Machine Learning and Evolutionary Techniques in Interplanetary Trajectory Design

Dario Izzo, Christopher Iliffe Sprague, and Dharmesh Vijay Tailor

Abstract After providing a brief historical overview on the synergies between artificial intelligence research, in the areas of evolutionary computations and machine learning, and the optimal design of interplanetary trajectories, we propose and study the use of deep artificial neural networks to represent, on-board, the optimal guidance profile of an interplanetary mission. The results, limited to the chosen test case of an Earth–Mars orbital transfer, extend the findings made previously for landing scenarios and quadcopter dynamics, opening a new research area in interplanetary trajectory planning.

1 Introduction

The use of artificial intelligence (AI) techniques for the design of interplanetary trajectories, in particular in their early design phases, has been proposed and studied by numerous scientists over the past few decades. Among the AI techniques deployed to help the optimisation of spacecraft trajectories are evolutionary algorithms [15, 22, 23, 28, 31, 35, 48, 52], machine learning techniques [2, 4, 27, 33], evolutionary neuro-controllers [6, 7], tree search methods [17, 18, 23, 27, 50] to only name a few widely studied methods. While the synergies between research in AI and in trajectory design were becoming increasingly apparent, the field of AI as a whole experienced a renaissance in the second decade of the millennium, delivering exciting developments that are of significance to many scientific fields. Certainly the area of machine learning is one that is powering such an AI renaissance, in particular with deep learning techniques [47] becoming fundamental to define

D. Izzo (✉) · D. V. Tailor
European Space Agency, Noordwijk, The Netherlands
e-mail: dario.izzo@esa.int; dharmesh.tailor@live.co.uk

C. I. Sprague
KTH Royal Institute of Technology, Stockholm, Sweden
e-mail: sprague@kth.se

G. Fasano, J. D. Pintér (eds.), *Modeling and Optimization in Space Engineering*, Springer Optimization and Its Applications 144,
https://doi.org/10.1007/978-3-030-10501-3_8

new benchmarks in the most diverse applications. It thus seems likely that also the optimisation of interplanetary trajectories, already receptive of AI methods in general, is going to benefit from the vast amount of new knowledge being produced in the context of AI research.

In this chapter we first briefly provide an overview on the state-of-the art of the use of evolutionary techniques (Sect. 2) and machine learning techniques (Sect. 3) when it comes to optimising interplanetary trajectories, and we then introduce a novel idea concerning the application of deep learning for the on-board representation of the optimal guidance profile for an interplanetary probe, extending previous work made on planetary landing scenarios [42–44].

2 Evolutionary Algorithms for Trajectory Planning: An Overview

Evolutionary algorithms are a class of global optimisation techniques that make use of heuristic rules, often inspired but not limited to natural paradigms such as Darwinian evolution, to search for optimal solutions in discontinuous, rugged landscapes. As such they are a good match to solve interplanetary trajectory optimisation problems where the planetary geometry defines a quite complex solution landscape already in simple cases. Already in 1985 a genetic algorithm (GA) [28] was proposed and studied in the context of interplanetary trajectory optimisation concluding that

> ...a considerable effort is still needed for developing efficient schemes using genetic algorithms. However, they appear to offer an entirely original way for solving a large class of [interplanetary trajectory] global optimisation problems ...

Given the CPU power available at the time, it is only natural that this pioneering work complained about efficiency. In the following decade the "entirely original way" was consolidated and more successful applications of GAs for both low-thrust propelled spacecraft [40] and impulsive thrust strategies [15] were deployed. Studies on the use of some form of GA, also leveraging on the ever increasing computational power available, continued (see for example [3, 9, 12, 14, 41]) showing how the original intuition, back in 1985, was one rich of consequences. It was indeed relatively consequential to substitute the GA with any other form of stochastic optimiser, so that in 2004 the European Space Agency performed a series of studies, in collaboration with academia, to test and compare several global optimisers on interplanetary trajectory problems [10, 22, 34]. It then became clear how other evolutionary algorithms were offering alternative, and in many ways superior, choices to help the preliminary phases of trajectory design. Among them, differential evolution (DE) and particle swarm optimisation (PSO) were identified and benchmarked on several chemical propulsion problems opening the way to further independent confirmations of their performances [32, 35, 37, 48, 52, 55].

Interestingly, in 2013, a self-adaptive differential evolution algorithm was used to plan the grand tour of the Jupiter moons that was awarded the golden Humies medal [23] for "human-competitive results produced by genetic and evolutionary computation". Many other evolutionary approaches have been proved to be of use by various researchers during the early 2000s, including simulated annealing (SA) [32] and ant colony optimisation (ACO) [5, 39]. More recently, covariance matrix evolutionary strategy (CMA-ES) was shown to be potentially outperforming self-adaptive DE on a class of transfers [24].

While the no free lunch theorem [54] guarantees that no evolutionary approach is better than any other on average, restricting the class of optimisation problems to those representing interplanetary transfers allows to identify the most useful approaches. To this end, a set of problems called GTOP database [53] (a sort of open trajectory gym) was created and made available to the scientific community and is still the subject of active research [1, 4, 20, 45, 49, 51]. Some of the interplanetary trajectory problems (i.e. Messenger and Cassini2) in the GTOP database were also used during the CEC2011 competition attracting the attention of the larger scientific community of evolutionary computations (see [12] for the competition winner).

Most of the research mentioned so far considers continuous optimisation problems with a single objective and box constraints. In its most general case, though, the problem encountered in interplanetary trajectory design is multi-objective, with nonlinear constraints and, possibly, integer decision variables. Genetic approaches to multi-objective trajectory design were benchmarked already in 2005 [31] followed by deeper studies on non-dominated sorting genetic algorithm (NSGA-II) that included also, as integer variables, the planetary fly-by sequence [9]. Multi-objective versions of PSO have also been considered early on [30]. The more modern multi-objective evolutionary algorithm by decomposition (MOEA/D) was later identified as a most performing technique in applicable cases [24]; in the same work, several constraint handling techniques including co-evolution and immune systems were tested.

Whenever the representation of an interplanetary trajectory requires integer and continuous variables, the resulting optimisation problem (a MINLP) typically becomes intractable also for evolutionary approaches and while some results have been obtained, for example using the ACO paradigm [5, 46], a more convincing approach is to consider the continuous part and the integer part of the problem separately and architect some optimisation scheme tackling the two problems with different nested techniques (bi-level optimisation). Evolutionary approaches based on GAs [13, 25, 26] or ACO [50] have been used in bi-level optimisation schemes, often coupled with smart tree search strategies such as beam search (BS) [17, 50], Monte Carlo tree search (MCTS) [18], multi-objective beam search (MOBS) [27], or lazy race tree search (LRTS) [23] to take care of the integer part.

Evolutionary approaches to interplanetary trajectory planning have proven their worth beyond any criticism and while they are still not as widely used by the aerospace industry as they could, it is likely that we will see a larger penetration in the industrial sector in the upcoming years, as the interest in artificial intelligence

methods powered by deep learning and machine learning will put also evolutionary techniques in the spotlight.

