Astrodynamics Vol. 9, No. 2, 195–215, 2025 https://doi.org/10.1007/s42064-024-0216-6

Transcription and optimization of an interplanetary trajectory through quantum annealing

Federico De Grossi[1], Andrea Carbone[1](✉), Dario Spiller[2], Daniele Ottaviani[3], Riccardo Mengoni[3], and Christian Circi[1]

1. Sapienza University of Rome, Rome 00138, Italy
2. School of Aerospace Engineering, Sapienza University of Rome, Rome 00138, Italy
3. CINECA, Casalecchio di Reno, Bologna 40033, Italy

ABSTRACT

This study employed a quantum-annealing framework to solve spacecraft trajectory optimization problems. Quantum annealing belongs to the field of quantum computing and is a promising technique for tackling hard binary optimization problems by employing quantum annealers. To address the optimal control of a trajectory using quantum annealing, a transcription procedure was introduced to express the problem in the binary optimization form required. The proposed procedure leverages the pseudospectral method to discretize the trajectory and represents the dynamical constraints as algebraic equality constraints at specific nodes. Subsequently, both a linearization procedure and binary representation strategy for the real-valued variables of the problem were presented, leading to the quadratic binary unconstrained optimization form. The quantum-annealing-based method was tested in the context of an interplanetary low-thrust transfer from the Earth to Mars. First, we discussed which instances of the problem, especially in terms of their dimensions, are implementable on currently available quantum annealers; then, a solution was sought by employing annealers from D-Wave systems. Solutions from *hybrid* solvers that combine classical and quantum resources, and fully quantum solvers were explored. The results demonstrate the validity of the transcription approach, demonstrate the ability of the hybrid solver to tackle the case-study problem, and highlight the promising features and current limitations of practical trajectory optimization with quantum annealing.

KEYWORDS

trajectory optimization
pseudospectral method
quantum computing (QC)
quantum annealing (QA)
quantum-classical hybrid computing

Research Article

Received: 30 October 2023
Accepted: 19 April 2024

1 Introduction

The trajectory of a space mission directly affects its feasibility and design, and it influences various aspects of the mission-planning process. The design of an optimal trajectory is essential to ensure that mission objectives can be met within the given constraints. By carefully analyzing and optimizing the trajectory, space missions can be made more efficient, cost-effective, and feasible. Therefore, trajectory optimization is a crucial aspect of space mission planning and holds significant importance in the field of aerospace engineering. Active research has been conducted to discover new approaches, methods, and algorithms for trajectory optimization.

In the space engineering community, considerable attention has been devoted to methods for transcribing the trajectory design problem into an optimization problem; hence, traditionally, optimization methods have been divided into indirect and direct methods [1, 2]. Indirect methods rely on the theoretical results from the calculus of variations to derive the necessary conditions for optimality [3]. These methods involve expanding the system with costate equations that evolve over time and transform the problem into a two-point

✉ and.carbone@uniroma1.it

boundary-value problem. While indirect methods provide guaranteed extremal solutions, they have drawbacks such as sensitivity to initial estimates and convergence challenges. Conversely, direct methods discretize the problem, thereby allowing a finite set of parameters to be optimized. This transforms the problem into a nonlinear programming (NLP) problem that is solvable using optimization algorithms. Occasionally, metaheuristic methods are considered a separate category, although they are more closely related to the algorithms used to solve optimization problems. Metaheuristics have also achieved noteworthy results in the field of spacecraft trajectory optimization, providing good-quality solutions to complex problems in a reasonable time [4]. In the metaheuristic optimization field and partly related to the quantum computing field, although still classical, quantum-inspired algorithms have also been found useful [5, 6].

Over the past two decades, significant advancements have been made in trajectory optimization methods owing to increasing computational power and the emergence of new tools that facilitate implementation. Notably, the utilization of neural networks has opened new possibilities and has been applied in various scenarios. For example, Federici *et al.* [7] employed reinforcement learning techniques to autonomously guide a spacecraft during a mission targeting a binary asteroid. This approach utilized neural networks for processing optical observations and demonstrated a notable success rate in reaching the mission target despite uncertainties. Similarly, Scorsoglio *et al.* [8] applied reinforcement meta-learning techniques combined with hazard detection and avoidance strategies. Their system relies on both image and radar data to identify potential hazards and select secure landing locations. Jiang *et al.* [9] addressed the challenges associated with exploring asteroid surfaces characterized by a complex terrain and irregular gravity. Their proposed solution deployed hopping rovers and explored deep reinforcement learning methods for three-dimensional path planning. Gaudet *et al.* [10] optimized an asteroid-hovering controller using reinforcement meta-learning. This approach allows stable positioning even when precise asteroid data are not readily available. The integration of reinforcement learning and meta-learning methods has demonstrated potential advantages for achieving optimal outcomes, as discussed in Refs. [11, 12]. The proposed frameworks were built upon a combination of particle swarm optimization for trajectory planning and an extreme learning machine for accurately estimating asteroid gravitational acceleration. These advancements have paved the way for more sophisticated and efficient trajectory optimization techniques.

