Physics Constrained Neural Networks for Designing Gravity Assist Trajectories

Master Thesis Midterm Review

Mitchell van Doorn

4-11-2024

Contents

1

Recap Research Proposal

Research questions

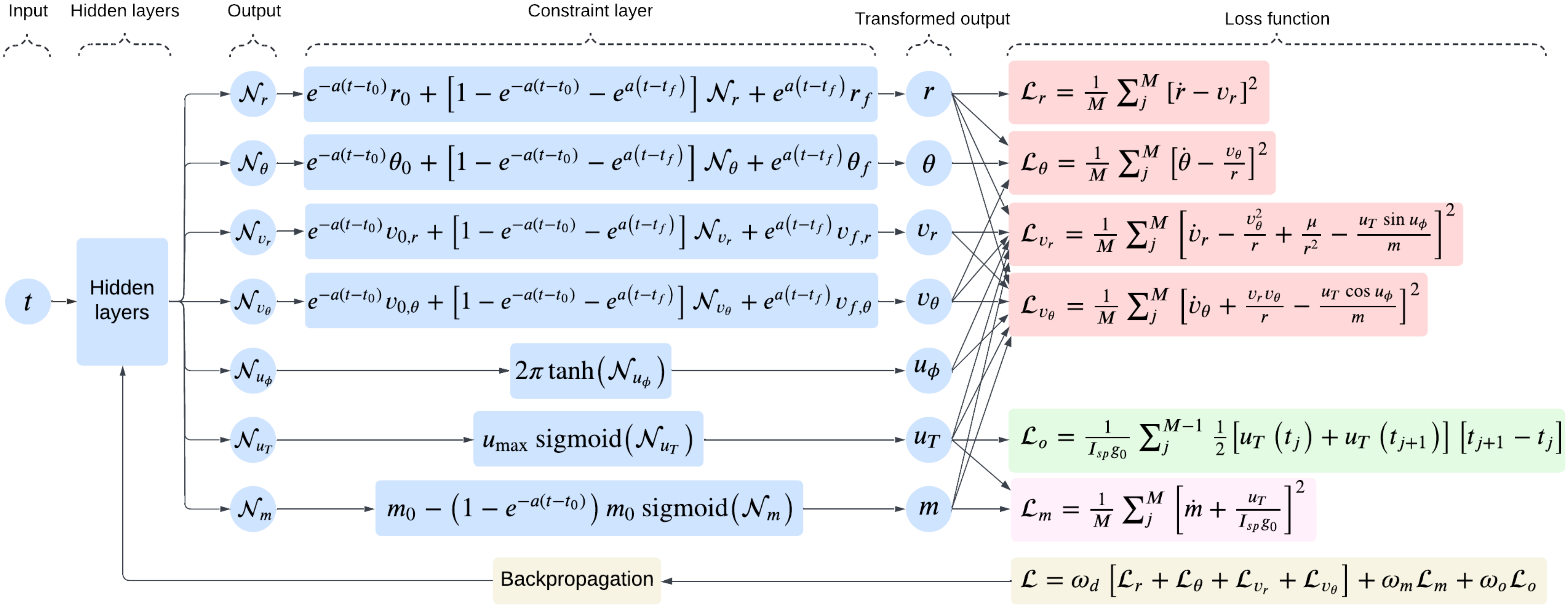
Main Research Question:

“How can Physics-Constrained Neural Networks (PCNNs) be effectively utilized to design optimal trajectories involving gravity assist for spacecraft with low-thrust propulsion systems?”

Sub questions:

- How can the dynamics of gravity assist trajectories be integrated in PCNNs?
- How can PCNNs learn to perform gravity assist maneuvers?
- How can suitable neural network architectures be established?
- How can the PCNN model be trained effectively?
- What are the benefits and limitations of using PCNNs for optimizing low-thrust gravity assists trajectories?

Basic PCNN design



Subproblems to be solved

Subproblem 1: Earth → Jupiter rendezvous

- Fixed x_0, x_f, ToF

Subproblem 2: Earth → Fixed Time Jupiter GA → Pluto

- Fixed x_0, x_f, ToF, t_{GA}
- Variable x_{GA}

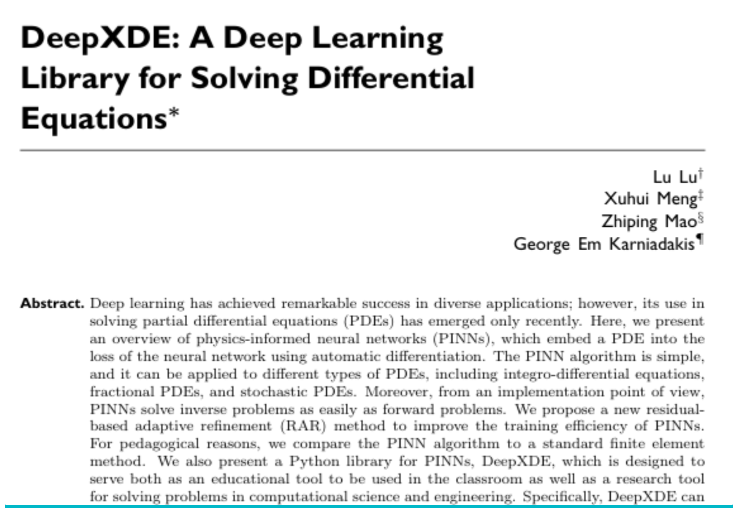
Subproblem 3: Earth → Jupiter GA → Pluto

- Fixed x_0, x_f, ToF
- Variable x_{GA}, t_{GA}

2

Results so far

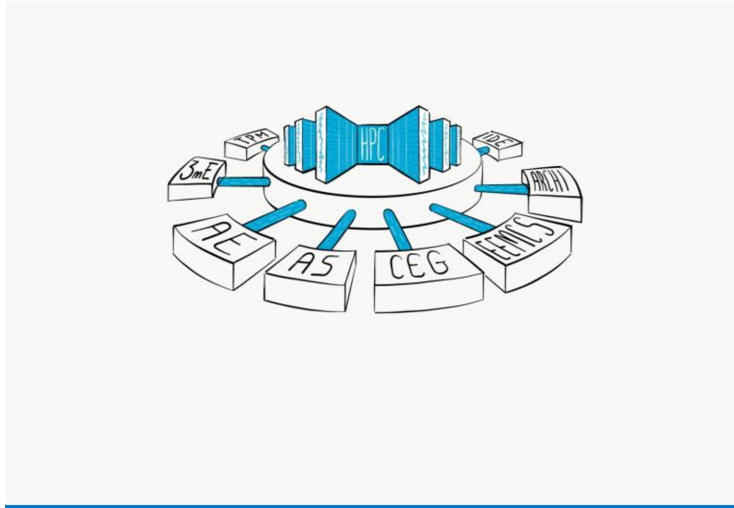
Software/hardware Exploration



DeepXDE: A Deep Learning Library for Solving Differential Equations*

Lu Lu[†]
Xuhui Meng[‡]
Zhiping Mao[§]
George Em Karniadakis[¶]

Abstract. Deep learning has achieved remarkable success in diverse applications; however, its use in solving partial differential equations (PDEs) has emerged only recently. Here, we present an overview of physics-informed neural networks (PINNs), which embed a PDE into the loss of the neural network using automatic differentiation. The PINN algorithm is simple, and it can be applied to different types of PDEs, including integro-differential equations, fractional PDEs, and stochastic PDEs. Moreover, from an implementation point of view, PINNs solve inverse problems as easily as forward problems. We propose a new residual-based adaptive refinement (RAR) method to improve the training efficiency of PINNs. For pedagogical reasons, we compare the PINN algorithm to a standard finite element method. We also present a Python library for PINNs, DeepXDE, which is designed to serve both as an educational tool to be used in the classroom as well as a research tool for solving problems in computational science and engineering. Specifically, DeepXDE can



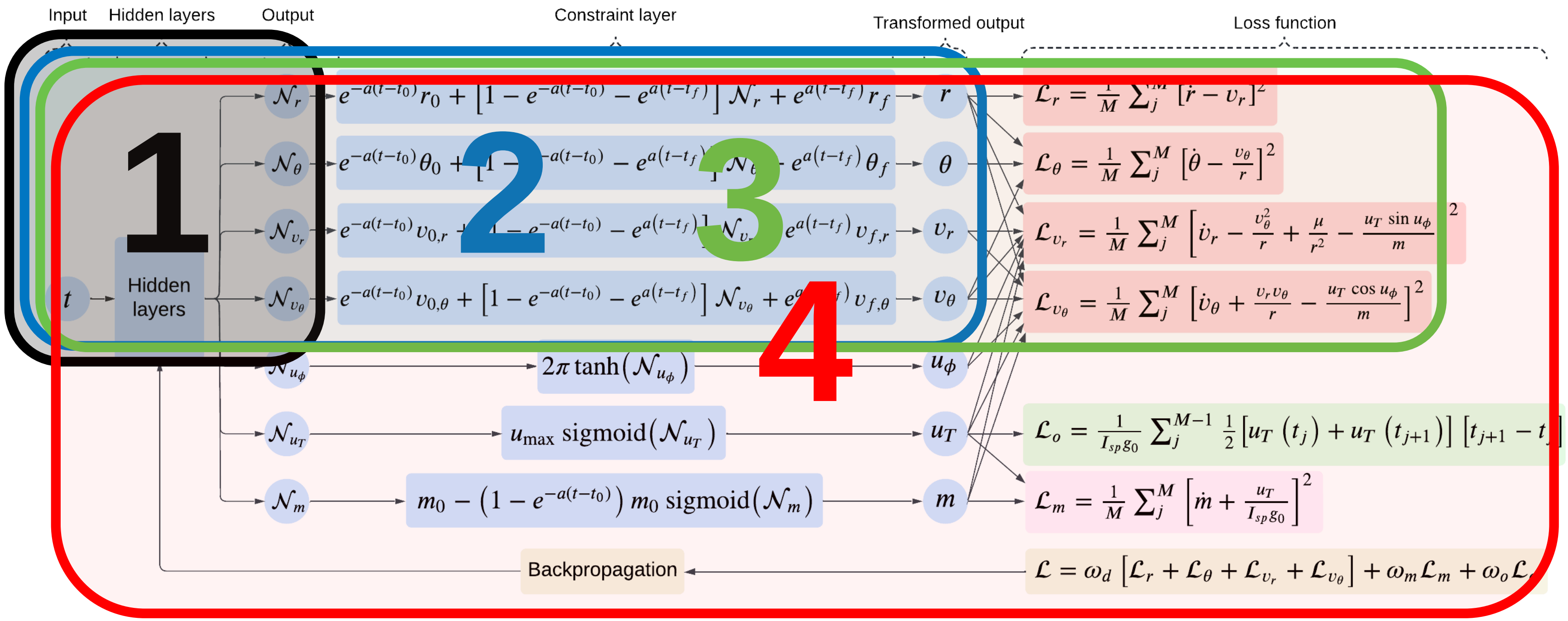
TensorFlow

DeepXDE

Delft Blue

The TU Delft supercomputer

Basic PCNN implementation



1. NN supervised
2. NN supervised + constraint layer
3. NN unsupervised + constraint layer + custom loss function (dynamics only)
4. NN unsupervised + constraint layer + custom loss function (dynamics + objective + mass)