3 Machine Learning and Interplanetary Trajectories

The use of machine learning (ML) algorithms to aid the design of interplanetary trajectory is not as widely researched as that of evolutionary techniques and is limited to fewer works. The reasons are to be found in the less obvious applicability of these methods to the problems encountered in the design of interplanetary trajectories and in the lack of data sets produced and made available by the aerospace community. In this section we try to summarise the ideas that have so far been proposed.

During the optimisation of an interplanetary transfer, as in any optimisation task, a large number of solutions are computed and assessed to inform the search for better candidates. Typically all these design points are discarded and lost after they have been used to produce new promising search directions. The idea of applying supervised learning on such a data set in order to build a model improving the further selection of initial guesses to guide successive evolutionary runs was proposed and tested [4] on some of the trajectory problems in the GTOP database [53] using support vector machines (SVM). Following a similar reasoning, a ML model can be trained, using the points sampled during the evolution, in order to construct a surrogate model of the trajectory worth, which then avoids expensive evaluations of the objective function [2]. Building surrogate models is particularly relevant when the interplanetary mission fitness requires a high degree of computational resources such as in the case of optimal low-thrust transfers. In that case a surrogate model approximating the final optimal transfer mass enables to quickly search for good launch and arrival epochs, as well as planetary body sequences, e.g. in the case of multiple asteroid or debris rendezvous missions [19, 33]. Unsupervised learning techniques such as clustering or nearest neighbours have also been used to select the target of transfers in multiple asteroid rendezvous missions, upon proper definition of a metric coping with the orbital nonlinearities [27], or to define new box bounds and hence focus successive evolutionary runs in promising areas of the search space (cluster pruning [21]).

A second field where ML algorithms have been used in the context of research in interplanetary trajectory design methods is that of the on-board representation of the optimal guidance profile. Already in 2004 [6] the idea was put forward of using machine learning to learn a representation of the optimal spacecraft guidance profile. In later years the technique was studied exclusively in the context of evolutionary neuro-controllers and optimal control. More recently, neuro-evolution has been substituted with supervised learning (back-propagation), the artificial neural network structure has become deeper, and focus has been shifted to the on-board real time computation of guidance profiles [42–44]. These last works

have all been limited to landing problems and the applicability of these ideas to interplanetary trajectories has not been studied. In the next section we will try to contribute closing this gap showing how the optimal guidance profile of a phase-less Earth–Mars transfer can also be satisfactorily represented by a deep artificial neural network.

4 On-Board Optimal Guidance via a Deep Network

In planetary landing problems, the possibility to generate optimal guidance profiles on-board has been studied and suggested to significantly enhance a vehicles landing accuracy [8, 11]. The resulting algorithms need to run on a radiation-hardened flight processor, that is on a significantly slower processor than those available on modern desktops and one having significant architectural differences. As a rule of thumb, a space qualified processor is an order of magnitude slower than the average desktop processor. The resulting CPU load on the spacecraft has been estimated in a practical scenario [11] to be 0.7s for the computation of one optimal action. To the same end, an alternative is offered by deep feed-forward neural networks trained on the ground to approximate the optimal control and used on-board in real time. The on-board computational effort associated to this architecture is that of one forward pass of the network which, though deep, is not necessarily large [42, 43]. Extending on previous studies on landing and quadcopter dynamics, the feasibility of such a scheme for interplanetary low-thrust transfers is studied here. The same overall scheme is used [43]:

- Step 1: We solve thousands of optimal control problems using Pontryagin's maximum principle. We store, with some time sampling, the obtained solutions in a data set.
- Step 2: We train deep feed-forward neural networks on the data set to learn the optimal control structure.
- Step 3: We use, on-board, the trained network to compute the optimal feedback.

With respect to the previously studied case of planetary landings the methodology is, essentially, unchanged and it is of great interest to see how it can be applied on a radically different problem having larger dimensionality, different nonlinearities and, arguably, a more complex structure.

A Short Note on Notation

We use boldface symbols (e.g. $\boldsymbol{\lambda}$, $\mathbf{r}$, etc.) to denote vector quantities, while scalars or the vector norm will be indicated by normal symbols (e.g. λ, r, etc.). Unit vectors

will be indicated with a small hat (e.g. $\hat{\mathbf{i}}$) and time derivatives with a dot (e.g. $\dot{x}$). We will sometime use the asterisk as a superscript to indicate optimal quantities (e.g. u^*).

4.1 Spacecraft Dynamics

The motion of a spacecraft equipped with a constant specific impulse nuclear electric low-thrust propulsion system can be described in some heliocentric, inertial, frame using Cartesian coordinates by the equations:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\frac{\mu}{r^3}\mathbf{r} + c_1 \frac{u(t)}{m}\hat{\mathbf{i}}_{\mathbf{u}}(t) \\ \dot{m} &= -c_2 u(t) \end{aligned} \tag{1}$$

where $\mathbf{r}$, $\mathbf{v}$ and m denote the spacecraft position, velocity and mass (also denoted as spacecraft state $\mathbf{x}$), μ is the gravitational parameter of the Sun, c_1 is the maximum thrust achievable by the on-board propulsion system and $c_2 = c_1/I_{sp}g_0$. We have denoted with I_{sp} the propulsion specific impulse and with g_0 the Earth gravity constant at sea level. The control variables $u(t)$ and $\hat{\mathbf{i}}_{\mathbf{u}}(t)$ (or $\mathbf{u}(t)$ for brevity) describe the thrust level (here also called throttle) and its direction and are constrained as follows $|u(t)| \leq 1$ and $|\hat{\mathbf{i}}_{\mathbf{u}}(t)| = 1, \forall t \in [t_1, t_2]$.

We are interested in the problem of finding the controls $u(t)$ and $\hat{\mathbf{i}}_{\mathbf{u}}(t)$ to minimise the functional:

$$J(t_1, t_2, u(t)) = \alpha \int_{t_1}^{t_2} u(t)dt + (1-\alpha)\int_{t_1}^{t_2} u^2(t)dt \tag{2}$$

and able to steer the spacecraft from some initial point $x_1 \in \mathscr{S}_1$ to some final point $x_2 \in \mathscr{S}_2$ where the sets $\mathscr{S}_1$ and $\mathscr{S}_2$ are some closed subsets (hypersurfaces) of the state space. Note that the functional J is parameterised by the continuation parameter $\alpha \in [0, 1]$ which weights two contributions corresponding to mass optimality ($\alpha = 1$) and quadratic control optimality ($\alpha = 0$).

We now derive the, known, necessary optimality conditions for the above stated problem applying the maximum principle from Pontryagin (note that we have stated a minimisation problem, hence the conditions are actually slightly different from the ones originally derived in Pontryagin work [38]).