Quantum computing (QC) is a fundamentally different model of computation based on the principles of quantum mechanics, which are known to solve some specific mathematical problems more quickly (up to exponential factors) compared to the best-known classical algorithms. Just as the fundamental unit of information in a classical computer is the *bit*, an object that can assume two values, in a quantum computer, the fundamental unit is the *qubit*, which is a quantum system with only two possible states. Because qubits can exist in superposition and entangled states, operations that are impossible for classical computers can be performed. Owing to these characteristics, quantum computers can process information in parallel and explore multiple solutions simultaneously (quantum parallelism) [13]. Some problems for which QC provides an advantage over classical computers include the factorization of prime numbers [14], unstructured database search [15], solution of linear systems of equations [16], and simulation of quantum systems [17–19]. Research on QC is ongoing, and many algorithms have been proposed in other areas, such as differential equations [20], artificial intelligence and machine learning [21, 22], and optimization [23, 24].

While the above-cited literature mainly concerns *gate-based* QC, that is, quantum computers, where the qubits are manipulated by applying a series of gates, similar to classical computers, another important model of QC is the quantum adiabatic computing (QAC) framework. QAC leverages and drives the natural evolution of a quantum system to obtain the results for a problem [25]. Quantum annealing (QA) is an algorithm in the QAC framework that is particularly well suited for solving combinatorial optimization problems. During computation, the system gradually transitions from an initial state to a low-energy state, resembling the cooling process in metallurgy, hence the term “annealing”. QA offers more efficient optimization solutions than classical optimization algorithms, particularly for large-scale and complex problems [26].

Although QAC and QA are formulated as optimization problem solvers, they can be generalized and applied to various mathematical problems. In fact, they have

been proposed for several of the same problems as gate-based QC, allowing for even faster computations because quantum annealers can perform computations with a number of qubits that is approximately one order of magnitude higher than that of gate-based machines. The proposed applications of QA include the factorization of prime numbers [27], polynomial systems of equations [28], and differential equations [29, 30].

Applying QA to practical engineering problems is also being researched [31]; however, there are limited resources available for current quantum annealers. In the field of aerospace engineering, a few applications of QA have been proposed such as casting some classes of artificial intelligence problems as combinatorial [32], aircraft trajectory deconfliction [33], and agile Earth observation scheduling for the maximum number of image acquisitions [34].

Although QA has gained significant attention in the optimization community, its potential in space trajectory optimization remains unexplored. Thus, in this study, a transcription procedure is proposed to transform a trajectory optimization problem into a binary optimization problem, specifically a quadratic unconstrained binary optimization (QUBO) form suitable for QA computation. The procedure can be applied to a wide range of optimal control problems that involve satisfying dynamic, terminal, and path boundary constraints. Dynamic constraints capture the system dynamic behavior and ensure that the state and control inputs obey the relevant equations of motion. The terminal constraints define the initial and final states that the system must attain. The boundary constraints are active throughout the trajectory and can express limits on the state and control variables. To apply QA, the pseudospectral method is employed to discretize the state and control variables as a sum of the interpolated Lagrange polynomials over the time interval [35]. The pseudospectral method is an established direct optimization technique suitable for low-thrust trajectory design [36, 37]. Thus, the continuous-time optimal control problem is transformed into a finite-dimensional NLP problem. The pseudospectral method provides a compact and simple formulation of dynamic constraints, which is leveraged to transcribe the optimal control problem into a quadratic problem with linearized constraints. Subsequently, the binary representation of the real-valued variables is transformed into the QUBO form. The resulting QUBO problem is solved using QA to determine the optimal binary variables that minimize the cost function while satisfying the imposed constraints. A sequential optimization approach is employed in which the solution obtained from one iteration is used as the reference trajectory for the next iteration, gradually improving the optimization process. The proposed transcription procedure is tested in a low-thrust Earth–Mars transfer—a relatively simple problem chosen because it is commonly used to test new methods and techniques, and can be simplified and implemented on existing quantum annealers.

The remainder of this paper is structured as follows. Section 2 provides a summary description of QA. Section 3 presents an overview of the proposed transcription procedure and explains the mathematical formulation of the problem, including its cost function and relevant constraints. Section 4 presents the low-thrust Earth–Mars transfer problem with the relevant assumptions used to implement it on a quantum annealer. Section 5 discusses the results of the transcription procedure and optimization, where both full-quantum and hybrid solvers are explored. Finally, Section 6 concludes the paper and discusses potential future research directions in the field of QA for trajectory optimization.

