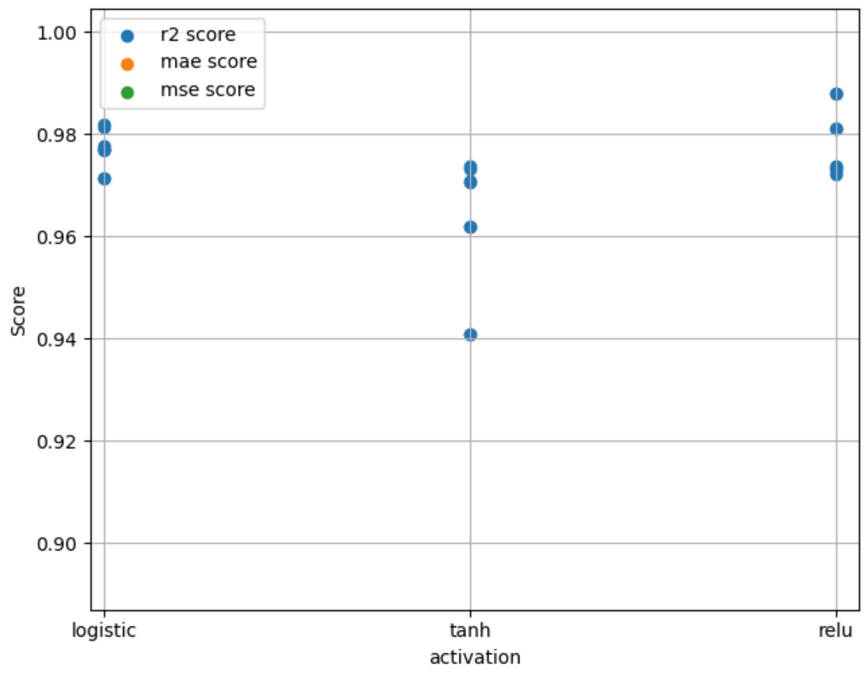


Figure D.37: Scores for different activation functions using the StandardScaler for the best case. Only R^2 scores are shown for better visualization and comparison.

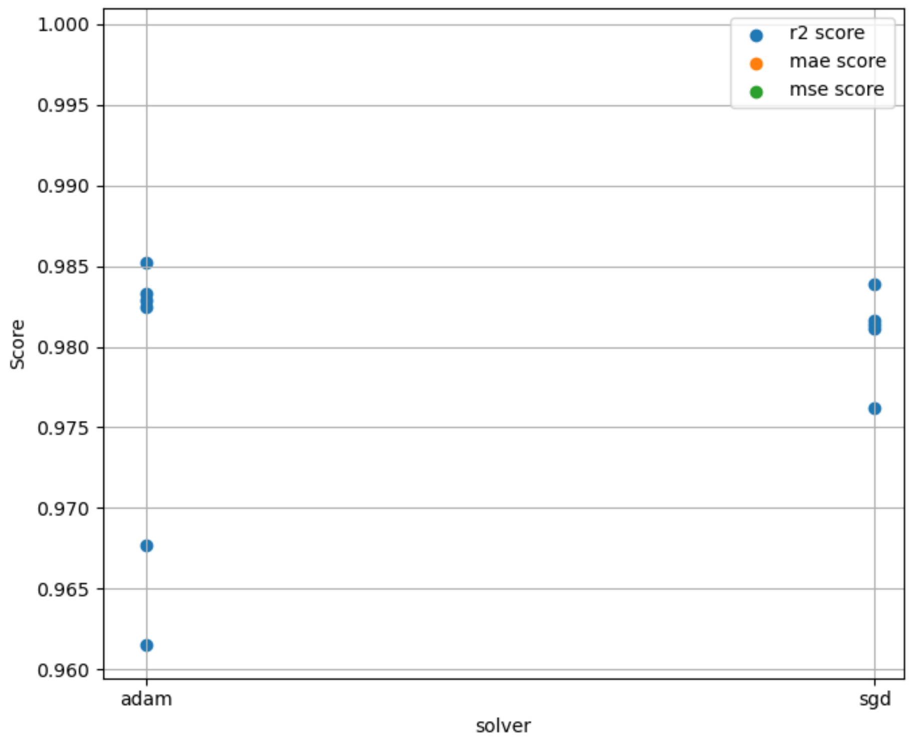


Figure D.38: Scores for different solvers using the MinMaxScaler for the best case. Only R^2 scores are shown for better visualization and comparison.