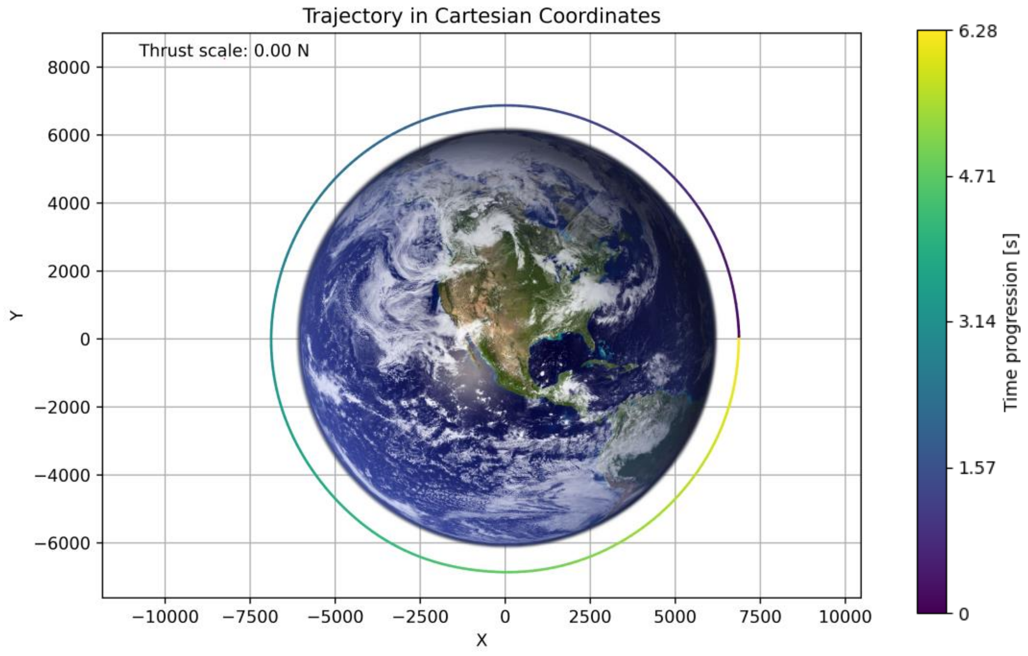
Toy problem:

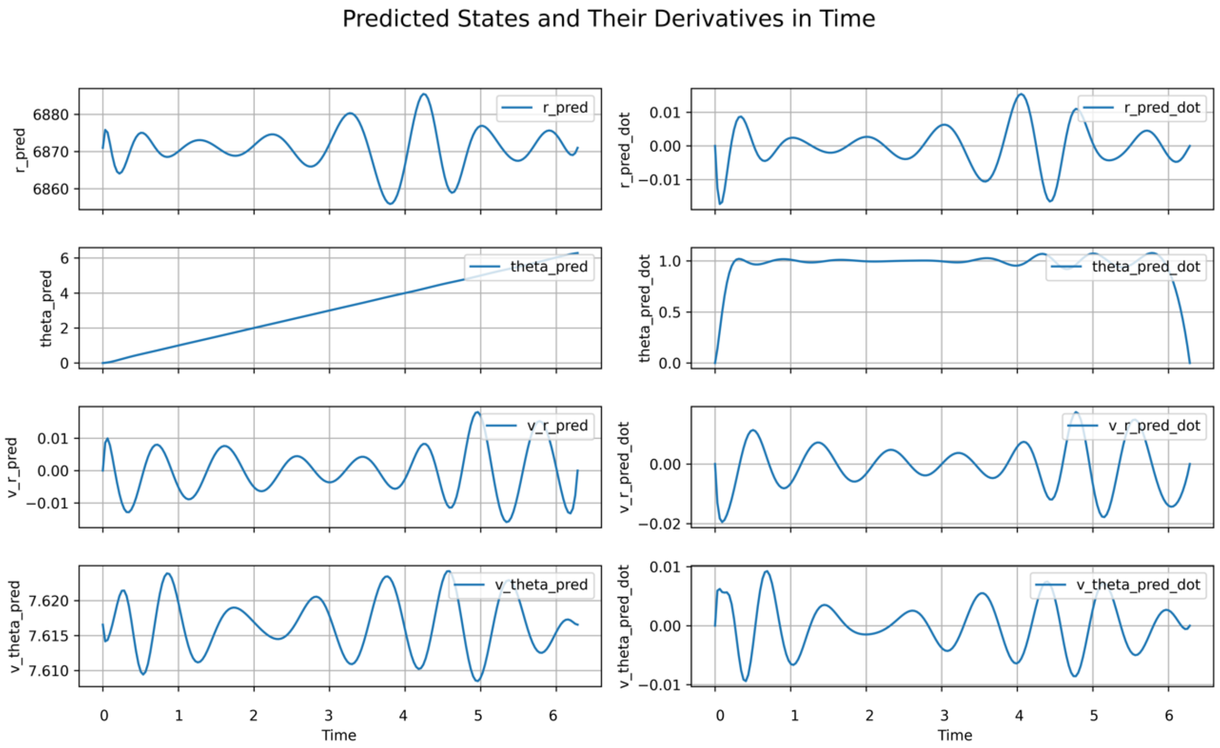
Circular Earth orbit (500km altitude)

Setting	Baseline Configuration
Neurons	**20**
Hidden Layers	**5**
Activation	sin
Training points M	200
Learning rate Schedule	1
$[\omega_d \quad \omega_m \quad \omega_o]$	$[1 \quad 10^{-5} \quad 10^{-7}]$
a	10