2 Quantum annealing optimization

To present a concept based on QA, we introduce the Hamiltonian of a quantum system and its eigenstates. The Hamiltonian is an operator—an object that in quantum mechanics represents a physical quantity—in this case, *energy*. Every operator gives rise to the so-called eigenvalue problem in the form

$$H\left|\psi_n\right\rangle = E_n\left|\psi_n\right\rangle \tag{1}$$

where $|\psi_n\rangle$ is the state of the system satisfying the equation (the Dirac "ket" notation is used, expressing a quantum state in the brackets $|\cdot\rangle$), H is the Hamiltonian, and E_n is the n-th eigenvalue. The eigenvalue E_n is a physical interpretation of one of the possible values of energy that the system can have after a measurement, and $|\psi_n\rangle$ is the state of the system that corresponds to that energy, called an eigenvector or eigensate. Among all the possible energy values, one that is lower than all the others is called the *ground state*, and it is the minimum possible energy value of the system.

The system evolution in time is given by the Schrödinger equation, where the Hamiltonian has a central role. If the system is at the start of evolution in one of the energy eigenstates, it remains in that state every subsequent time, provided that the Hamiltonian is constant in time. The Schrödinger equation is

$$\frac{\partial |\Psi\rangle}{\partial t} = -\frac{\mathrm{i}}{\hbar} H |\Psi\rangle \tag{2}$$

where $|\Psi\rangle$ is the system state, i is an imaginary unit, $\hbar = h/2\pi$, and h is Planck's constant.

When H varies over time, the situation becomes more complicated. However, the quantum adiabatic theorem provides insight into the time evolution of a *slowly* varying system.

The quantum adiabatic theorem states that if a system is in the nth eigenstate of the Hamiltonian $H(0)$ at time $t = 0$, and if the Hamiltonian changes sufficiently slowly in time, then the system will remain in the nth eigenstate of the *instantaneous* Hamiltonian $H(t)$ [32]. Therefore, for each t, the system state is a solution to the eigenvalue equation in Eq. (3):

$$H(t) |\psi_n(t)\rangle = E_n(t) |\psi_n(t)\rangle \tag{3}$$

This implies that if the system is prepared in an eigenstate (for example, the ground state) at the end of the evolution, it will be in the corresponding eigenstate of the final Hamiltonian.

From the adiabatic evolution, it is easy to infer that the QAC framework, of which QA belongs, naturally accommodates optimization problems. If the cost function corresponds to the energy of a Hamiltonian, then it is in the ground state, where the minimum energy value is the global optimum of the cost function.

A quantum annealer is composed of a network of qubits, and for such a quantum system, the Hamiltonian can be written as a matrix with dimensions $2^n \times 2^n$ if n is the number of qubits. The optimization procedure starts by preparing the systems in the ground state of a simple Hamiltonian, called the *Hamiltonian driver* (H_D). The optimization problem is encoded in the energy eigenvalues of a *Hamiltonian problem* H_P. Then, the system is evolved adiabatically from H_D to H_P, so that the system state at the end of the evolution is the ground state of the Hamiltonian problem. Finally, to obtain the solution, the qubits of the register are measured. Eq. (4) shows a linear interpolation between H_D and H_P, and Eq. (5) shows the eigenvalues equation at the start and end of the annealing process. In general, the interpolation between H_D and H_P may also be nonlinear with the general functions $A(t/T)$ and $B(t/T)$ such that $A(0) = 1$, $B(0) = 0$, and $A(1) = 0$, $B(1) = 1$.

$$H(t) = \left(1 - \frac{t}{T}\right) H_\mathrm{D} + \frac{t}{T} H_\mathrm{P} \tag{4}$$

$$H_\mathrm{D} |\psi_{0,\mathrm{D}}\rangle = E_{0,\mathrm{D}} |\psi_{0,\mathrm{D}}\rangle \xrightarrow{t=[0,T]} H_\mathrm{P} |\psi_{0,\mathrm{P}}\rangle = E_{0,\mathrm{P}} |\psi_{0,\mathrm{P}}\rangle \tag{5}$$

The driver Hamiltonian is related to the fully symmetric state of the quantum register $|\psi_\mathrm{S}\rangle$, where all qubits in superposition and all qubit configurations are equally probable and have the same energy, as Eq. (6):

$$|\psi_{0,\mathrm{D}}\rangle = |\psi_\mathrm{S}\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle \tag{6}$$

The velocity of the adiabatic evolution, that is, how slowly the Hamiltonian must change, is related to the *minimum energy gap*, which is the minimum difference between the evolution of the ground state energy and the first excited state. When the gap is small and the evolution is too fast, the system may jump from the ground state to the upper state, and the computation results in a suboptimal value. This condition can be expressed by

$$|\langle\psi_0(t)| \dot{H}(t) |\psi_1(t)\rangle| \ll \frac{|E_1(t) - E_0(t)|}{\hbar} \tag{7}$$

Therefore, by calling g_min the minimum energy gap, the annealing time is of the order of $T = O(1/g_\mathrm{min}^2)$.