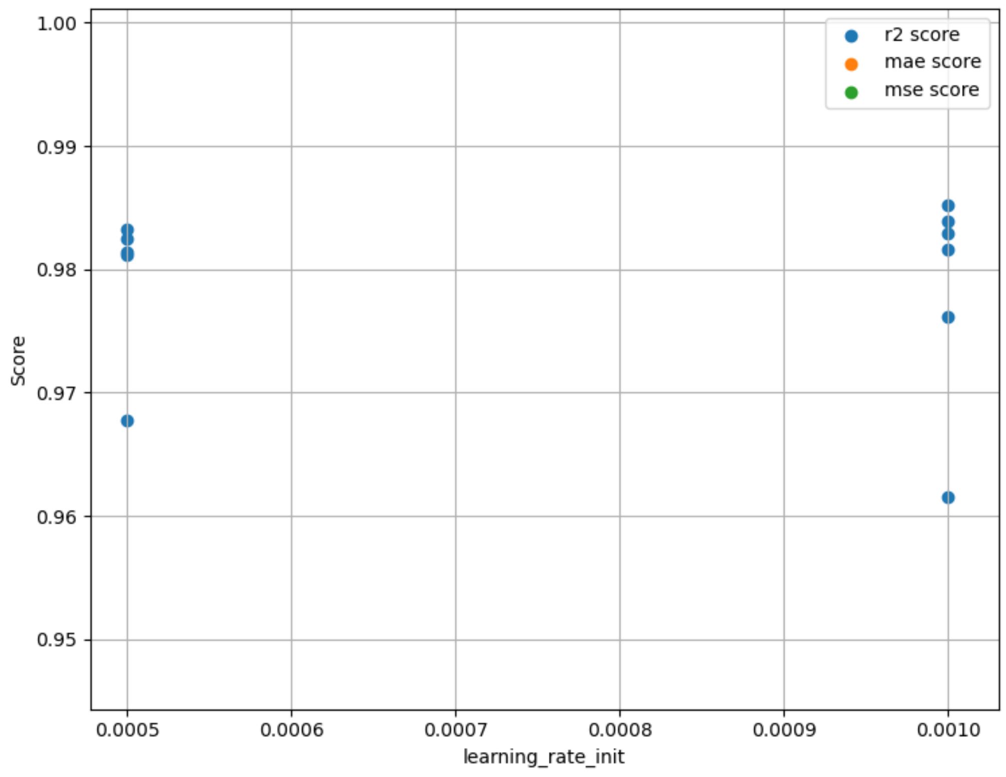


Figure D.39: Scores for different learning rates using the MinMaxScaler for the best case. Only R^2 scores are shown for better visualization and comparison.

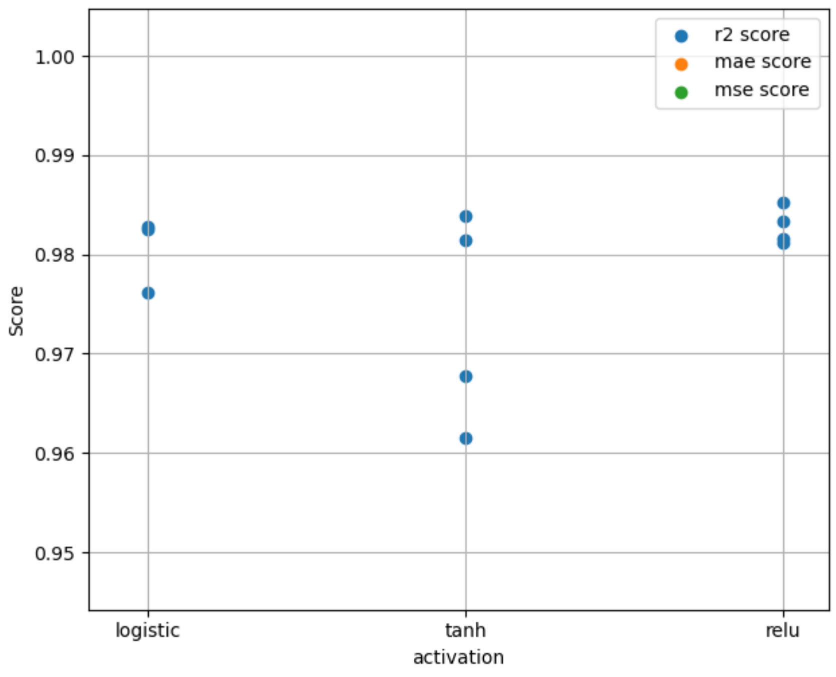


Figure D.40: Scores for different activation functions using the MinMaxScaler for the best case. Only R^2 scores are shown for better visualization and comparison.