4.2 Minimisation of the Hamiltonian $\mathscr{H}$

We start by introducing seven auxiliary functions defined in $t \in [t_1, t_2]$, the co-states, which we will denote with $\boldsymbol{\lambda}_r$,$\boldsymbol{\lambda}_v$ and λ_m, or, for brevity, $\boldsymbol{\lambda}$. We then introduce the Hamiltonian function:

$$\mathscr{H}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}) = \boldsymbol{\lambda}_r \cdot \mathbf{v} + \boldsymbol{\lambda}_v \cdot \left(-\frac{\mu}{r^3}\mathbf{r} + c_1 \frac{u}{m}\hat{\mathbf{i}}_{\mathbf{u}}\right) - \lambda_m c_2 u + \alpha u + (1-\alpha)u^2 \quad (3)$$

which, following Pontryagin theory, needs to be minimised by our controls along an optimal trajectory. Isolating the relevant part of the Hamiltonian that depends on the control $\hat{\mathbf{i}}_{\mathbf{u}}$ we have that $\mathscr{H} = \ldots + c_1 \frac{u}{m} \boldsymbol{\lambda}_v \cdot \hat{\mathbf{i}}_{\mathbf{u}} + \ldots$, which allows to conclude that, since m, u and c_1 are all positive numbers, the thrust direction must be in the opposite direction of $\boldsymbol{\lambda}_v$ for $\mathscr{H}$ to be minimised, more formally:

$$\hat{\mathbf{i}}^*_{\mathbf{u}} = -\frac{\boldsymbol{\lambda}_v}{\lambda_v} \quad (4)$$

Substituting this expression back into the Hamiltonian we get

$$\mathscr{H}(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}) = \boldsymbol{\lambda}_r \cdot \mathbf{v} - \frac{\mu}{r^3}\boldsymbol{\lambda}_v \cdot \mathbf{r} - c_1 \frac{u}{m}\lambda_v - \lambda_m c_2 u + \alpha u + (1-\alpha)u^2$$

which we now need to minimise with respect to u. Isolating the relevant part of $\mathscr{H}$, i.e. the terms that depend on u, we have

$$\mathscr{H} = \ldots + (1-\alpha)u^2 + u\left(\alpha - \frac{c_1}{m}\lambda_v - \lambda_m c_2\right) + \ldots$$

which is a convex parabola that will take its minimal value in $u = \frac{\frac{c_1}{m}\lambda_v + \lambda_m c_2 - \alpha}{2(1-\alpha)}$. Since $u \in [0, 1]$, we get the final expression for an optimal u^*:

$$u^* = \min\left[\max\left(\frac{\frac{c_1}{m}\lambda_v + \lambda_m c_2 - \alpha}{2(1-\alpha)}, 0\right), 1\right] \quad (5)$$

Note that in the corner case $\alpha = 1$, which corresponds to a mass-optimal control, the above expression results to be singular and the minimiser of the Hamiltonian can be conveniently rewritten introducing the switching function

$$S_{sw} = c_1 \lambda_v + m c_2 \lambda_m - m\alpha$$

as

$$u^* = \begin{cases} 1 \text{ if } & S_{sw} > 0 \\ 0 \text{ if } & S_{sw} < 0 \end{cases} \quad (6)$$

4.3 The Co-state Equations

The states $\mathbf{x}(t)$ and co-states $\boldsymbol{\lambda}(t)$ must be a solution to the set of differential equations that are elegantly written using the Hamiltonian formalism as

$$\begin{aligned}\dot{\mathbf{r}} &= \frac{\partial \mathscr{H}}{\partial \boldsymbol{\lambda}_r}, \;\; \dot{\boldsymbol{\lambda}}_r = -\frac{\partial \mathscr{H}}{\partial \mathbf{r}} \\ \dot{\mathbf{v}} &= \frac{\partial \mathscr{H}}{\partial \boldsymbol{\lambda}_v}, \;\; \dot{\boldsymbol{\lambda}}_v = -\frac{\partial \mathscr{H}}{\partial \mathbf{v}} \\ \dot{m} &= \frac{\partial \mathscr{H}}{\partial \lambda_m}, \; \dot{\lambda}_m = -\frac{\partial \mathscr{H}}{\partial m}\end{aligned}$$

while for the first three equations it is trivial to get back the expressions in Equation (1), the co-states equations do require some extra work, in particular when deriving the gravity gradient. Let us then start deriving the co-state equations by computing $\frac{\partial \mathscr{H}}{\partial \mathbf{r}}$ starting from the expression in Equation (3). We have

$$\frac{\partial \mathscr{H}}{\partial \mathbf{r}} = -\boldsymbol{\lambda}_v \cdot \nabla \left(\frac{\mu}{r^3} \mathbf{r} \right)$$

where the symbol ∇ denotes the gradient operator. It is easier to compute the expression regrouping as follows:

$$\begin{aligned}-\frac{1}{\mu} \frac{\partial \mathscr{H}}{\partial \mathbf{r}} &= \nabla \left(\boldsymbol{\lambda}_v \cdot \frac{1}{r^3} \mathbf{r} \right) = \\ &= \frac{1}{r^3} \nabla \left(\boldsymbol{\lambda}_v \cdot \mathbf{r} \right) + \left(\boldsymbol{\lambda}_v \cdot \mathbf{r} \right) \nabla \left(\frac{1}{r^3} \right) = \\ &= \frac{\boldsymbol{\lambda}_v}{r^3} - 3 \left(\boldsymbol{\lambda}_v \cdot \mathbf{r} \right) \frac{\nabla r}{r^4} = \\ &= \frac{\boldsymbol{\lambda}_v}{r^3} - 3 \left(\boldsymbol{\lambda}_v \cdot \mathbf{r} \right) \frac{\mathbf{r}}{r^5}\end{aligned}$$

We may thus write the corresponding Hamilton equation as

$$\dot{\boldsymbol{\lambda}}_r = -\frac{\partial \mathscr{H}}{\partial \mathbf{r}} = \mu \frac{\boldsymbol{\lambda}_v}{r^3} - 3\mu \left(\boldsymbol{\lambda}_v \cdot \mathbf{r} \right) \frac{\mathbf{r}}{r^5}$$

The remaining two equations are then also easily obtained as

$$\dot{\boldsymbol{\lambda}}_v = -\frac{\partial \mathscr{H}}{\partial \mathbf{v}} = -\boldsymbol{\lambda}_r$$

and

$$\dot{\lambda}_m = -\frac{\partial \mathscr{H}}{\partial m} = c_1 u \frac{\boldsymbol{\lambda}_v \cdot \hat{\mathbf{i}}_{\mathbf{u}}}{m^2}$$

The final system of differential equations describing the state and co-states evolution along an optimal trajectory may be now summarised:

$$\begin{cases} \dot{\mathbf{r}} = \mathbf{v} \\ \dot{\mathbf{v}} = -\frac{\mu}{r^3}\mathbf{r} - c_1 \frac{u^*(t)}{m} \frac{\boldsymbol{\lambda}_v}{\lambda_v} \\ \dot{m} = -c_2 u^*(t) \\ \dot{\boldsymbol{\lambda}}_r = \mu \frac{\boldsymbol{\lambda}_v}{r^3} - 3\mu \left(\boldsymbol{\lambda}_v \cdot \mathbf{r}\right) \frac{\mathbf{r}}{r^5} \\ \dot{\boldsymbol{\lambda}}_v = -\boldsymbol{\lambda}_r \\ \dot{\lambda}_m = -c_1 u^*(t) \frac{\lambda_v}{m^2} \end{cases} \tag{7}$$

4.4 The Two-Point Boundary Value Problem

Following the maximum principle, we know that there exist initial values for the co-states $\boldsymbol{\lambda}(t_1) = \boldsymbol{\lambda}_1$ such that an optimal trajectory steering the system from a starting state $\mathbf{x}_1 \in \mathscr{S}_1$ to a final state $\mathbf{x}_2 \in \mathscr{S}_2$ will be found integrating the system of equations stated in Equation (7) from the initial condition $\mathbf{x}_1, \boldsymbol{\lambda}_1$, for a time $t_2 - t_1$ leading to a final condition $\mathbf{x}_2, \boldsymbol{\lambda}_2$. Since we also want to find optimal values for $\mathbf{x}_1 \in \mathscr{S}_1$, $\mathbf{x}_2 \in \mathscr{S}_2$ and t_2 we need some additional conditions. The problem of finding the optimal t_2 (t_1 can be assumed to be 0 as our system is autonomous) is solved by adding a condition on the Hamiltonian:

$$\mathscr{H}(\mathbf{x}(t_2), \boldsymbol{\lambda}(t_2), \mathbf{u}^*(t_2)) = 0 \tag{8}$$

To optimally select $\mathbf{x}_1 \in \mathscr{S}_1$ and $\mathbf{x}_2 \in \mathscr{S}_2$, instead, we will need to add some conditions, called transversality conditions and derived in the next section.