Figure 1 illustrates the above physical concepts, where the energy levels of the two Hamiltonians are schematically represented. Given the capability of transcribing the cost function values of the optimization problem into the energy values of H_P and that the annealing process can start at the minimum energy level of the Hamiltonian H_D, adiabatic evolution allows the path between the two problems to remain at the minimum energy level. In Fig. 1, this transition is represented by the blue line connecting the two ground states, while the bar below shows the initial state (Eq. (7)) and final solution for optimization. In the figure, the minimum energy gap dictates the time required for QA to be considered adiabatic. In fact, at time t_c during evolution, the ground state and first excited states are the closest.

The optimization problem that can be solved by QA has a specific structure and quantum annealers have been built to implement this structure. The problem has two equivalent formulations: the Ising model inherited

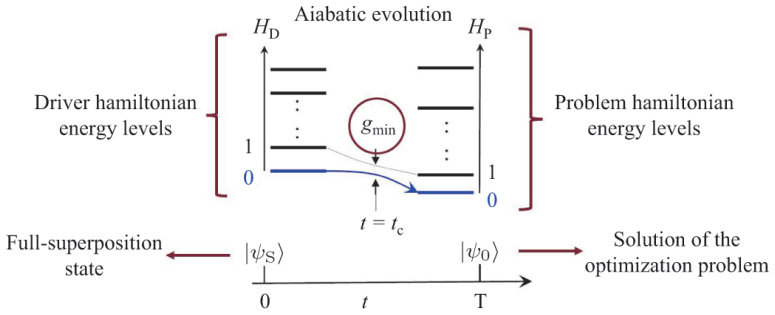


Fig. 1 Adiabatic evolution and annealing process.

from the physics of spin glasses and QUBO from the combinatorial optimization field [38].

- *The Ising model* is commonly used in physics and statistical mechanics to represent certain physical systems. In this model, the minimum energy configuration of a system of interacting spins must be found. Let us consider an Ising model with n spins, denoted as $\boldsymbol{s} = (s_1, s_2, \cdots, s_n)$, where each spin can be in the states $+1$ (up) or -1 (down). The energy of the Ising model is given by

$$E_{\mathrm{I}}(\boldsymbol{s}) = \sum_i h_i s_i + \sum_{i<j,(i,j)\in\mathcal{E}} b_{ij} s_i s_j \tag{8}$$

  where h_i and b_{ij} are the linear coupling coefficients for the individual and pairs of spins, respectively. The first sum takes into account the energy associated with the individual spin states, while the second sum represents the energy due to interactions between spins. The set of interacting spins is $\mathcal{E}$.
- *QUBO* uses a quadratic function of binary variables. Consider a problem with n binary variables, denoted as $\boldsymbol{z} = (z_1, z_2, \cdots, z_n)$, where each z_i value can be 0 or 1. The objective is to minimize the quadratic function in Eq. (9):

$$E_Q(\boldsymbol{z}) = \boldsymbol{z}^{\mathrm{T}} Q \boldsymbol{z} = \sum_{i=1}^{n}\sum_{j=1}^{n} Q_{ij} z_i z_j \tag{9}$$

  where $\boldsymbol{z}^{\mathrm{T}}$ is the transpose of the binary variable vector $\boldsymbol{z}$ and Q is an $n \times n$ matrix of coefficients. The goal is to find the binary values of z_i that minimize the quadratic function $E_Q(\boldsymbol{z})$.

Eqs. (8), and (9) are strictly related through the transformation $z_i = (s_i + 1)/2$, which maps the Ising spins (s_i) to binary variables (z_i). In particular, in the Ising model, the linear coupling coefficients (h_i) are transformed into diagonal components of the matrix Q in the QUBO, whereas the coupling coefficients for spin pairs (b_{ij}) are represented by off-diagonal components.

The Ising model and QUBO provide powerful mathematical frameworks for representing and solving optimization problems. However, they are mathematical abstractions that facilitate the expression of optimization problems in a convenient and general format, and to implement these formulations in physical quantum computers, additional challenges stemming from the physical constraints imposed by the underlying qubit lattice must be addressed. These challenges include the physical qubits, which are arranged in a specific connectivity pattern or topology to form a lattice structure, and current annealers, which have limited connectivity between qubits; therefore, an arbitrary QUBO or Ising problem does not match the topology of a specific machine. To overcome these challenges, an *embedding* technique was employed, which was adapted for the working graph of the annealer by forming chains of qubits to create the required connections. Thus, the working graph includes a logical graph that includes the connection of the QUBO/Ising problem to be solved and a physical graph that represents the actual connectivity of the annealer. If a logical variable in a problem requires interactions between qubits that are not directly connected in a physical lattice, an embedding technique may create a chain by connecting multiple qubits in series. Figure 2 shows a logical graph with three interacting variables (represented as circles), where variable 1 interacts with variables 3 and 4. However, the physical graph on the right does not provide a direct connection between qubits 1 and 4. Therefore, an embedding can create a chain including qubits 2 and 4 such that connections 1–4 can be implemented by exploiting connections 1–2 and 2–4, with qubit 2 acting as a bridge. Although the logical problem includes three variables, the embedded problem must use four qubits,