E
Close approach data fraction results

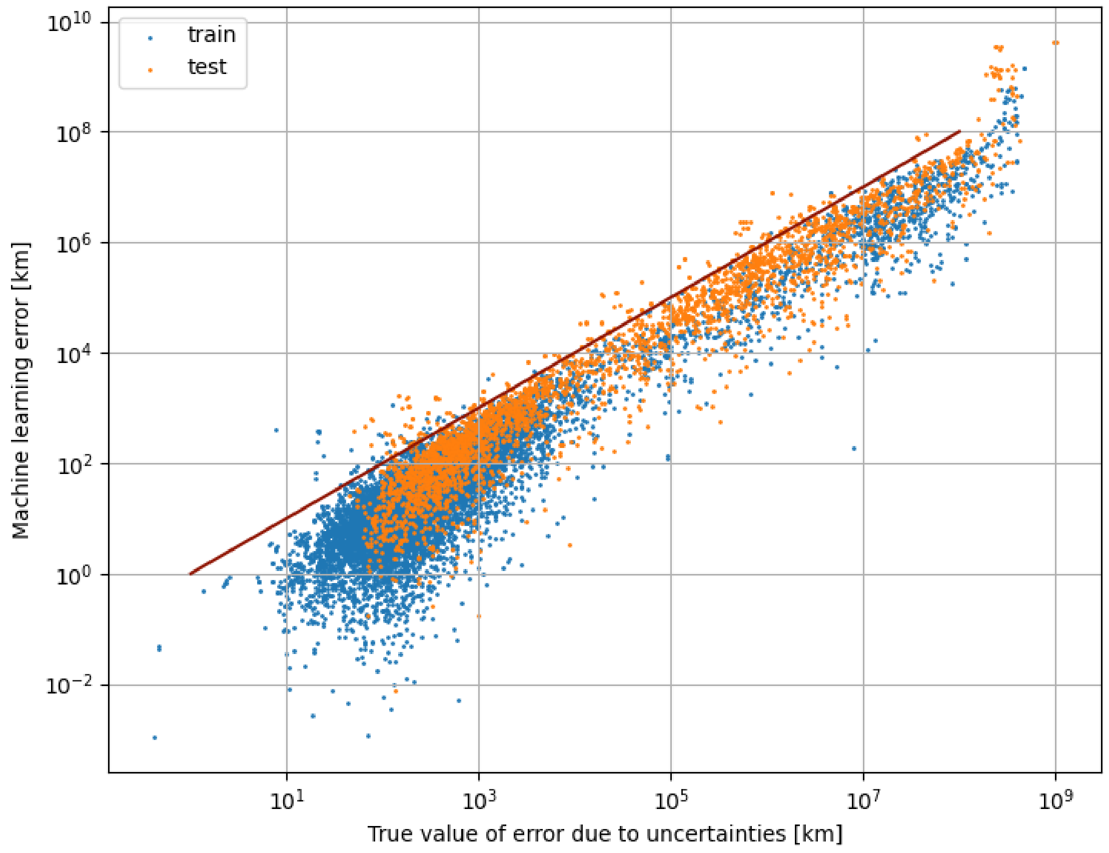


Figure E.1: Data fraction - uncertainty - 15,000 data points.