4.5 Transversality Conditions

Transversality conditions ensure that the initial and final states are selected optimally in the allowed sets $\mathscr{S}_1$ and $\mathscr{S}_2$. They can be elegantly written introducing the vectors $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ belonging to the hyperplane tangent to the hypersurfaces $\mathscr{S}_1$ and $\mathscr{S}_2$ in $\mathbf{x}_1$ and $\mathbf{x}_2$. For all such vectors it must be that

$$\boldsymbol{\lambda}(t_1) \cdot \boldsymbol{\theta}_1 = 0$$

and equivalently,

$$\boldsymbol{\lambda}(t_2) \cdot \boldsymbol{\theta}_2 = 0$$

Let us now derive explicitly these relations in the case of the interplanetary transfer dynamics. We start focusing on the terminal condition $\mathbf{x}_2 \in \mathscr{S}_2$. Typically the final mass of the spacecraft is not specified and we may, in this case, express $\mathscr{S}_2$ as $\mathbf{x}_2 = [\mathbf{r}_2, \mathbf{v}_2, c]$ with c being a free parameter and $\mathbf{r}_2 and \mathbf{v}_2$ being at this stage considered as fixed. Trivially, in this case, $\boldsymbol{\theta}_2 = [\mathbf{0}, \mathbf{0}, \theta_2]$ and the transversality condition for a free final mass is thus

$$\boldsymbol{\lambda}_m(t_2) = 0 \tag{9}$$

More work is needed to derive transversality conditions in the case where also $\mathbf{r}_2$ *and* $\mathbf{v}_2$ are not fixed but constrained to belong to some manifold. A typical situation would be, for example, to leave some of the final Keplerian orbital elements as free. In this case, in order to find $\boldsymbol{\theta}_2$, i.e. the vector tangent to the defined manifold, we need to write a parameterisation of such manifold, that is an expression of $\mathbf{r}_2$ and $\mathbf{v}_2$ as a function of the desired orbital parameter. Using the Keplerian elements $a, e, i, \omega, \Omega, E$ we may use well-known relations to establish such a parameterisation:

$$\begin{aligned} \mathbf{r}_2 &= \mathbf{R} \begin{bmatrix} a(\cos E - e) \\ a\sqrt{1-e^2}\sin E \\ 0 \end{bmatrix} \\ \mathbf{v}_2 &= \sqrt{\tfrac{\mu}{a^3}} \tfrac{1}{1-e\cos E} \mathbf{R} \begin{bmatrix} -a\sin E \\ a\sqrt{1-e^2}\cos E \\ 0 \end{bmatrix} \end{aligned} \tag{10}$$

where $\mathbf{R}$ is the rotation matrix from the orbital reference frame to the inertial defined as

$$\mathbf{R} = \begin{bmatrix} \cos\Omega\cos\omega - \sin\Omega\sin\omega\cos i & -\cos\Omega\sin\omega - \sin\Omega\cos\omega\cos i & \sin\Omega\sin i \\ \sin\Omega\cos\omega + \cos\omega\sin\omega\cos i & -\sin\Omega\sin\omega + \cos\Omega\cos\omega\cos i & -\cos\Omega\sin i \\ \sin\omega\sin i & \cos\omega\cos i & \cos i \end{bmatrix}$$

the tangent vector to the resulting manifold will then be simply obtained deriving the above expression with respect to the chosen parameter.

For example, let us take the case of a free final anomaly. This corresponds to a transfer to some final target orbit regardless of the phase. Deriving the above expressions with respect to E, or in this case equivalently t, we get the transversality condition:

$$\boldsymbol{\lambda}_r \cdot \mathbf{v}_2 - \boldsymbol{\lambda}_v \cdot \frac{\mu}{r_2^3}\mathbf{r}_2 = 0 \tag{11}$$

Similarly if we leave the semi-major axis a as free, the conditions are easily obtained as

$$2\boldsymbol{\lambda}_r \cdot \mathbf{r}_2 - \boldsymbol{\lambda}_v \cdot \mathbf{v}_2 = 0 \tag{12}$$

Similar expressions are derived in [36] using true anomaly and not the eccentric anomaly and using a different method.

5 Test Case Description (Nominal Trajectories)

As a test case, we consider a low-thrust Earth–Mars orbital transfer. A spacecraft having mass $m_0 = 1000$ (kg) and equipped with a propulsion system able to deliver a constant $T_{max} = 0.3$ (N) with a specific impulse of $I_{sp} = 2500$ (s) starts its transfer from Earth's orbit to reach, after a time of flight Δt, Mars' orbit. Both starting and target orbits are considered Keplerian. The optimal transfer strategy is found by solving the resulting TPBVP where we seek the time of flight Δt, departure eccentric anomaly E_0, arrival eccentric anomaly E_f and departure co-state variables λ_0 that satisfy the boundary conditions, the dynamics Equation (7), the transversality condition on free time Equation (8), free mass Equation (9) and free anomalies Equation (12).

The TPBVP is solved first for quadratic control optimality (QOC - $\alpha = 0$). Once the optimal trajectory is found, the problem is then solved with gradually increasing α until mass-optimal control (MOC) is achieved $\alpha = 1$, at which point both solutions (nominal trajectories visualised in Figure 1) are stored as

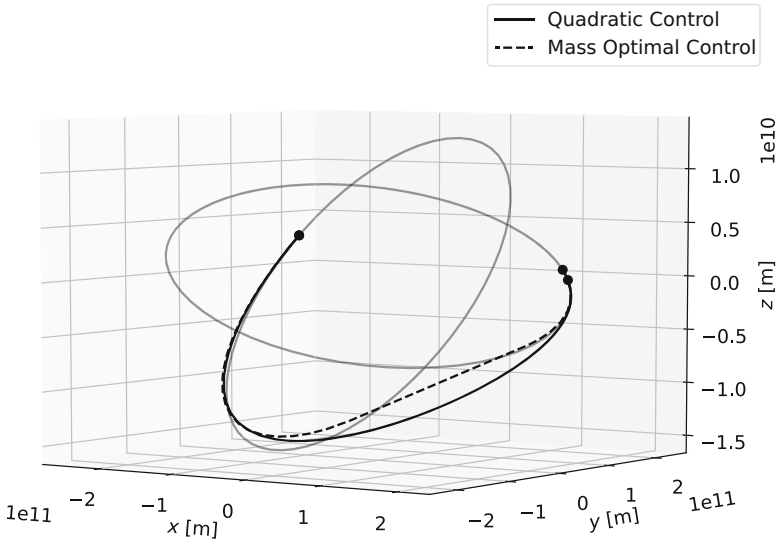

Figure 1 The two nominal trajectories considered

$$\mathbf{T}_{nom} = \left\{ (\mathbf{x}, \boldsymbol{\lambda}, t)_{nom,j} \right\}, j = 0, \ldots, J_{nom}$$

where the grid points are determined by the adaptive integrator used. This will result in more points allocated in areas where the dynamics gradient is higher, which are also areas where we would want to have more training samples, and is thus considered as an appropriate mechanism.