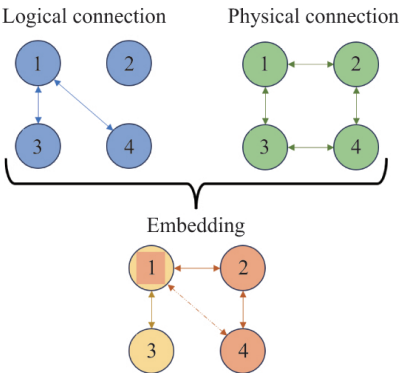


Fig. 2 Simple embedding procedure. Embedding connects qubits 1 and 4 by creating a chain with qubit 2.

and the embedding procedure increases the number of qubits required to implement the problem. Consequently, a problem that requires embedding could require an annealer with a number of qubits that is much larger than the number of variables.

Thus, the embedding procedure in QA solves the generic QUBO problem even if it does not match the annealer topology, but it uses more qubits than the logical binary variables of the problem. Moreover, if the chains are long, there may be an increased susceptibility to noise and error. This susceptibility arises because some chains may break, creating a discrepancy between the logical problem and annealer.

The D-Wave system is a commercialized quantum annealer [39]. Several options are available to users who wish to employ D-Wave quantum annealers to tackle optimization problems with the different available *samplers*. Full-quantum solvers, called quantum processing units, tackle a user-provided QUBO problem directly on a quantum annealer, but embedding is required and these solvers have size limitations. Quantum-classical hybrid solvers combine classical algorithms with quantum processing, and are better suited for more complex optimization problems because the size limitations are much larger. These solvers employ classical algorithms to preprocess the problem before leveraging its quantum-processing capabilities to complete the optimization process.

3 Transcription procedure

To implement a trajectory optimization problem with a quantum annealer, a suitable transcription must be found such that the optimal control problem can be transformed into an equivalent binary optimization problem in the QUBO form; therefore, a transcription procedure is proposed in this section. This procedure can be applied to a wide range of optimal-control problems. Its limitations, as discussed, are due to the necessity of obtaining a QUBO structure.

In general, trajectory optimization, or optimal control, is a fundamental problem in control theory that deals with finding the optimal control inputs that guide a system from an initial state to a desired final state while minimizing a specified cost function and satisfying a set of constraints of various types. In trajectory optimal control, the cost function is typically expressed in the Bolza form as Eq. (10):

$$J = \chi(\boldsymbol{x}_0, \boldsymbol{x}_\mathrm{f}, t_0, t_\mathrm{f}) + \int_{t_0}^{t_\mathrm{f}} \mathcal{L}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\mathrm{d}t \tag{10}$$

where $\boldsymbol{x}(t)$ is the system state at time t, $\boldsymbol{u}(t)$ denotes the control, t_0 is the initial time, $\boldsymbol{x}_0$ is the initial state, t_f is the final time, and $\boldsymbol{x}_\mathrm{f}$ is the final state. The optimal trajectory control problem involves satisfying a set of constraints that can include dynamic, terminal, and path constraints. Dynamic constraints capture the dynamic behavior of the system and ensure that the state and control input of the system obey the relevant equations of motion. Therefore, these constraints have a differential form as Eq. (11):

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \tag{11}$$

where $\boldsymbol{f}$ is the system dynamics. Terminal constraints express a relationship that depends on the state and must be verified at the initial and final times. In this study, to impose the initial and final states that the system must attain, they are expressed as

$$\begin{cases} \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \\ \boldsymbol{x}(t_\mathrm{f}) = \boldsymbol{x}_\mathrm{f} \end{cases} \tag{12}$$

Path constraints are active throughout the entire trajectory. In general, they can be defined as inequalities and are typically expressed as

$$\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \leqslant \boldsymbol{0} \tag{13}$$

The path constraints express the common boundary constraints on the state and control, which express the state and control variables as belonging to a determined limited space, as Eq. (14):

$$\begin{cases} \boldsymbol{x}_\mathrm{min} \leqslant \boldsymbol{x} \leqslant \boldsymbol{x}_\mathrm{max} \\ \boldsymbol{u}_\mathrm{min} \leqslant \boldsymbol{u} \leqslant \boldsymbol{u}_\mathrm{max} \end{cases} \tag{14}$$