Results

1. NN supervised
2. NN supervised + constraint layer

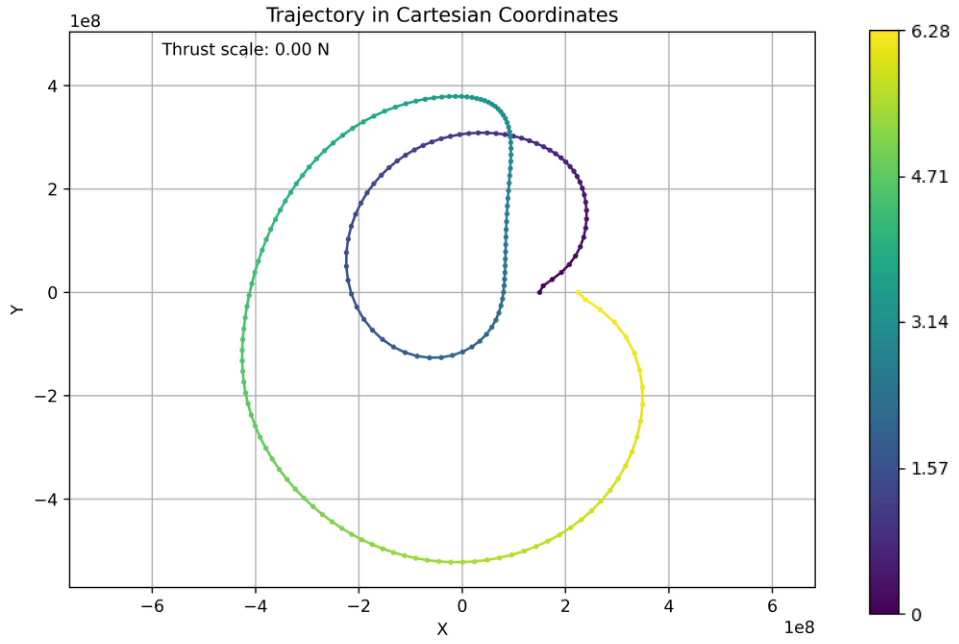


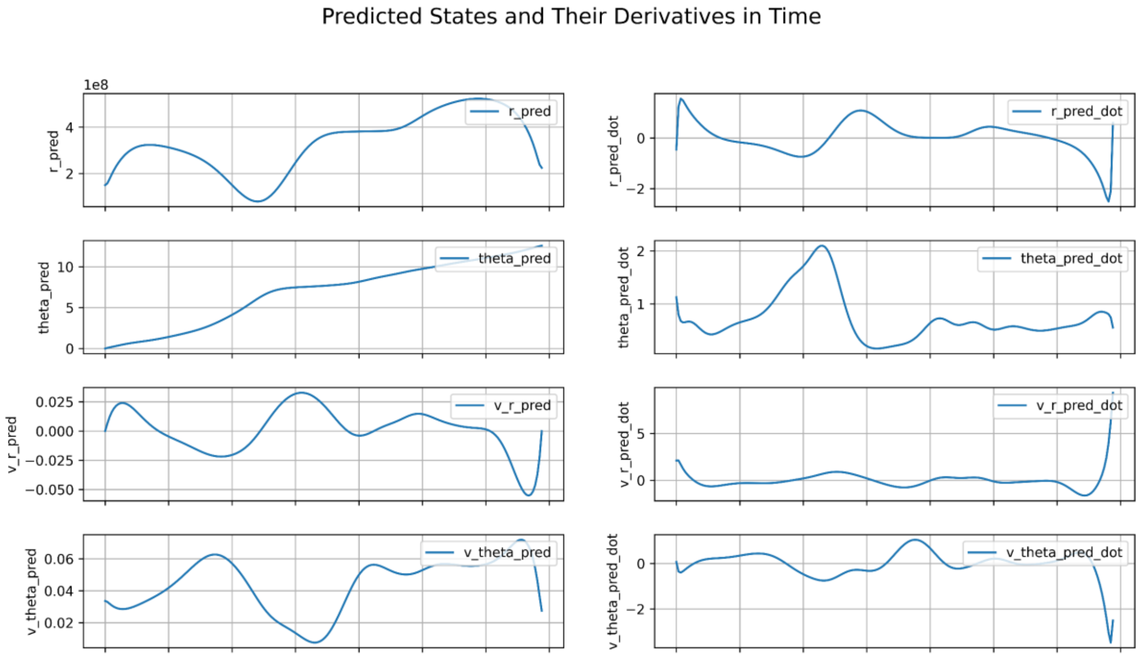


$$v = \sqrt{\frac{GM}{r}} = \sqrt{\frac{\mu}{r}}$$

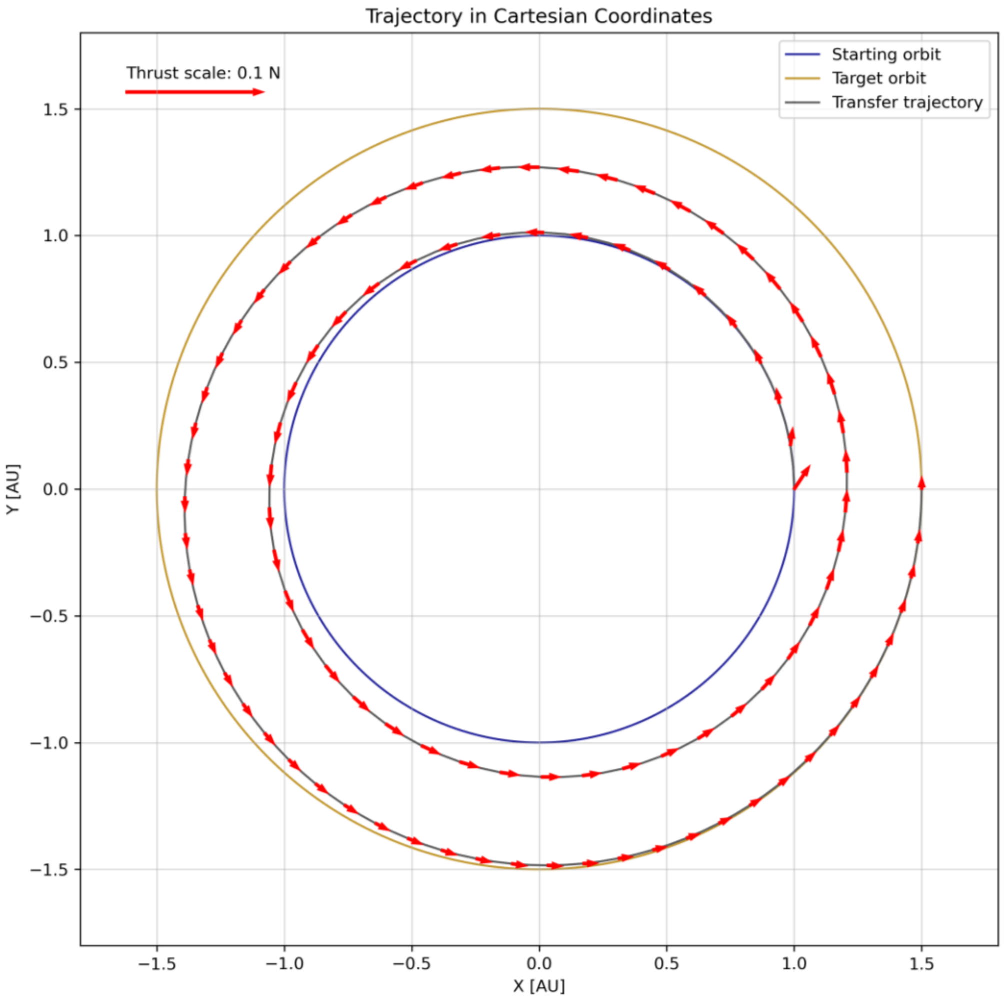
Results

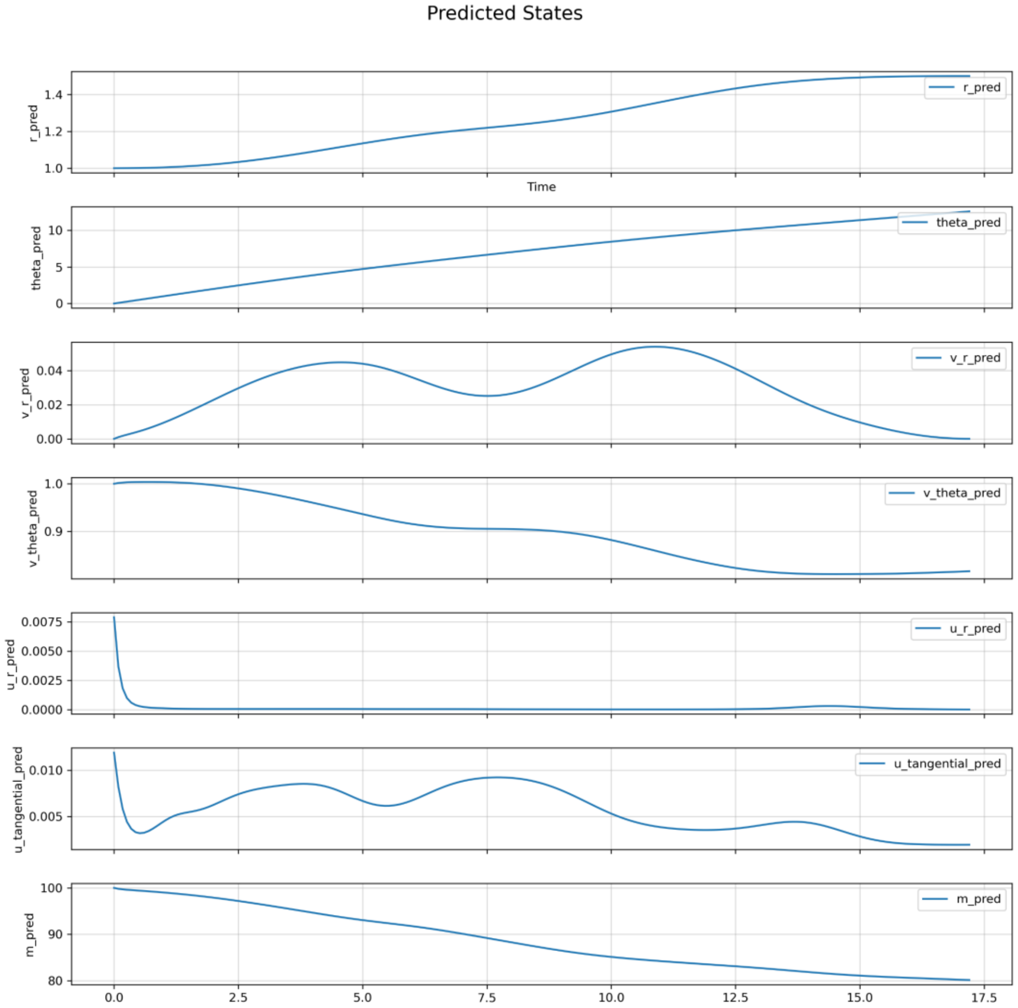
3. NN unsupervised + constraint layer + custom loss function (dynamics only)
4. NN unsupervised + constraint layer + custom loss function (dynamics + objective + mass)