6 Generating the QOC Data Set

As we want to train a deep neural architecture to represent the optimal control around our nominal trajectories, we need to generate a large data set describing the functional relationship that is to be learned. Such a functional relationship is the optimal state feedback ($\mathbf{x} \rightarrow \mathbf{u}^*$) which can be computed along single trajectories neighbouring the nominal ones at the cost of solving the resulting TPBVPs. In order to make use of previously computed solutions (starting from the nominal trajectories) to help the convergence of the TPBVP solver, a continuation technique is then employed where new TPBVPs are created using as initial states those of the nominal trajectories perturbed along a random walk and as initial co-states those computed from previous steps. This allows to generate efficiently the database continuing one starting solution, i.e. the nominal trajectory. The TPBVP is solved by means of a simple single shooting method. The pseudo algorithm used to fill the data set is given in Algorithm 1 which is run starting from ten different points equally spaced along the nominal trajectory and using $\alpha = 0$. Let the final data set

$$\mathscr{T}_{\text{QOC}} = \{\mathbf{T}_i\}, i = 0, \ldots, I \tag{13}$$

contain I trajectories

$$\mathbf{T}_i = \left\{ (\mathbf{x}, \boldsymbol{\lambda}, t)_{i,j} \right\}, j = 0, \ldots, J_i \tag{14}$$

sampled in J_i nodes (as determined by the outcome of an adaptive step numerical integration) each recording the states $\mathbf{x}$, the co-states $\boldsymbol{\lambda}$ and the times t. We refer to this data set as QOC (quadratic optimal control).

6.1 Generating the MOC Data Set

A second data set containing mass-optimal trajectories is obtained via continuation over the parameter α (homotopy). Iterating through the initial states and co-states in $\mathscr{T}_{\text{QOC}}$, an attempt is made to solve, starting from that guess, directly a mass-optimal control ($\alpha = 1$). If the resulting trajectory is feasible, it is stored and the next data set entry is considered. If the trajectory is not feasible, the homotopy

Algorithm 1 Random walk

procedure RANDOM WALK($\mathbf{x}_{nom,j}, \boldsymbol{\lambda}_{nom,j}, \Delta t_{nom,j}, E_{f_{nom}}, \alpha, \overline{\gamma}, n$)
  $\mathscr{T} \leftarrow \{\}$ ▷ Set of optimal trajectories
  $\mathbf{x}_0 \leftarrow \mathbf{x}_{nom,j}$ ▷ Set initial state as nominal
  $\boldsymbol{\lambda}_0 \leftarrow \boldsymbol{\lambda}_{nom,j}$ ▷ Set initial co-state as nominal
  $\Delta t \leftarrow \Delta t_{nom,j}$ ▷ Set time of flight as nominal
  $E_f \leftarrow E_{f_{nom}}$ ▷ Set arrival eccentric anomaly as nominal
  $\gamma \leftarrow \overline{\gamma}$ ▷ Nominal perturbation percentage
  for $i \leftarrow 1, \ldots, n$ **do** ▷ n random perturbations
    $\boldsymbol{\beta} \leftarrow U(-1, 1) \in \mathbb{R}^7$ ▷ Vector of random uniformly distributed numbers
    $\mathbf{x}_1 \leftarrow \mathbf{x}_0 + \mathbf{x}_0 \odot \boldsymbol{\beta}\gamma$ ▷ Perturb state in random direction by γ
    $(\boldsymbol{\lambda}_1, \Delta t_1, E_{f_1}) \leftarrow$ TPBVP($\mathbf{x}_1, \boldsymbol{\lambda}_0, \Delta t, E_f, \alpha$)
    if successful **then** ▷ If the TPBVP is successfully solved
      Update $\mathscr{T}$ ▷ Save successful trajectory
      $\mathbf{x}_0 \leftarrow \mathbf{x}_1$ ▷ Accept perturbed state
      $\boldsymbol{\lambda}_0 \leftarrow \boldsymbol{\lambda}_1$ ▷ Accept new computed co-states
      $\Delta t_0 \leftarrow \Delta t_1$ ▷ Accept new computed time of flight
      $E_{f_0} \leftarrow E_{f_1}$ ▷ Accept new computed eccentric anomaly
      $\gamma \leftarrow (\gamma + \overline{\gamma})/2$ ▷ Increase perturbation size
    else ▷ If solution is unfeasible
      $\gamma \leftarrow \gamma/2$ ▷ Decrease perturbation size and solve again
    end if
  end for
  return $\mathscr{T}$ ▷ Return set of optimal trajectories
end procedure

parameter is decreased, and the optimisation is reattempted. If successful, the homotopy parameter is saved as the current best and its current value is increased. If, instead, the TPBVP solver is not converging, the homotopy parameter is decreased, and the algorithm continues. The pseudo code describing the homotopy method is outlined in Algorithm 2 and results in a new data set $\mathscr{T}_{\text{MOC}}$ containing mass-optimal trajectories and visualised in Figure 2.

7 Learning Details and Network Architecture

We consider for both of the data sets, $\mathscr{T}_{\text{MOC}}$ and $\mathscr{T}_{\text{QOC}}$, the optimal state-action pairs $(\mathbf{x}, \mathbf{u})$ and we build a deep model of the relation $\mathbf{u}(\mathbf{x})$, i.e. the optimal state feedback. The dimension of the values we have to model is $D = 3$ as they represent a three dimensional vector (i.e. the thrust vector). We choose to represent such a vector in its polar coordinates so that

$$\mathbf{u} = u\,[\sin\theta\cos\phi, \sin\theta\sin\pi, \cos\theta]$$

where θ is the polar angle, ϕ is the azimuth angle and u is the throttle magnitude.