To solve the optimal control problem, a control law that minimizes the cost function while satisfying all constraints

must be determined. In the context of direct optimization, the pseudospectral method is a known and successful technique, belonging to the collocation-based methods. The pseudospectral method discretizes the trajectory by representing the state and control variables as the sum of interpolated Lagrange polynomials over a time interval [36]. Thus, the continuous-time optimal control problem is transformed into a finite-dimensional NLP problem. The Lagrange polynomial ϕ_i is expressed as

$$\phi_i(\tau) = \left(\frac{\tau-\tau_0}{\tau_i-\tau_0}\right)\left(\frac{\tau-\tau_1}{\tau_i-\tau_1}\right)\cdots \left(\frac{\tau-\tau_{i-1}}{\tau_i-\tau_{i-1}}\right)\left(\frac{\tau-\tau_{i+1}}{\tau_i-\tau_{i+1}}\right)\cdots\left(\frac{\tau-\tau_N}{\tau_i-\tau_N}\right) \quad (15)$$

The Lagrange polynomial is expressed over a non-dimensional time interval $\tau \in [-1, 1]$, and if it is evaluated at one of the points τ_j, denoted as nodes, it is equal to zero or one, as Eq. (16):

$$\phi_i(\tau_j) = \delta_{i,j}\ , \quad \text{if } j \in \{0, 1, \cdots, N\} \quad (16)$$

Considering a dynamic system of state $\boldsymbol{x}$ with dimension d_x and control $\boldsymbol{u}$ with dimension d_u, the interpolation for the state and control over $N+1$ nodes or interpolation points is given by

$$\boldsymbol{x}(\tau) = \sum_{i=0}^{N} \phi_i(\tau)\boldsymbol{x}_i\ , \quad \boldsymbol{u}(\tau) = \sum_{i=0}^{N} \phi_i(\tau)\boldsymbol{u}_i \quad (17)$$

Note that the number of nodes is related to the degree of the overall polynomial.

For interpolation points in the interval $[-1, 1]$, there are some common choices with suitable quadrature properties, where these set of points are obtained as roots of Legendre polynomials or of their derivative [40]. Legendre–Gauss points (LG) are the most commonly used, which correspond to the roots of the Nth degree Legendre polynomial. They do not include the extreme points -1 and 1. The Legendre–Gauss–Radau (LGR) roots are $p_N(\tau)+p_{N-1}(\tau)$, where $p_N(\tau)$ is the Nth degree Legendre polynomial. These include only initial or final points. The Legendr–Gauss–Lobatto (LGL) are the roots of $\dot{p}_{N-1}(\tau)$, with the addition of the extreme points -1 and 1. In this study, the LGL points are considered because they include both terminal points; therefore, for a Lagrange polynomial of degree N, there are $N+1$ collocation points.

To enforce the dynamic constraints, the continuous-time differential equations are transformed into a finite set of algebraic equality constraints imposed at the nodes. This transformation is achieved by evaluating the system dynamics (Eq. (11)) for each node, where the resulting constraints link the state and control variables at adjacent nodes, effectively capturing the dynamic behavior of the system throughout the trajectory, as Eq. (18):

$$\dot{\boldsymbol{x}}(\tau_j) = \sum_{i=0}^{N} \boldsymbol{x}_i\dot{\phi}_i(\tau_j) = \frac{\Delta t_\mathrm{f}}{2}\boldsymbol{f}(\boldsymbol{x}_j, \boldsymbol{u}_j, \tau_j) \quad (18)$$

where $\Delta t_\mathrm{f}/2 = (t_\mathrm{f} - t_0)/2$ appears because the time interval is normalized by $[-1, 1]$. Eq. (18) is an equality between vectors of dimension d and the number of state variables. There are $N+1$ of such equalities that constitute the set of dynamics constraints, which can be expressed compactly in matrix form, as Eq. (19):

$$\begin{bmatrix} \dot{\phi}_{0,0}I_{d_x} & \dot{\phi}_{1,0}I_{d_x} & \cdots & \dot{\phi}_{N,0}I_{d_x} \\ \dot{\phi}_{0,1}I_{d_x} & \dot{\phi}_{1,1}I_{d_x} & \cdots & \dot{\phi}_{N,1}I_{d_x} \\ \vdots & \vdots & \ddots & \vdots \\ \dot{\phi}_{0,N}I_{d_x} & \dot{\phi}_{1,N}I_{d_x} & \cdots & \dot{\phi}_{N,N}I_{d_x} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix} = \frac{\Delta t_\mathrm{f}}{2} \begin{bmatrix} \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{u}_0, \tau_0) \\ \boldsymbol{f}(\boldsymbol{x}_1, \boldsymbol{u}_1, \tau_1) \\ \vdots \\ \boldsymbol{f}(\boldsymbol{x}_N, \boldsymbol{u}_N, \tau_N) \end{bmatrix} \quad (19)$$

$$D_\mathrm{tot}X_\mathrm{tot} = \frac{\Delta t_\mathrm{f}}{2}F_\mathrm{tot}(X_\mathrm{tot}, U_\mathrm{tot}) \quad (20)$$

where I_d is an identity matrix of dimensions d, and $\dot{\phi}_{j,i} = \dot{\phi}_i(\tau_j)$. The matrices in Eq. (20) depend on the variable X_tot, which includes all the state vectors at the nodes and is therefore a vector of dimension $d_x(N+1)$, and U_tot, which is analogous to a vector composed of all the control vectors at the nodes and has dimension $d_u(N+1)$.