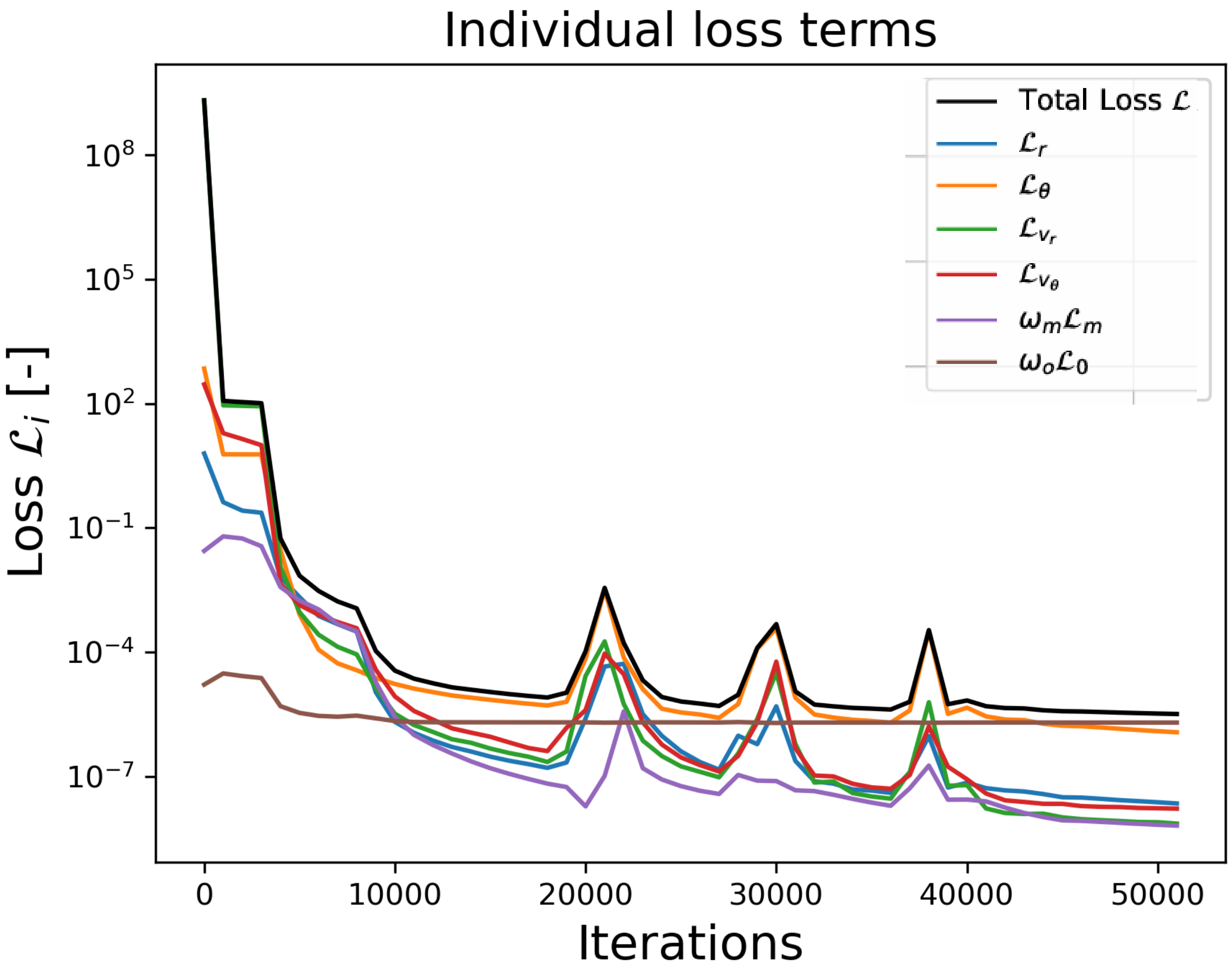


Results with DeepXDE - Trajectory

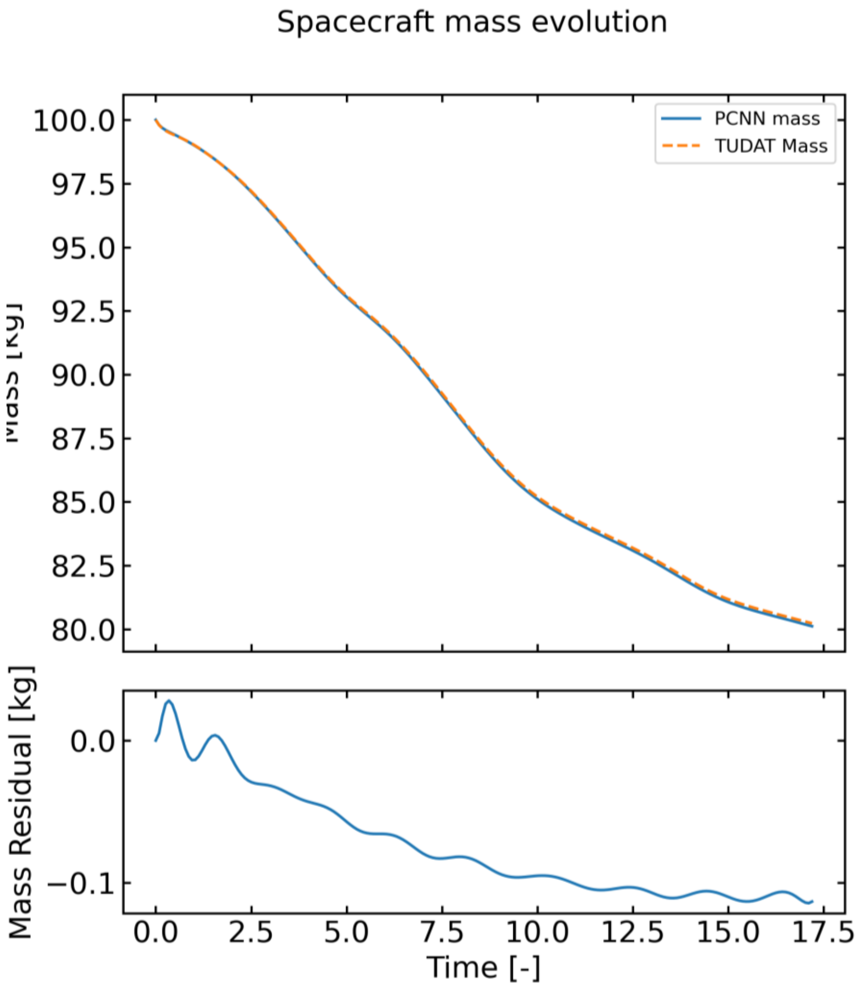


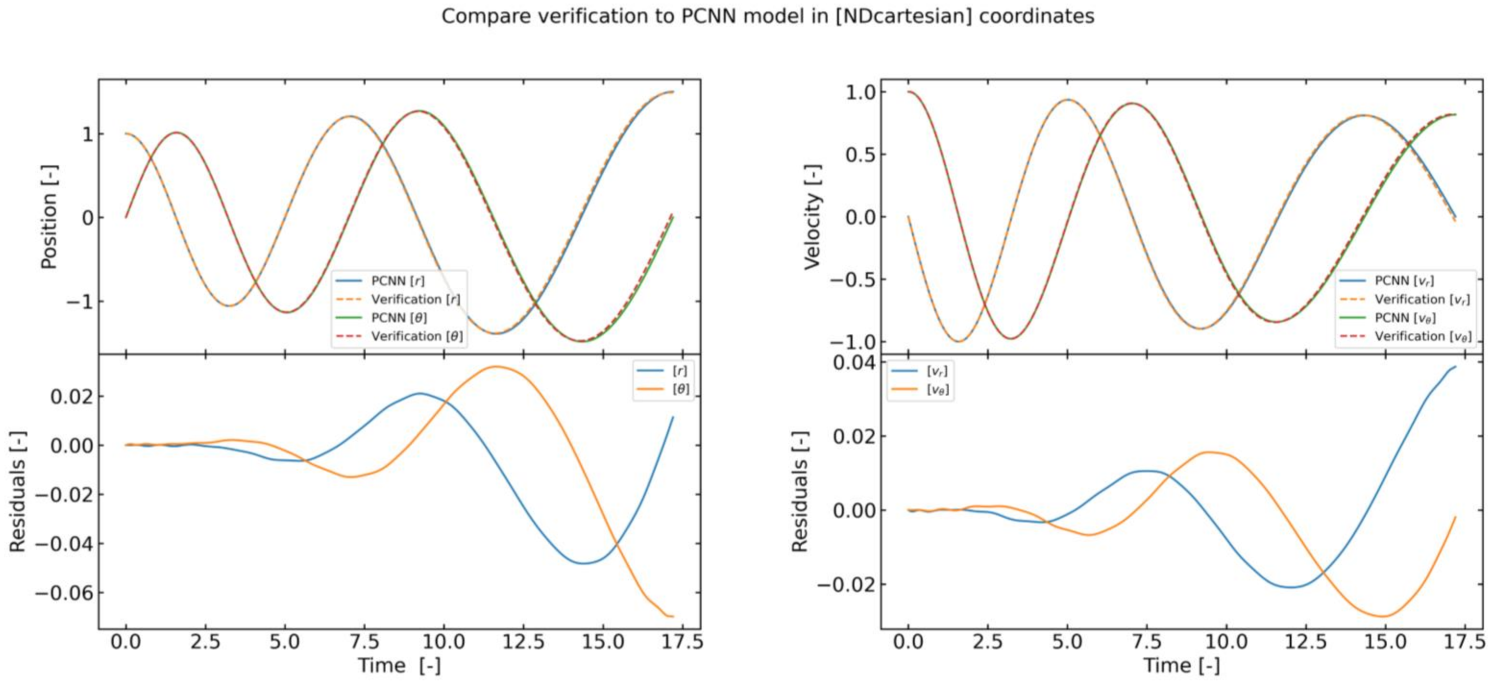


Results with DeepXDE – Loss evolution



Results with DeepXDE – Comparison with Tudat





Numerical integrator used:

Runge-Kutta 7(8) variable step-size integrator

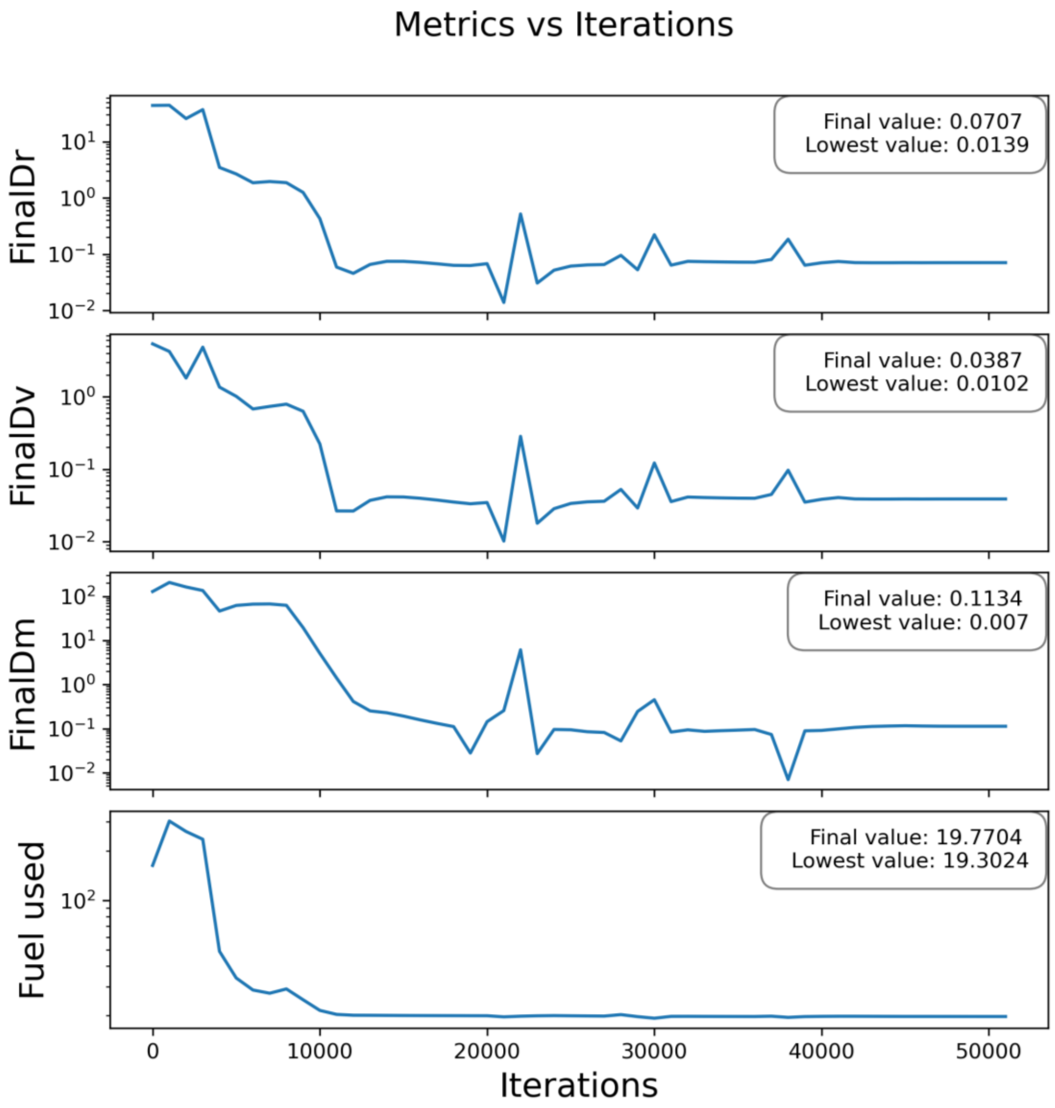
Rel. & abs. tolerances: 10e-10

Results with DeepXDE – Metrics

$$dx = \sqrt{\left(x(t_f) - x_v(t_f)\right)^2 + \left(y(t_f) - y_v(t_f)\right)^2}$$

$$dv = \sqrt{\left(v_x(t_f) - v_{x,v}(t_f)\right)^2 + \left(v_y(t_f) - v_{y,v}(t_f)\right)^2}$$

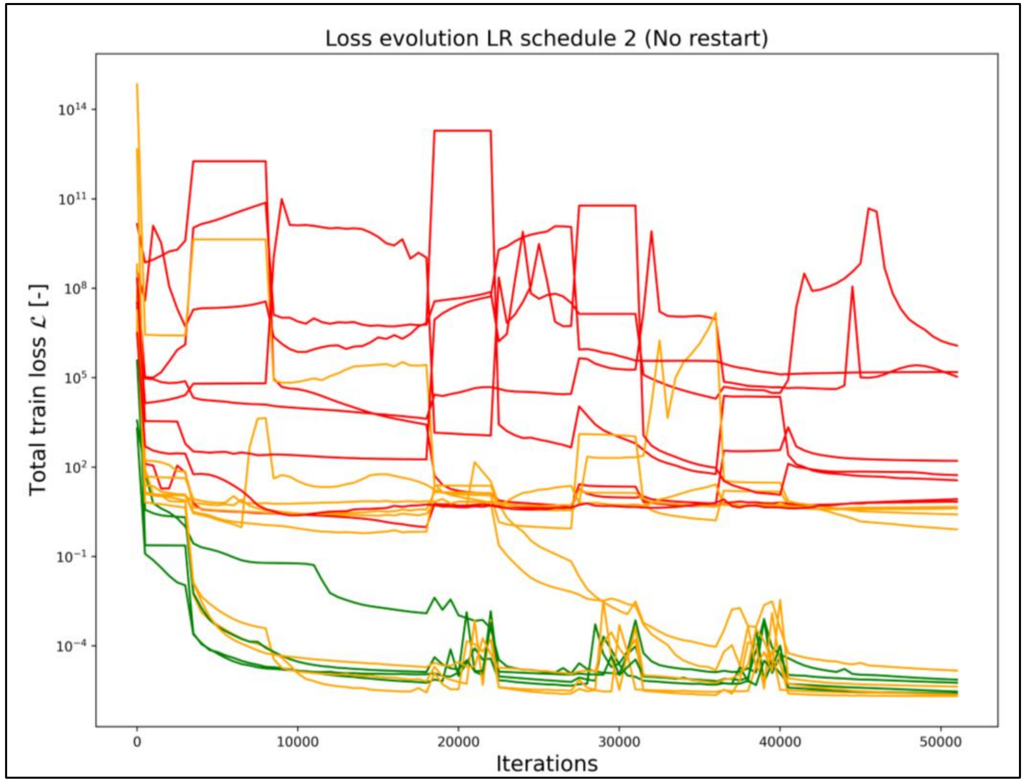
$$dm = m(t_f) - m_v(t_f)$$



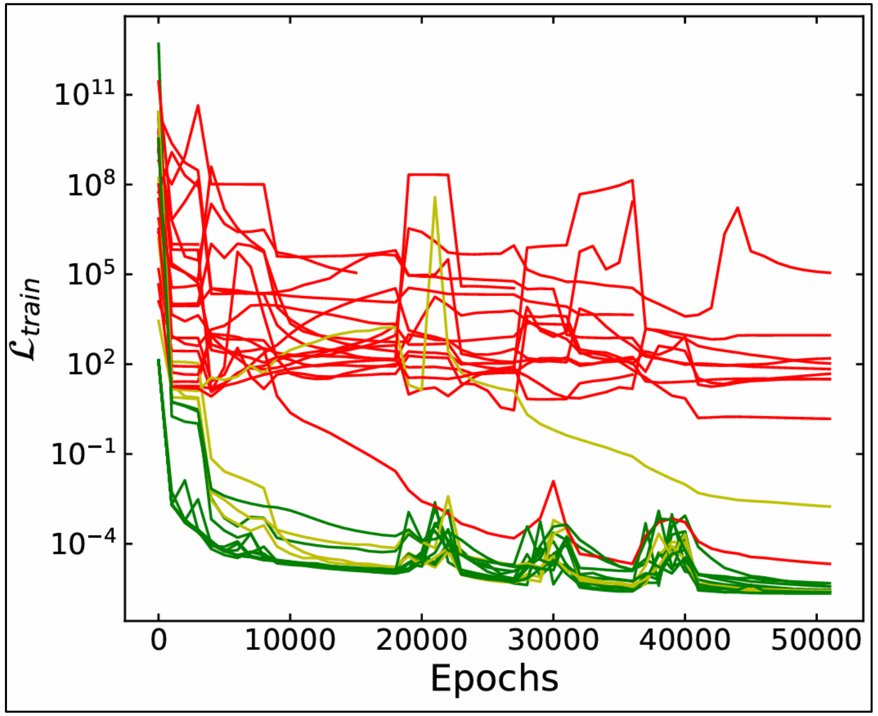
Verification & Validation

Loss evolutions

Schedule 2 | $[10^{-2}, 3\text{k}] \rightarrow [10^{-3}, 5\text{k}] \rightarrow [10^{-4}, 10\text{k}]$
$\rightarrow [5 \times 10^{-3}, 4\text{k}] \rightarrow [10^{-4}, 5\text{k}]$
$\rightarrow [5 \times 10^{-3}, 4\text{k}] \rightarrow [10^{-4}, 5\text{k}]$
$\rightarrow [5 \times 10^{-3}, 4\text{k}] \rightarrow [10^{-4}, 5\text{k}] \rightarrow [10^{-5}, 6\text{k}]$