Algorithm 2 Homotopy

procedure HOMOTOPY($\mathbf{x}_{QOC,0}, \boldsymbol{\lambda}_{QOC,0}, \Delta t_{QOC}, E_{f_{QOC}}, \alpha_{\text{tol}}$)
  $\alpha \leftarrow 1$ ▷ Try to solve for mass-optimal at first
  loop
    $(\boldsymbol{\lambda}, \Delta t, E_f) \leftarrow$ TPBVP($\mathbf{x}_{QOC,0}, \boldsymbol{\lambda}_{QOC,0}, \Delta t_{QOC}, E_{f_{QOC}}, \alpha$)
    if successful **then** ▷ If the TPBVP is successfully solved
      $\alpha^\star \leftarrow \alpha$ ▷ Current best α
      $(\boldsymbol{\lambda}_{QOC,0}, \Delta t_{QOC}, E_{f_{QOC}}) \leftarrow (\boldsymbol{\lambda}, \Delta t, E_f)$ ▷ $\mathbf{x}_0^\star$ doesn't change
      if $\alpha < \alpha_{\text{tol}}$ **then**
        $\alpha \leftarrow (1 + \alpha)/2$ ▷ Increase α
      else if $\alpha \geq \alpha_{\text{tol}}$ **then** ▷ If α is close to 1
        $\alpha \leftarrow 1$
      else if $\alpha = 1$ **then** ▷ If trajectory is feasible and mass-optimal
        return T ▷ Break the loop and return trajectory
      end if
    else
      $\alpha \leftarrow (\alpha + \alpha^\star)/2$ ▷ Decrease α and retry
    end if
  end loop
end procedure

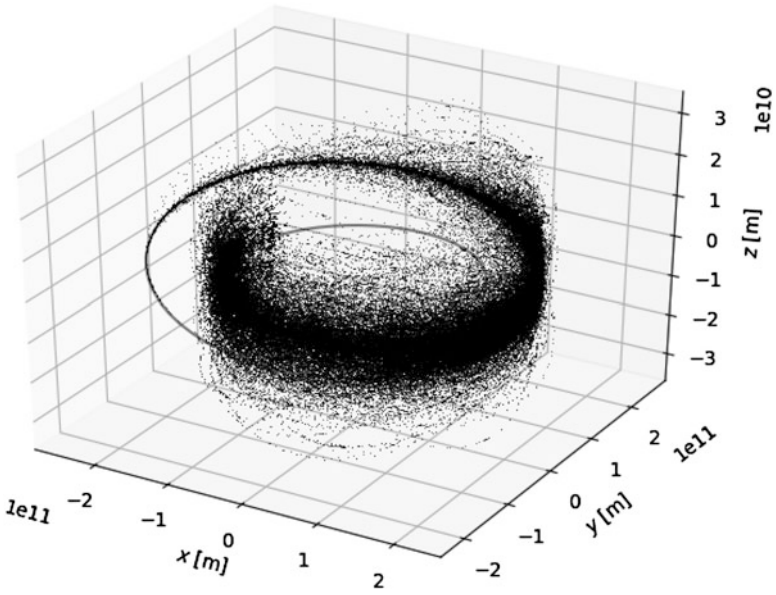

Figure 2 Visualisation of the MOC data set containing 308,579 optimal state-control pairs around the nominal trajectory

In previous work [42], it was found that a feed-forward, fully connected neural network with 3 hidden layers and 32 units/layer could satisfactorily represent the optimal guidance profile for a problem with simpler dynamics. Starting from that knowledge, and from the fact that we are here considering a higher dimensional

case with a higher degree of complexity, we experimented with deeper and wider networks. The result of our investigation indicated that peak performance could be obtained by a neural network with four hidden layers and 200 units per layer. Larger networks either matched this performance or performed worse. We settled on the rectified linear unit (ReLU) as the activation function for the hidden layers. In addition, we used a hyperbolic tangent activation function for the output layer. Thus we scaled the targets in the data set to the range $[-1, 1]$. We initialised the network's weights using the heuristic in [16] in which random values are sampled from a uniform distribution close to zero. Biases were initialised to zero. We also normalised our input data such that features had a mean of zero and standard deviation of one; this helped to speed up the optimisation process. We used the Adam training algorithm [29], a variant of stochastic gradient descent. Weights were updated based on a minibatch of 64 training examples. For the performance metric we use the commonly used mean squared error loss function (MSE). A 10% split is used to define training and validation data. We used an adaptive learning rate starting from an initial value of 10^{-3}. The learning rate was reduced by a factor of 10 when we observed a plateau in the training loss for a period greater than 10 epochs. Furthermore, we stopped training when we observed a plateau in the training loss for greater than 50 epochs ('early stopping'). The plateau was defined to be a decrease in loss less than 10^{-4}. We experimented with different values for the hyperparameters described above before settling on the final configuration.

8 Results

We train different networks to predict the three outputs both concurrently or separately and on both $\mathcal{T}_{\text{MOC}}$ and $\mathcal{T}_{\text{QOC}}$ data sets. We indicate each network using superscripts and subscripts. For example, $\mathcal{N}^{MOC}_{\phi,\theta}$ indicates the network trained to model the variables ϕ and θ in the $\mathcal{T}_{\text{MOC}}$ data set. We report in Table 1 the obtained validation loss in the cases tested. We note that the best results for the output variables are obtained from different models indicating the absence of a clear

Table 1 MSE for neural networks trained on the MOC and QOC data sets

Network	MSE			Epochs
	u	ϕ	θ	
$\mathcal{N}^{QOC}_{u,\phi,\theta}$	0.0178	**0.0690**	**0.0870**	453
$\mathcal{N}^{QOC}_{\phi,\theta}$	–	0.0755	0.0907	461
$\mathcal{N}^{QOC}_{u}$	**0.0075**	–	–	247
$\mathcal{N}^{QOC}_{\phi}$	–	0.0698	–	394
$\mathcal{N}^{QOC}_{\theta}$	–	–	0.0976	503

Network	MSE			Epochs
	u	ϕ	θ	
$\mathcal{N}^{MOC}_{u,\phi,\theta}$	**0.7250**	0.0255	0.0337	794
$\mathcal{N}^{MOC}_{\phi,\theta}$	–	0.0212	**0.0289**	316
$\mathcal{N}^{MOC}_{u}$	0.7740	–	–	721
$\mathcal{N}^{MOC}_{\phi}$	–	**0.0204**	–	248
$\mathcal{N}^{MOC}_{\theta}$	–	–	0.0305	324

The networks differ on the output dimension

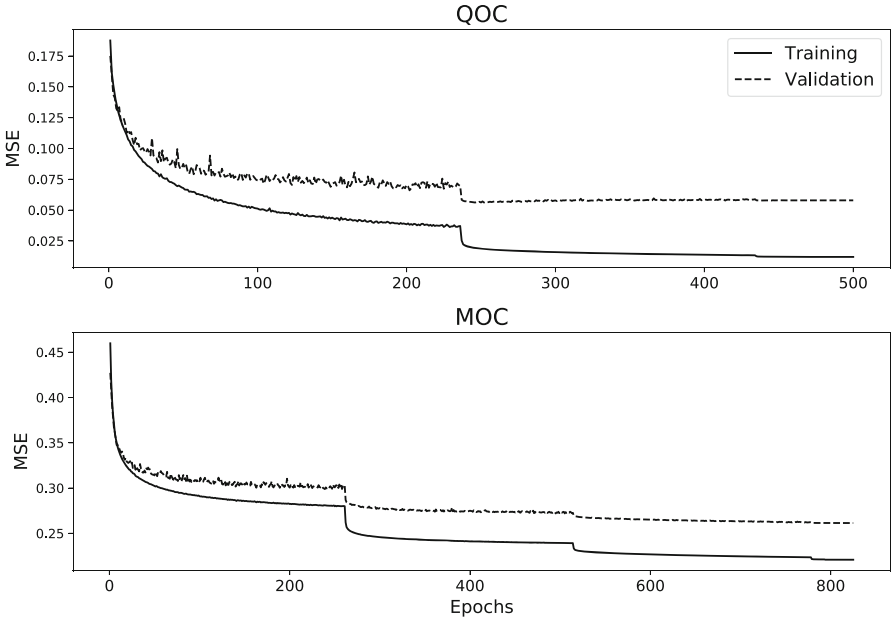

Figure 3 Loss history for the networks $\mathcal{N}_{u,\phi,\theta}^{QOC}$ and $\mathcal{N}_{u,\phi,\theta}^{MOC}$. Note the jumps correspond to the learning rate being reduced