Before including Eq. (20) as a constraint in the QUBO problem, two steps are required: linearizing F_tot and representing the variables of the problem $X_\mathrm{tot}, U_\mathrm{tot}$ as binary variables Z_tot. Dynamic linearization is expressed as

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \tau) \simeq \boldsymbol{f}(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{u}}_i, \tau_i) + \hat{A}_i(\boldsymbol{x} - \hat{\boldsymbol{x}}_i) + \hat{B}_i(\boldsymbol{u} - \hat{\boldsymbol{u}}_i) \quad (21)$$

Because the linearization is realized around a reference point $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{u}}$, a reference trajectory that can coincide with the first guess is required.

The linearized constraints (Δ_tot) are expressed in the matrix form:

$$\Delta_\mathrm{tot} = D_\mathrm{tot}X_\mathrm{tot} - \frac{\Delta t_\mathrm{f}}{2}[A_\mathrm{tot}X_\mathrm{tot} + B_\mathrm{tot}U_\mathrm{tot} - (A_\mathrm{tot}\hat{X}_\mathrm{tot} + B_\mathrm{tot}\hat{U}_\mathrm{tot} - \hat{F}_\mathrm{tot})] \stackrel{!}{=} 0 \quad (22)$$

where the matricies A_tot, B_tot are composed of the $\hat{A}_i = (\partial \boldsymbol{f}/\partial \boldsymbol{x})|_{\boldsymbol{x}_i}$, $\hat{B}_i = (\partial \boldsymbol{f}/\partial \boldsymbol{u})|_{\boldsymbol{u}_i}$ sub-matricies with a block-

diagonal structure. The terms in parentheses on the right side of Eq. (22) includes the known terms linked to the reference trajectory points—$\hat{b}$.

Suppose that the cost indices in Eq. (10) is reduced to

$$J_U = \int_{-1}^{1} \boldsymbol{u}(\tau)^{\mathrm{T}} \boldsymbol{u}(\tau) \mathrm{d}\tau = U_{\mathrm{tot}}^{\mathrm{T}} U_{\mathrm{tot}} \tag{23}$$

where the objective is to minimize the control over the time interval $\tau \in [-1, 1]$ and Lagrange polynomial interpolation is employed. The constraints expressed in Eq. (22) can be enforced using a penalty strategy, leading to a comprehensive cost function as Eq. (24):

$$J = c_u \, U_{\mathrm{tot}}^{\mathrm{T}} U_{\mathrm{tot}} + P \, \Delta_{\mathrm{tot}}^{\mathrm{T}} \Delta_{\mathrm{tot}} \tag{24}$$

where c_u and P are the opportune weights for the control minimization and constraint satisfaction terms, respectively.

By stacking the variables in a single array $Y_{\mathrm{tot}} = [X_{\mathrm{tot}}^{\mathrm{T}} \quad U_{\mathrm{tot}}^{\mathrm{T}}]^{\mathrm{T}}$, the constraints Δ_{tot} can be written to highlight the dependence on the optimization variables, as Eq. (25):

$$\Delta_{\mathrm{tot}}^{\mathrm{T}} \Delta_{\mathrm{tot}} = Y_{\mathrm{tot}}^{\mathrm{T}} Q_{\Delta} Y_{\mathrm{tot}} + 2\bar{q}^{\mathrm{T}} Y_{\mathrm{tot}} + \left(\frac{\Delta t_{\mathrm{f}}}{2}\right)^2 \hat{b}^{\mathrm{T}} \hat{b} \tag{25}$$

where the matrix Q_{Δ} and vector $\bar{q}$ depend on the matrices and vectors D_{tot}, A_{tot}, B_{tot}, $\hat{b}$ by simple matrix multiplication operations. The cost function is therefore expressed as

$$\begin{aligned} J = {} & Y_{\mathrm{tot}}^{\mathrm{T}} C_U Y_{\mathrm{tot}} \\ & + P \left[Y_{\mathrm{tot}}^{\mathrm{T}} Q_{\Delta} Y_{\mathrm{tot}} + 2\bar{q}^{\mathrm{T}} Y_{\mathrm{tot}} + \left(\frac{\Delta t_{\mathrm{f}}}{2}\right)^2 \hat{b}^{\mathrm{T}} \hat{b} \right] \end{aligned} \tag{26}$$

where C_U is a diagonal matrix whose first $d_x(N+1)$ diagonal elements are null (corresponding to the X_{tot} variables) and the remaining are equal to the weight value c_u (corresponding to U_{tot}).