This work

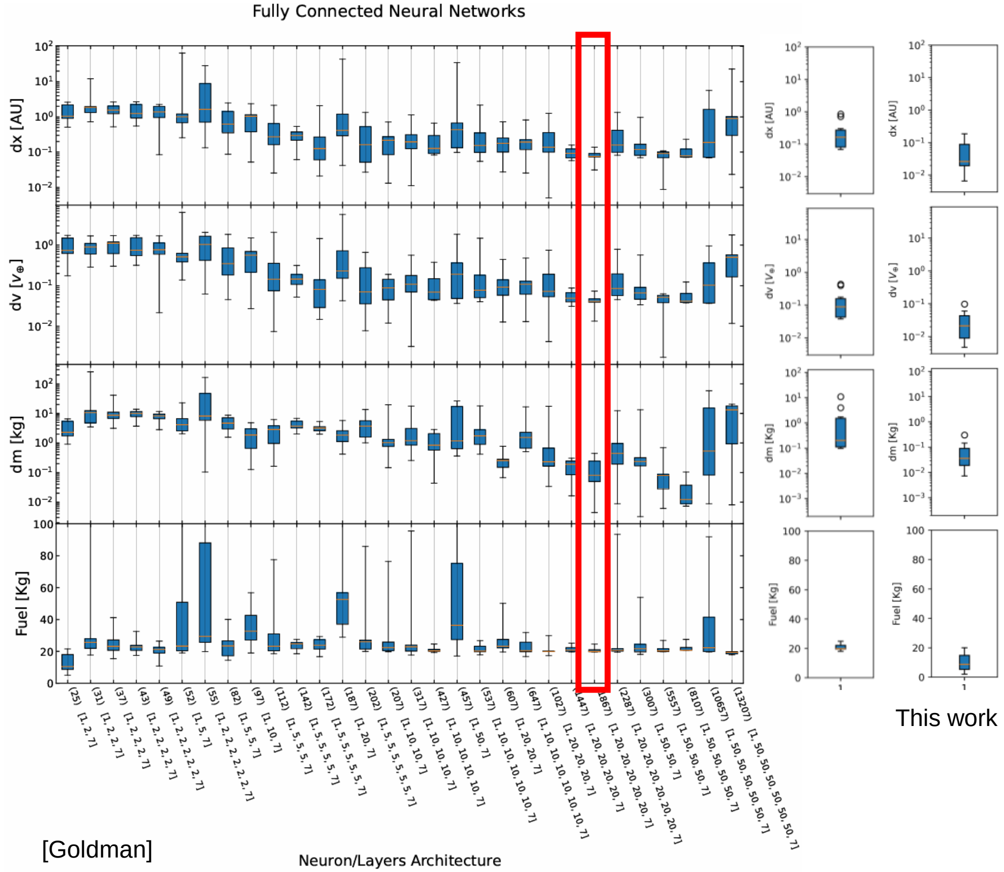


Goldman's work

Verification & Validation

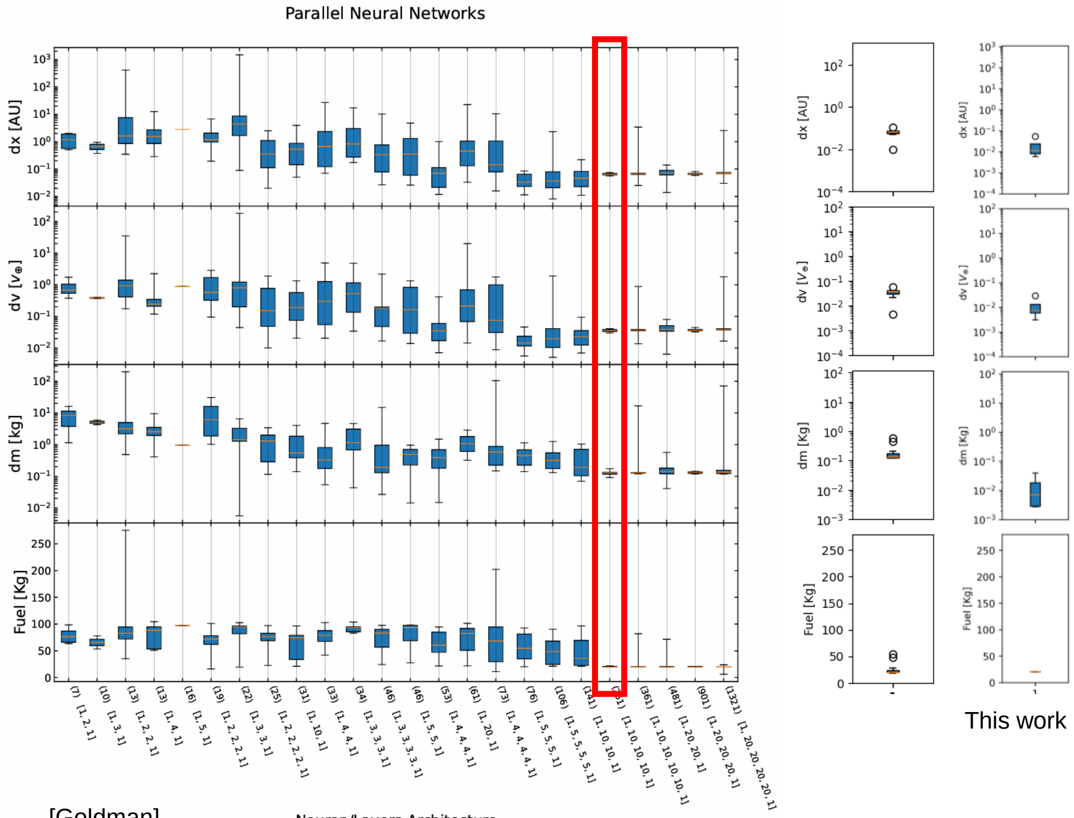
Metrics

Fully connected NNs



[Goldman]

Bundle of 7 parallel NNs



[Goldman]

Verification & Validation

Metrics

Median values comparison

Latest iteration

	FNN Goldman [1, 20, 20, 20, 20, 20, 7]	**FNN this work** [1, 20, 20, 20, 20, 20, 7]	**Bundle Goldman** [1, 10, 10, 10, 10, 1]	**Bundle this work** [1, 10, 10, 10, 10, 1]
Median dx [AU]	0.076	0.085	0.064	0.071
Median dv [$v_\oplus$]	0.041	0.045	0.035	0.0389
Median dm [kg]	0.081	0.123	0.129	0.1375
Median propellant mass m_p [kg]	19.93	20.04	19.96	20.53

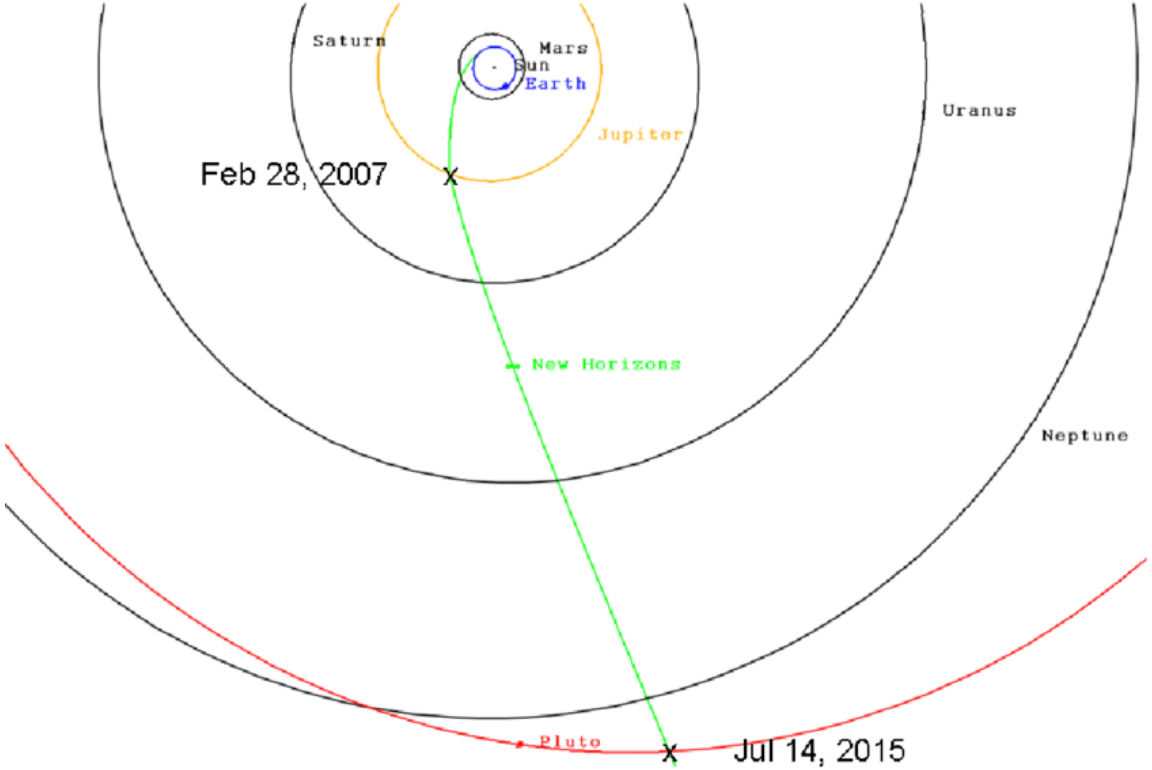
Lowest metrics:

	FNN Goldman [1, 20, 20, 20, 20, 20, 7]	**FNN this work** [1, 20, 20, 20, 20, 20, 7]	**Bundle Goldman** [1, 10, 10, 10, 10, 1]	**Bundle this work** [1, 10, 10, 10, 10, 1]
Median dx [AU]	0.076	0.029	0.064	0.011
Median dv [$v_\oplus$]	0.041	0.019	0.035	0.0064
Median dm [kg]	0.081	0.008	0.129	0.037
Median propellant mass m_p [kg]	19.93	13.77	19.96	20.27

Pre-liminary results subproblem 1

~~Subproblem 1: Earth → Jupiter rendezvous (New Horizons missions)~~

- Fixed x_0, x_f, ToF

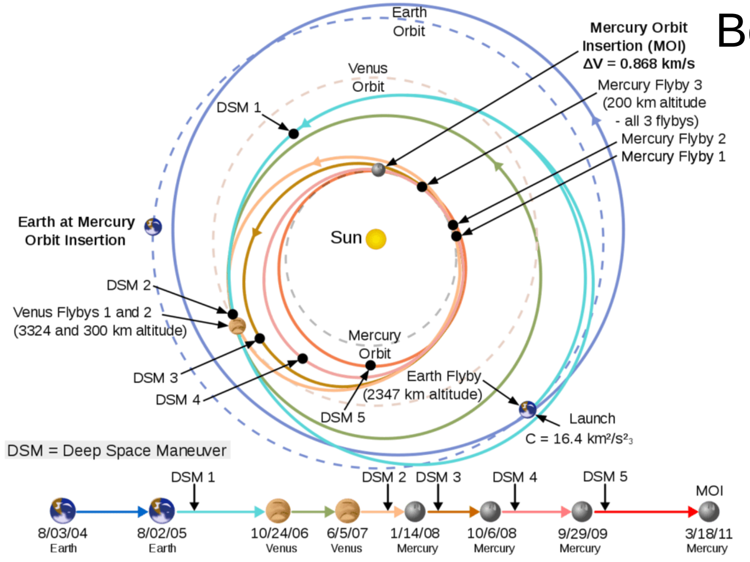


Pre-liminary results subproblem 1

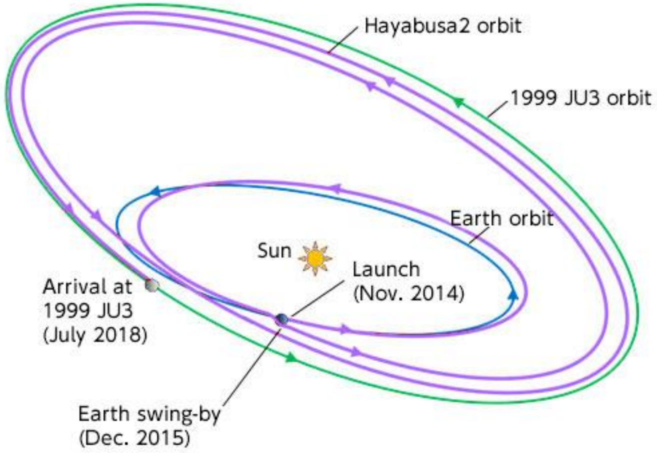
New Subproblem 1:

- Fixed x_0, x_f, ToF
- Requirements:
  - Involves gravity assists essential to reach the target destination.
  - Utilizes low-thrust propulsion (e.g., ion engines or solar electric propulsion)
  - Has publicly available trajectory data, including the spacecraft's state history

- Choice: BepiColombo



BepiColombo

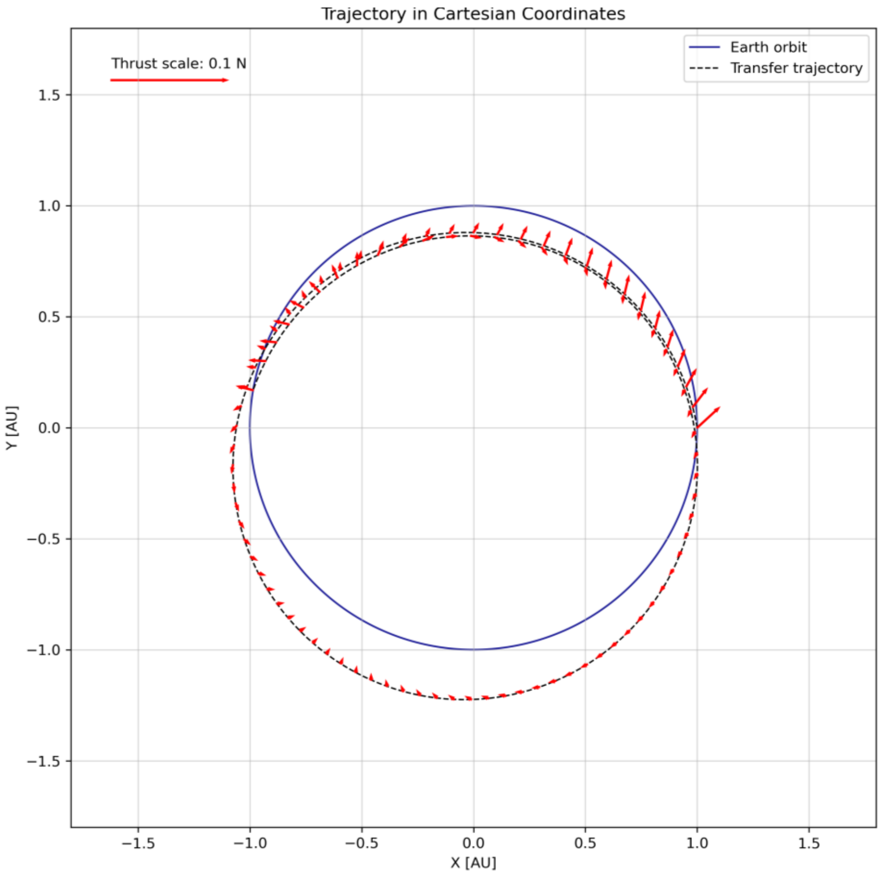


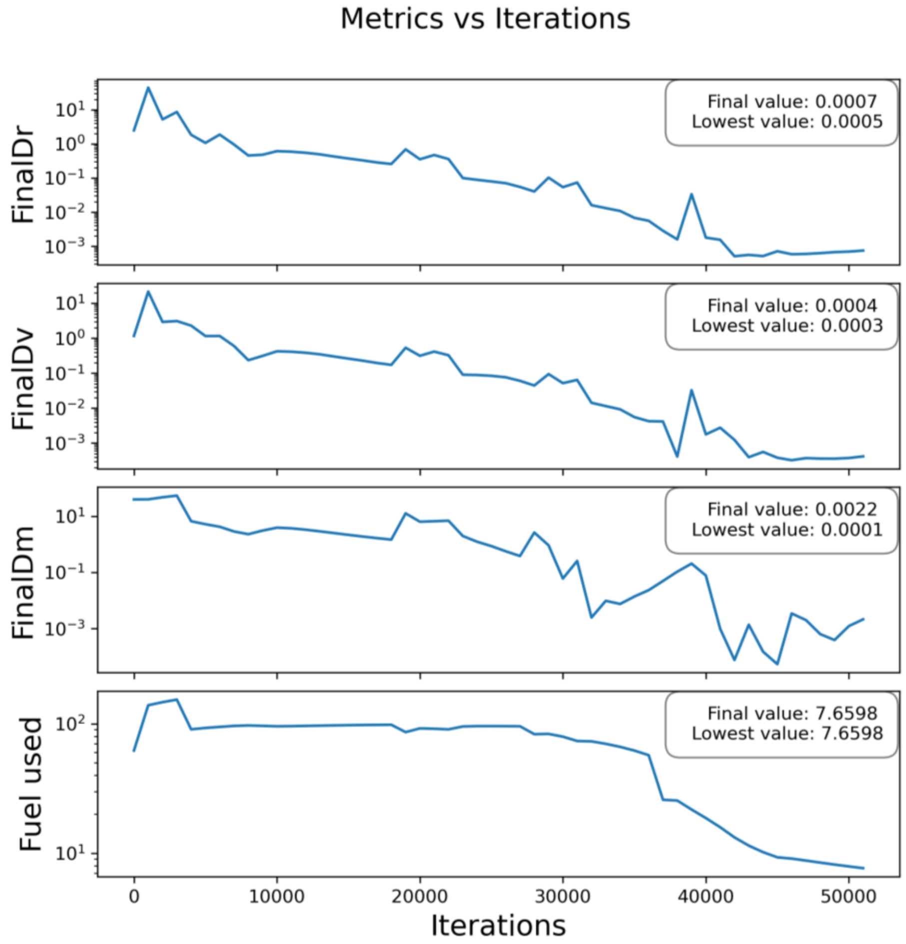
Hayabusa2

Pre-liminary results subproblem 1

Subproblem 1: Earth → Earth rendezvous

Trajectory leg 1

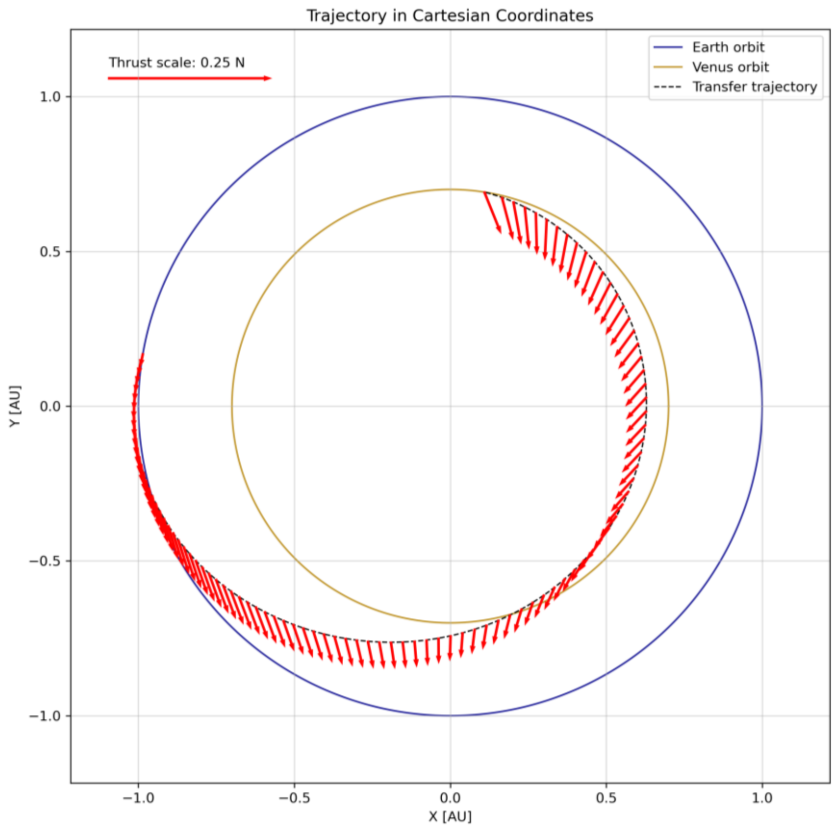


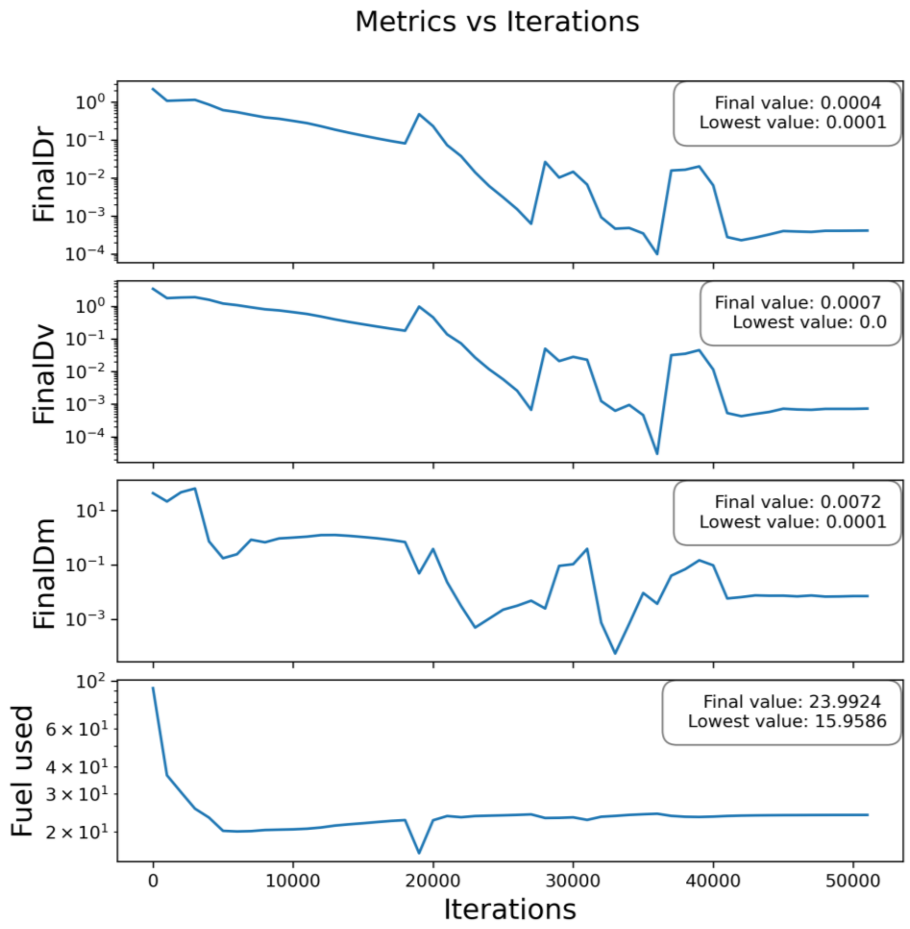


Pre-liminary results subproblem 1

Subproblem 1: Earth → Earth rendezvous

Trajectory leg 2



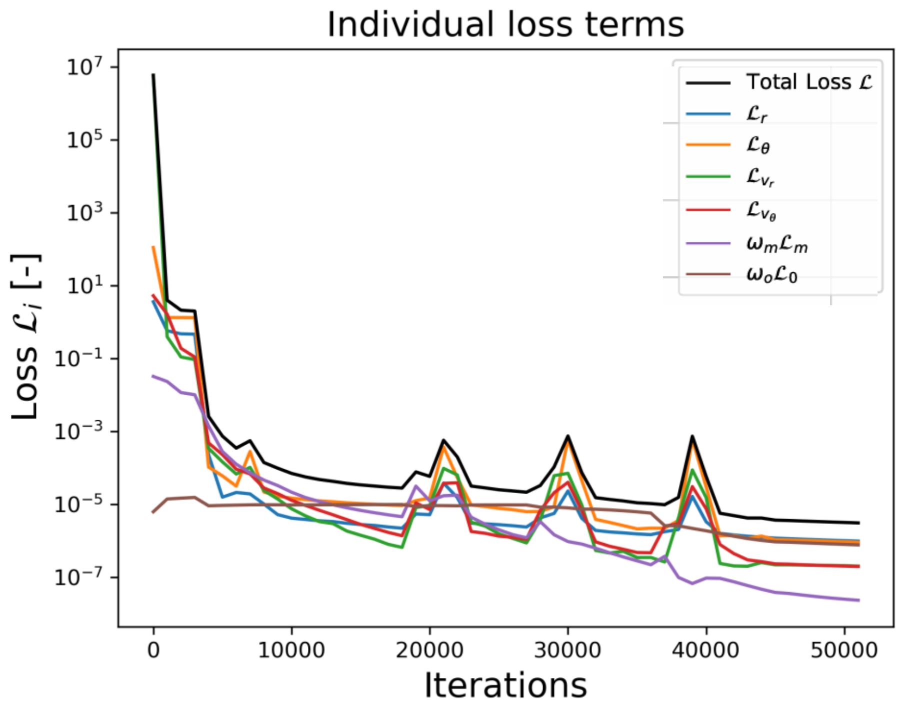


Pre-liminary results subproblem 1

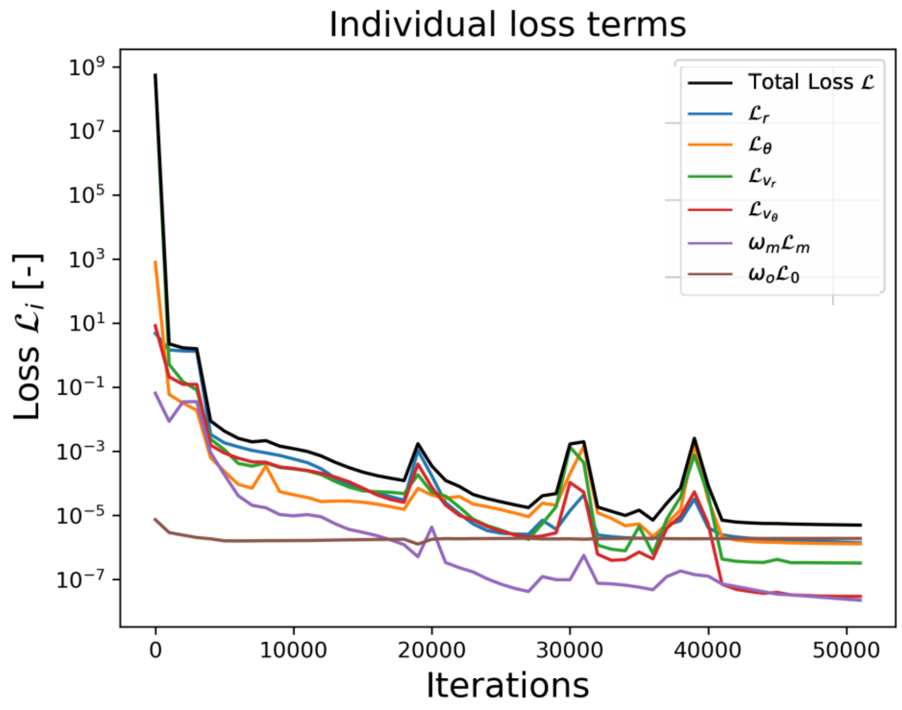
Subproblem 1: Earth → Earth rendezvous

Loss evolutions

Leg 1



Leg 2

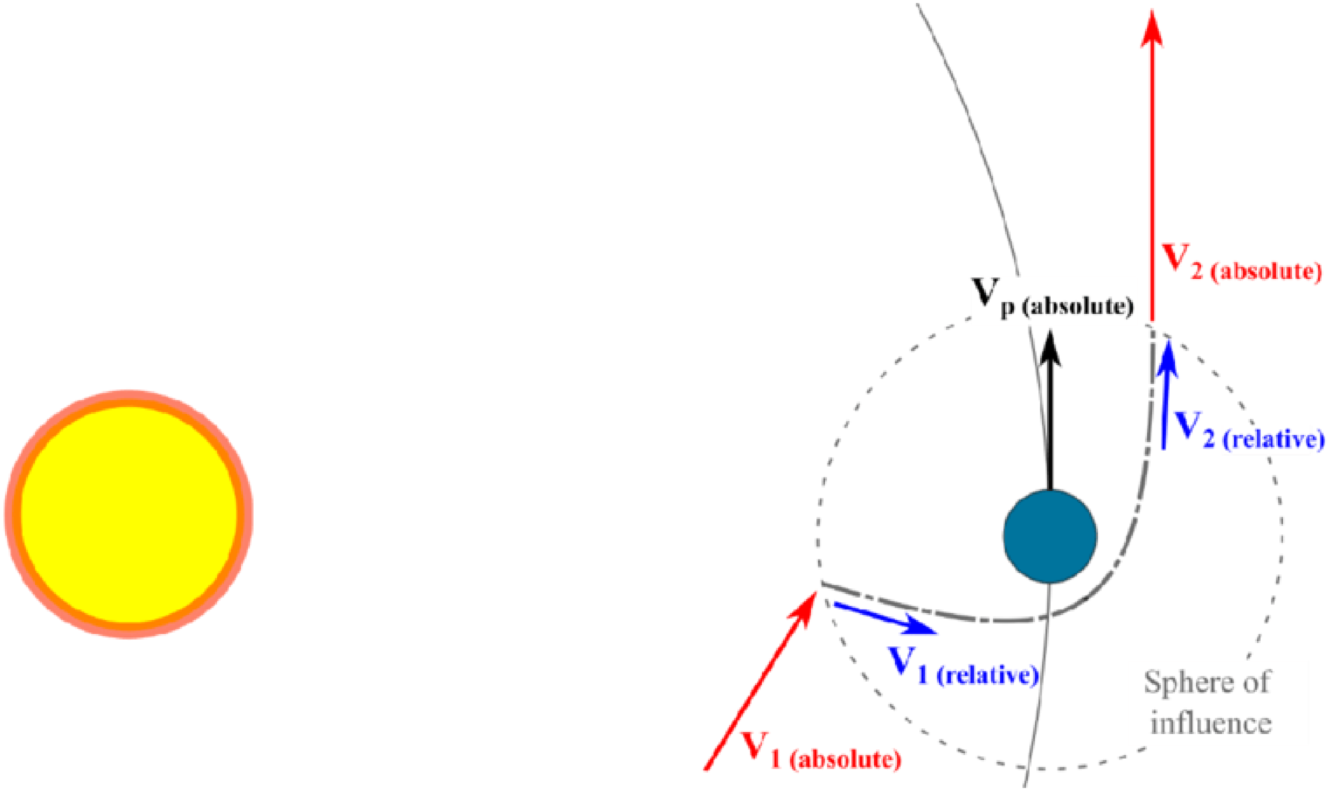


3

Analysis and reflection of results

Analysis and reflection of results

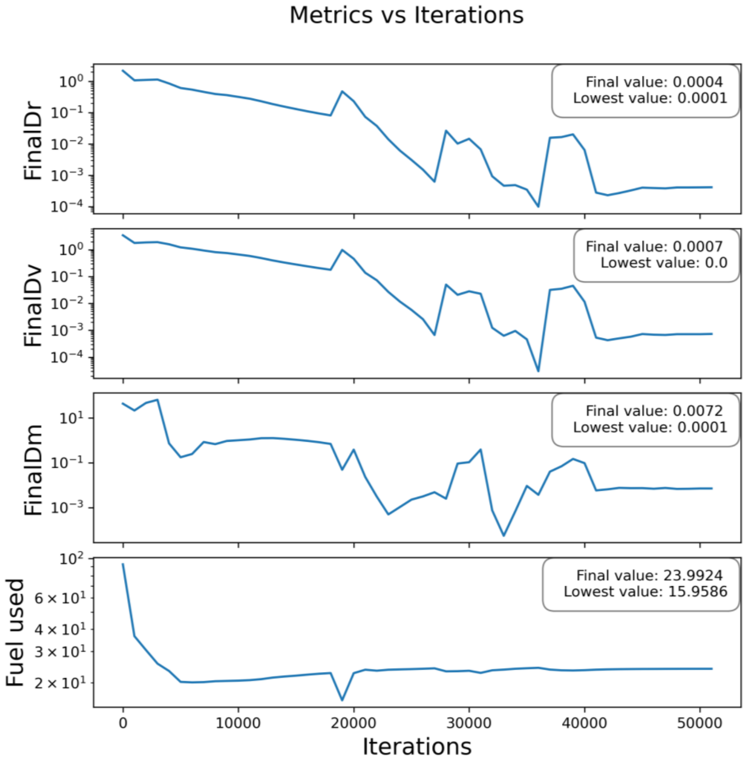
- Accuracy of the method



Analysis and reflection of results