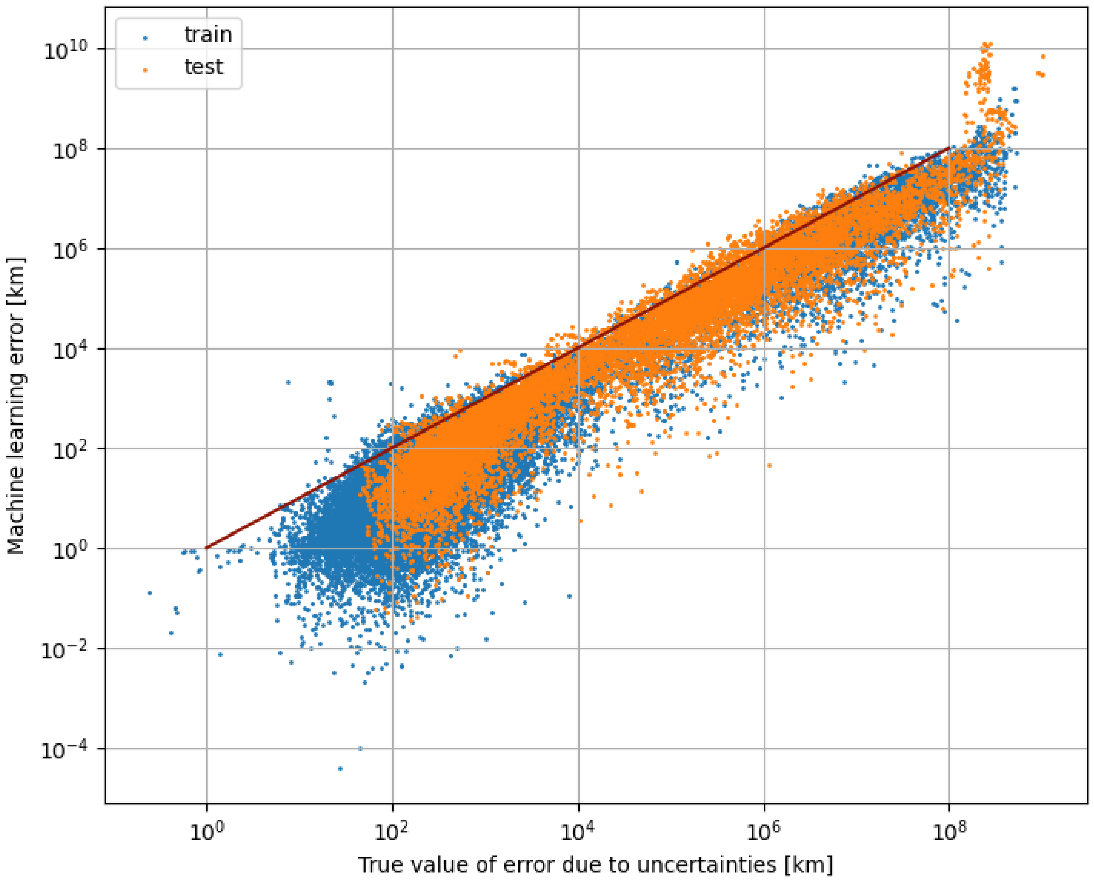


Figure E.2: Data fraction - uncertainty - 65,000 data points.

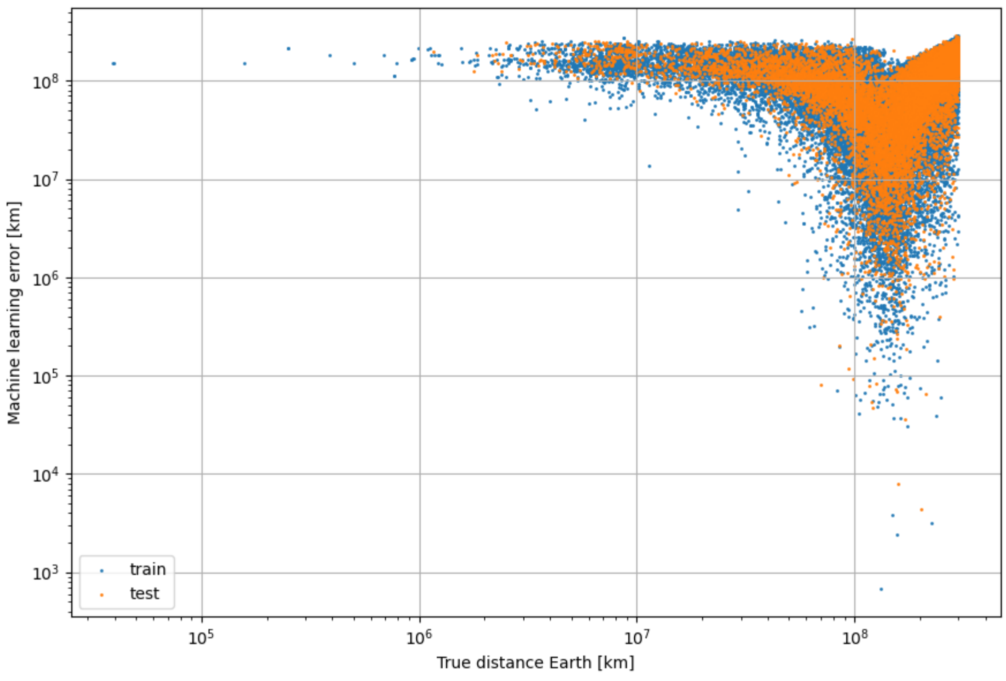


Figure E.3: Data fraction - Earth distance - 65,000 data points.