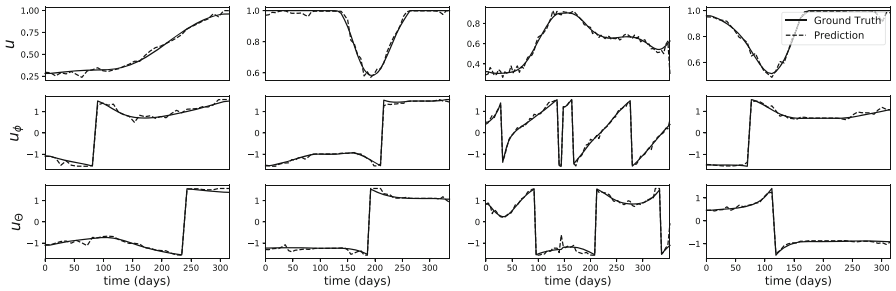

Figure 4 Network $\mathcal{N}_{u,\phi,\theta}^{QOC}$ predictions on four different transfers from the QOC test set

trend to be exploited, while one could expect that predicting the entire thrust vector at once would bring advantages as the network would be able to share some of the weights as part of the necessary computations. We also report in Figure 3 the loss during the training for the representative case of the $\mathcal{N}_{u,\phi,\theta}^{MOC}$ and $\mathcal{N}_{u,\phi,\theta}^{QOC}$ networks showing a relatively standard trend. We finally show four different optimal transfers and plot alongside them the optimal values against the predicted network values. The results are visualised in Figures 4 and 5.

It appears that a deep network is able to represent the optimal guidance structure of the Earth–Mars transfer quite satisfactorily introducing errors that are, on

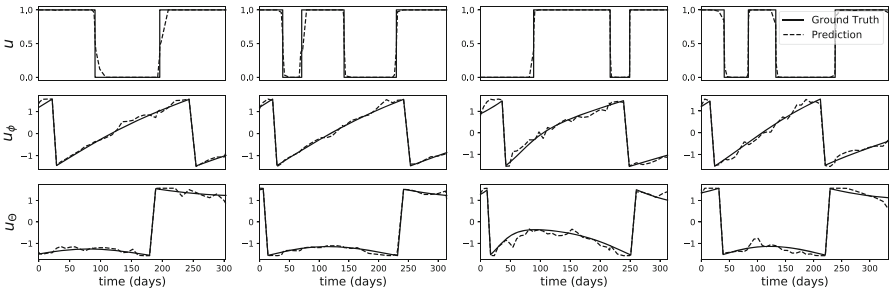

Figure 5 Network $\mathcal{M}_{u,\phi,\theta}^{MOC}$ predictions on four different transfers from the MOC test set

average, rather small. One should also keep in mind that the actual quantitative value of the losses can be improved by further fine-tuning of the hyperparameters used in training the models. In general, we observe the prediction of the throttle u is better attained in the QOC case. This is not surprising as the structure of the optimal control in the MOC case is highly discontinuous being a bang-bang control. We further note that data points close to the switching points are very difficult to predict correctly and when erroneously predicted result in big contributions to the overall loss. These points form a majority in the data sets; a consequence of the time grid defined by the numerical integrator for the TPBVP shooting method solver. This is the reason for the MSE on the u prediction to be of a greater magnitude in the case of the MOC networks, while the visualised plot of the predictions for the same cases is actually returning a much better scenario.

A definitive assessment on the capability of the trained deep models to steer correctly the spacecraft has to be conducted, in a similar fashion to what done in [42], by integrating forward the whole spacecraft dynamics considering the deep network predictions as a control and will be part of a separate future publication.

References

1. Addis, B., Cassioli, A., Locatelli, M., Schoen, F.: A global optimization method for the design of space trajectories. Comput. Optim. Appl. **48**(3), 635–652 (2011)
2. Ampatzis, C., Izzo, D.: Machine learning techniques for approximation of objective functions in trajectory optimisation. In: Proceedings of the IJCAI-09 Workshop on Artificial Intelligence in Space, pp. 1–6 (2009)
3. Biesbroek, R.G., Ancarola, B.P.: Optimization of launcher performance and interplanetary trajectories for pre-assessment studies. In: IAF Abstracts, 34th COSPAR Scientific Assembly (2002)
4. Cassioli, A., Di Lorenzo, D., Locatelli, M., Schoen, F., Sciandrone, M.: Machine learning for global optimization. Comput. Optim. Appl. **51**(1), 279–303 (2012)
5. Ceriotti, M., Vasile, M.: MGA trajectory planning with an ACO-inspired algorithm. Acta Astronaut. **67**(9), 1202–1217 (2010)

6. Dachwald, B.: Low-thrust trajectory optimization and interplanetary mission analysis using evolutionary neurocontrol. Ph.D. thesis, Doctoral thesis, Universität der Bundeswehr München Fakultät für Luft-und Raumfahrttechnik (2004)
7. Dachwald, B., Ohndorf, A.: Global optimization of continuous-thrust trajectories using evolutionary neurocontrol. In: Fasano, G., Pinter, J. (eds.) Modeling and Optimization in Space Engineering - 2018. Springer, Basel (2019)
8. de Croon, G., Izzo, D.: Real-time landing based on optimality principles and vision. In: 23rd International Symposium on Space Flight Dynamics (ISSFD) (2012)
9. Deb, K., Padhye, N., Neema, G.: Interplanetary trajectory optimization with swing-bys using evolutionary multi-objective optimization. In: International Symposium on Intelligence Computation and Applications, pp. 26–35. Springer, Berlin (2007)
10. Di Lizia, P., Radice, G.: Advanced global optimisation for mission analysis and design. Final Report Ariadna id 04/4101 (2004)
11. Dueri, D., Açıkmeşe, B., Scharf, D.P., Harris, M.W.: Customized real-time interior-point methods for onboard powered-descent guidance. J. Guid. Control. Dyn. **40**, 197–212 (2016)
12. Elsayed, S.M., Sarker, R.A., Essam, D.L.: GA with a new multi-parent crossover for solving IEEE-CEC2011 competition problems. In: IEEE Congress on Evolutionary Computation (CEC), 2011, pp. 1034–1040. IEEE, Piscataway (2011)
13. Englander, J.: Automated trajectory planning for multiple-flyby interplanetary missions. University of Illinois at Urbana-Champaign (2013)
14. Gad, A., Abdelkhalik, O.: Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization. J. Spacecr. Rocket. **48**(4), 629–641 (2011)
15. Gage, P., Braun, R., Kroo, I.: Interplanetary trajectory optimization using a genetic algorithm. J. Astronaut. Sci. **43**(1), 59–76 (1995)
16. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010)
17. Grigoriev, I., Zapletin, M.: Choosing promising sequences of asteroids. Autom. Remote. Control. **74**(8), 1284–1296 (2013)
18. Hennes, D., Izzo, D.: Interplanetary trajectory planning with Monte Carlo tree search. In: IJCAI, pp. 769–775 (2015)
19. Hennes, D., Izzo, D., Landau, D.: Fast approximators for optimal low-thrust hops between main belt asteroids. In: IEEE Symposium Series on Computational Intelligence (SSCI), 2016, pp. 1–7. IEEE, Piscataway (2016)
20. Islam, S.M., Das, S., Ghosh, S., Roy, S., Suganthan, P.N.: An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. IEEE Trans. Syst. Man Cybern. B Cybern. **42**(2), 482–500 (2012)
21. Izzo, D.: Global optimization and space pruning for spacecraft trajectory design. Spacecr. Trajectory Optim. **1**, 178–200 (2010)
22. Izzo, D., Becerra, V.M., Myatt, D.R., Nasuto, S.J., Bishop, J.M.: Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories. J. Glob. Optim. **38**(2), 283–296 (2007)
23. Izzo, D., Simões, L.F., Märtens, M., De Croon, G.C., Heritier, A., Yam, C.H.: Search for a grand tour of the Jupiter Galilean moons. In: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, pp. 1301–1308. ACM, New York (2013)
24. Izzo, D., Hennes, D., Riccardi, A.: Constraint handling and multi-objective methods for the evolution of interplanetary trajectories. J. Guid. Control. Dyn. **38**, 792–800 (2014)
25. Izzo, D., Simoes, L.F., Yam, C.H., Biscani, F., Di Lorenzo, D., Addis, B., Cassioli, A.: GTOC5: results from the European Space Agency and University of Florence. Acta Futura **8**, 45–55 (2014)
26. Izzo, D., Getzner, I., Hennes, D., Simões, L.F.: Evolving solutions to TSP variants for active space debris removal. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 1207–1214. ACM, New York (2015)