- Accuracy of the method

0.0001 AU = 15 000 km
0.0005 AU = 75 000 km
0.001 AU = 150 000 km
0.01 AU = 1 500 000 km



0.0640 AU = 9 600 000 km

SOI Earth = 924 000 km

SOI Jupiter = 48 200 000

Lowest metrics:

	FNN Goldman [1, 20, 20, 20, 20, 20, 7]	FNN this work [1, 20, 20, 20, 20, 20, 7]	Bundle Goldman [1, 10, 10, 10, 10, 1]	Bundle this work [1, 10, 10, 10, 10, 1]
Median dx [AU]	0.076	0.029	0.064	0.011
Median dv [$v_\oplus$]	0.041	0.019	0.035	0.0064
Median dm [kg]	0.081	0.008	0.129	0.037
Median propellant mass m_p [kg]	19.93	13.77	19.96	20.27

Latest iteration

	FNN Goldman [1, 20, 20, 20, 20, 20, 7]	FNN this work [1, 20, 20, 20, 20, 20, 7]	Bundle Goldman [1, 10, 10, 10, 10, 1]	Bundle this work [1, 10, 10, 10, 10, 1]
Median dx [AU]	0.076	0.085	0.064	0.071
Median dv [$v_\oplus$]	0.041	0.045	0.035	0.0389
Median dm [kg]	0.081	0.123	0.129	0.1375
Median propellant mass m_p [kg]	19.93	20.04	19.96	20.53

Analysis and reflection of results

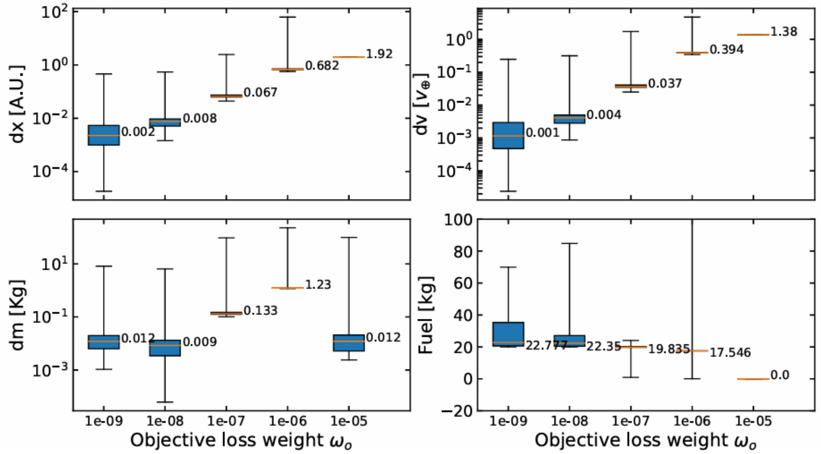
- Accuracy of the method

0.0001 AU = 15 000 km
0.0005 AU = 75 000 km
0.001 AU = 150 000 km
0.01 AU = 1 500 000 km

Fixes:

- Lower objective weight

- Try different activation function



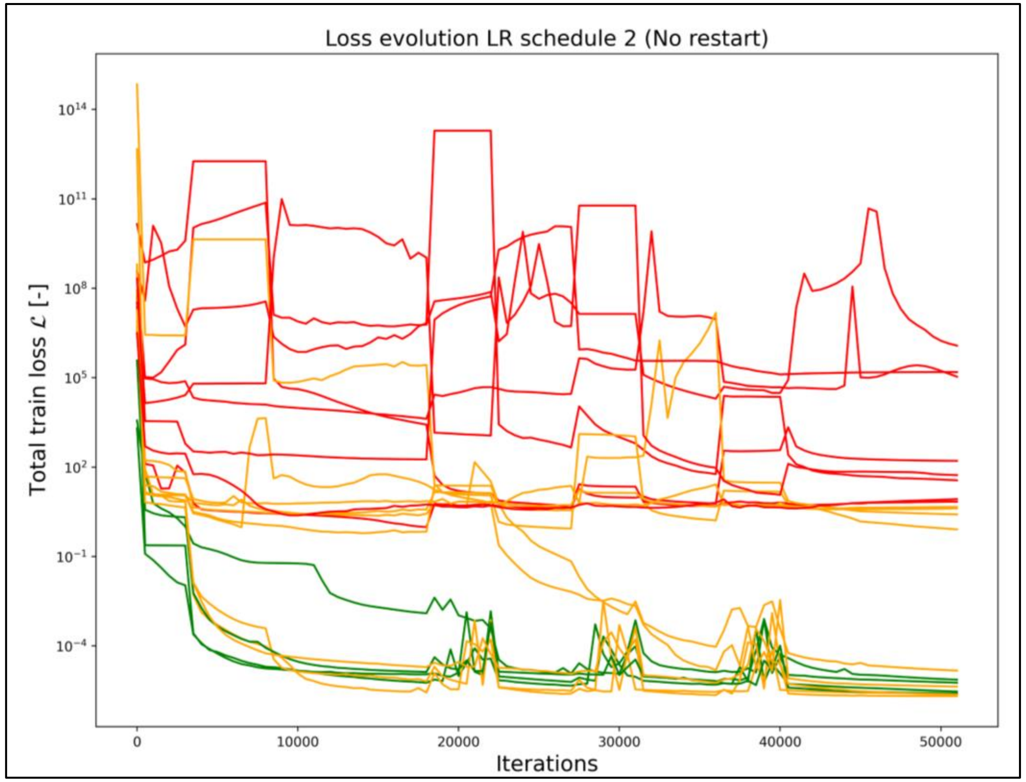
0.0640 AU = 9 600 000 km

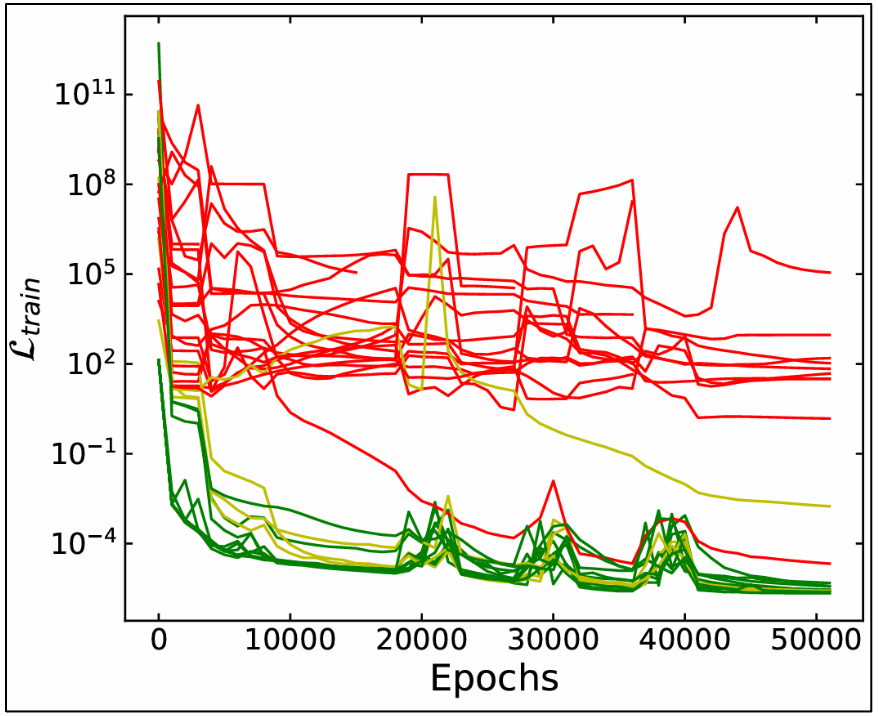
SOI Earth = 924 000 km

SOI Jupiter = 48 200 000

Analysis and reflection of results

- Dependence on initialization of NN





Analysis and reflection of results

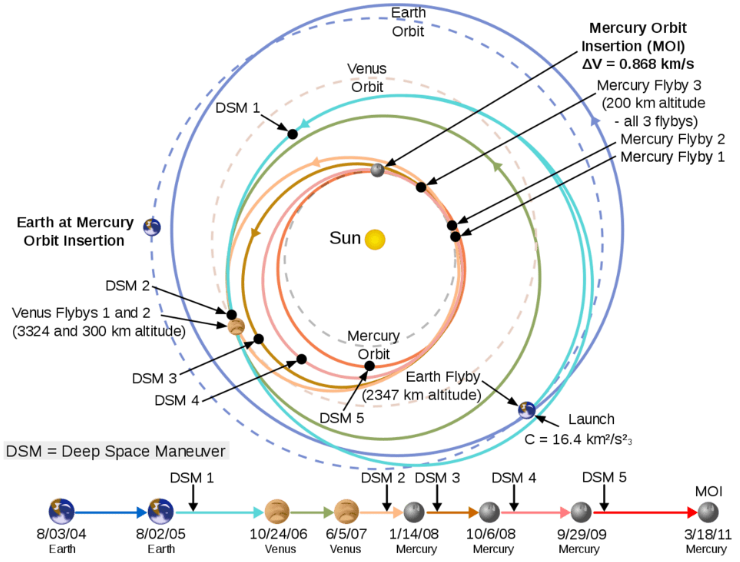
V&V subproblem 1 + 2

Verification

- Numerical integrations Tudat
- Data BepiColombo

Validation

- Hodographic shaping
- Data BepiColombo?



Note

Generated by nbsphinx from a Jupyter notebook. All the examples as Jupyter notebooks are available in the tudatpy-examples repo.

Hodographic-shaping MGA transfer optimization with PyGMO

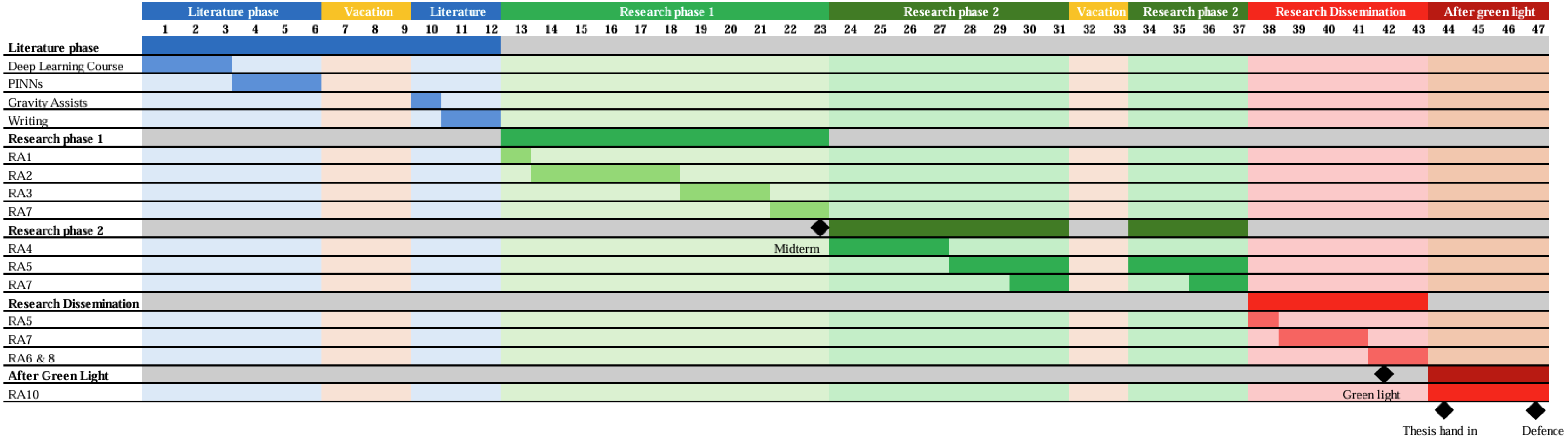
Context

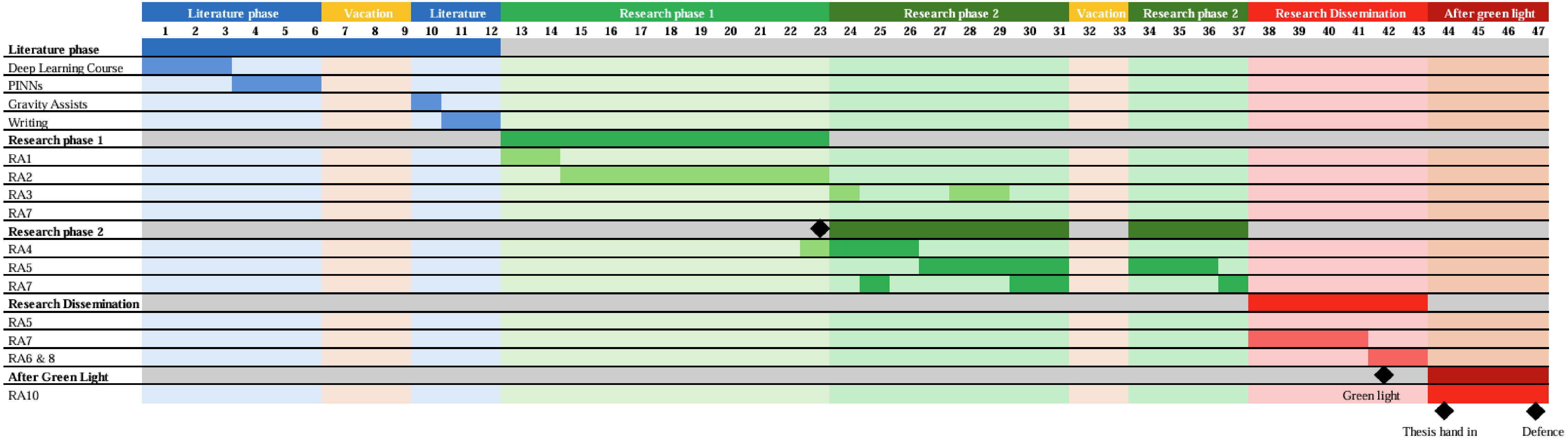
This example illustrates the usage of PyGMO to optimize a low-thrust interplanetary transfer trajectory simulated using the multiple gravity assist (MGA) module of Tudat. The low-thrust legs are simulated using hodographic shaping. The spacecraft is considered to depart from the edge of the sphere of influence (SOI) of the Earth, execute swingbys at the Mars and Earth, and arrive at the edge of Jupiter's SOI. Thus, the transfer includes 3 legs and 2 swingbys.

4

Updated plan for remainder of thesis

Updated plan for remainder of thesis





5

Q&A and Feedback

Q&A and feedback

Personal doubts/ points were I need feedback on:

1. Accuracy of solutions
2. Case scenario
3. V&V
4. Omit mass from output NN

Thank you!