27. Izzo, D., Hennes, D., Simões, L.F., Märtens, M.: Designing complex interplanetary trajectories for the global trajectory optimization competitions. In: Space Engineering, pp. 151–176. Springer, Berlin (2016)
28. Janin, G., Gomez-Tierno, M.: The genetic algorithms for trajectory optimization. In: Stockholm International Astronautical Federation Congress (1985)
29. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014). CoRR abs/1412.6980. http://arxiv.org/abs/1412.6980, 1412.6980
30. Lavagna, M.R.: Multi-objective pso for interplanetary trajectory design. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 175–175. ACM, New York (2007)
31. Lee, S., von Allmen, P., Fink, W., Petropoulos, A., Terrile, R.: Multi-objective evolutionary algorithms for low-thrust orbit transfer optimization. In: Genetic and Evolutionary Computation Conference (GECCO 2005) (2005)
32. Luo, Y.Z., Tang, G.J., Zhou, L.N.: Simulated annealing for solving near-optimal low-thrust orbit transfer. Eng. Optim. **37**(2), 201–216 (2005)
33. Mereta, A., Izzo, D., Wittig, A.: Machine learning of optimal low-thrust transfers between near-earth objects. In: International Conference on Hybrid Artificial Intelligence Systems, pp. 543–553. Springer, Berlin (2017)
34. Myatt, D., Becerra, V.M., Nasuto, S.J., Bishop, J.: Advanced global optimisation for mission analysis and design. Final Report Ariadna id 04/4101 (2004)
35. Olds, A.D., Kluever, C.A., Cupples, M.L.: Interplanetary mission design using differential evolution. J. Spacecr. Rocket. **44**(5), 1060–1070 (2007)
36. Pan, B., Chen, Z., Lu, P., Gao, B.: Reduced transversality conditions in optimal space trajectories. J. Guid. Control. Dyn. **36**, 1289–1300 (2013)
37. Pontani, M., Conway, B.A.: Particle swarm optimization applied to space trajectories. J. Guid. Control. Dyn. **33**(5), 1429–1441 (2010)
38. Pontryagin, L.S., Boltyanskii, V., Gamkrelidze, R., Mishchenko, E.F.: The Mathematical Theory of Optimal Processes. Interscience, New York (1962)
39. Radice, G., Olmo, G.: Ant colony algorithms for two-impulse interplanetary trajectory optimization. J. Guid. Control. Dyn. **29**(6), 1440 (2006)
40. Rauwolf, G.A., Coverstone-Carroll, V.L.: Near-optimal low-thrust orbit transfers generated by a genetic algorithm. J. Spacecr. Rocket. **33**(6), 859–862 (1996)
41. Rogata, P., Di Sotto, E., Graziano, M., Graziani, F.: Guess value for interplanetary transfer design through genetic algorithms. Adv. Astronaut. Sci. **114**, 613–627 (2003)
42. Sánchez-Sánchez, C., Izzo, D.: Real-time optimal control via deep neural networks: study on landing problems (2016). arXiv preprint arXiv:161008668
43. Sánchez-Sánchez, C., Izzo, D., Hennes, D.: Learning the optimal state-feedback using deep networks. In: IEEE Symposium Series on Computational Intelligence (SSCI), 2016, pp. 1–8. IEEE, Piscataway (2016)
44. Schiavone, G., Izzo, D., Simões, L.F., De Croon, G.C.: Autonomous spacecraft landing through human pre-attentive vision. Bioinspir. Biomim. **7**(2), 025,007 (2012)
45. Schlueter, M.: MIDACO software performance on interplanetary trajectory benchmarks. Adv. Space Res. **54**(4), 744–754 (2014)
46. Schlueter, M., Erb, S.O., Gerdts, M., Kemble, S., Rückmann, J.J.: MIDACO on MINLP space applications. Adv. Space Res. **51**(7), 1116–1131 (2013)
47. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015)
48. Sentinella, M.R., Casalino, L.: Hybrid evolutionary algorithm for the optimization of interplanetary trajectories. J. Spacecr. Rocket. **46**(2), 365 (2009)
49. Simões, L.F., Izzo, D., Haasdijk, E., Eiben, A.E.: Self-adaptive genotype-phenotype maps: neural networks as a meta-representation. In: International Conference on Parallel Problem Solving from Nature, pp. 110–119. Springer (2014)
50. Simões, L.F., Izzo, D., Haasdijk, E., Eiben, A.: Multi-rendezvous spacecraft trajectory optimization with beam P-ACO. In: European Conference on Evolutionary Computation in Combinatorial Optimization, pp. 141–156. Springer, Berlin (2017)

51. Stracquadanio, G., La Ferla, A., De Felice, M., Nicosia, G.: Design of robust space trajectories. In: SGAI Conference, pp. 341–354 Springer, Berlin (2011)
52. Vasile, M., Minisci, E., Locatelli, M.: Analysis of some global optimization algorithms for space trajectory design. J. Spacecr. Rocket. **47**(2), 334 (2010)
53. Vinkó, T., Izzo, D.: Global optimisation heuristics and test problems for preliminary spacecraft trajectory design. Eur Space Agency, Adv Concepts Team, ACT Tech Rep, id: GOHTPPSTD (2008)
54. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)
55. Yao, W., Luo, J., Macdonald, M., Wang, M., Ma, W.: Improved differential evolution algorithm and its applications to orbit design. J. Guid. Control. Dyn. **41**, 1–8 (2017)