In the formulation above, both the initial and final states, $\boldsymbol{x}_0$, $\boldsymbol{x}_N$, are free optimization variables; therefore, additional terms should be added to the cost function to impose the terminal constraints. Otherwise, the terminal constraints cannot be included because $\boldsymbol{x}_0$, $\boldsymbol{x}_N$ is fixed and therefore excluded from the X_{tot} vector of the state variables. In this case, let $\tilde{X}_{\mathrm{tot}} = [\boldsymbol{x}_1^{\mathrm{T}} \cdots \boldsymbol{x}_{N-1}^{\mathrm{T}}]^{\mathrm{T}}$, which differs from X_{tot} because it lacks initial and final states, and has $d_x(N-1)$ components. Additionally, the number of dynamic constraints remains equal to $d_x(N+1)$. In matrix form, the dynamical constraints become $\tilde{D}_{\mathrm{tot}} \tilde{X}_{\mathrm{tot}} + b_{\mathrm{TC}} = (\Delta t_{\mathrm{f}}/2) \, F_{\mathrm{tot}}(\tilde{X}_{\mathrm{tot}}, U_{\mathrm{tot}})$, where $\tilde{D}_{\mathrm{tot}}$ is modified by excluding the first and last d_x columns, which are part of an additional known term b_{TC} that is related to the $\boldsymbol{x}_0$, $\boldsymbol{x}_N$ fixed nodes, as Eq. (27):

$$\begin{aligned} \tilde{D}_{\mathrm{tot}} &= \begin{bmatrix} \dot{\phi}_{1,0} I_{d_x} & \dot{\phi}_{2,0} I_{d_x} & \cdots & \dot{\phi}_{N-1,0} I_{d_x} \\ \dot{\phi}_{1,1} I_{d_x} & \dot{\phi}_{2,1} I_{d_x} & \cdots & \dot{\phi}_{N-1,1} I_{d_x} \\ \vdots & \vdots & \ddots & \vdots \\ \dot{\phi}_{1,N} I_{d_x} & \dot{\phi}_{2,N} I_{d_x} & \cdots & \dot{\phi}_{N-1,N} I_{d_x} \end{bmatrix}, \\ b_{\mathrm{TC}} &= \begin{bmatrix} \dot{\phi}_{0,0} \boldsymbol{x}_0 + \dot{\phi}_{N,0} \boldsymbol{x}_N \\ \dot{\phi}_{0,1} \boldsymbol{x}_0 + \dot{\phi}_{N,1} \boldsymbol{x}_N \\ \vdots \\ \dot{\phi}_{0,N} \boldsymbol{x}_0 + \dot{\phi}_{N,N} \boldsymbol{x}_N \end{bmatrix} \end{aligned} \tag{27}$$

The size of the $\tilde{D}_{\mathrm{tot}}$ matrix is $d_x(N+1) \times d_x(N-1)$. The Δ_{tot} expression can be rewritten as

$$\begin{aligned} \Delta_{\mathrm{tot}} = {} & \tilde{D}_{\mathrm{tot}} \tilde{X}_{\mathrm{tot}} - \frac{\Delta t_{\mathrm{f}}}{2} \left(\tilde{A}_{\mathrm{tot}} \tilde{X}_{\mathrm{tot}} + B_{\mathrm{tot}} U_{\mathrm{tot}} \right) \\ & + \left(\frac{\Delta t_{\mathrm{f}}}{2} \hat{b} + b_{\mathrm{TC}} \right) \end{aligned} \tag{28}$$

The cost function J can be redefined in an equivalent way, as Eq. (29):

$$\begin{aligned} J = {} & \tilde{Y}_{\mathrm{tot}}^{\mathrm{T}} \tilde{C}_U \tilde{Y}_{\mathrm{tot}} + P \Bigg[\tilde{Y}_{\mathrm{tot}}^{\mathrm{T}} \tilde{Q}_{\Delta} \tilde{Y}_{\mathrm{tot}} + 2\tilde{q}^{\mathrm{T}} \tilde{Y}_{\mathrm{tot}} \\ & + \left(\frac{\Delta t_{\mathrm{f}}}{2} \hat{b} + b_{\mathrm{TC}} \right)^{\mathrm{T}} \left(\frac{\Delta t_{\mathrm{f}}}{2} \hat{b} + b_{\mathrm{TC}} \right) \Bigg] \end{aligned} \tag{29}$$

The matrix and vector $\tilde{Q}_{\Delta}$, $\tilde{q}$ are obtained analogously and depend on $\tilde{D}_{\mathrm{tot}}$, $\tilde{A}_{\mathrm{tot}}$, B_{tot}, $\hat{b}$, b_{TC}. With respect to A_{tot}, $\tilde{A}_{\mathrm{tot}}$ is a $d_x(N+1) \times d_x(N-1)$ matrix that lacks the first and last d_x columns. The vector of variables $\tilde{Y}_{\mathrm{tot}} = [\tilde{X}_{\mathrm{tot}}^{\mathrm{T}} \quad U_{\mathrm{tot}}^{\mathrm{T}}]^{\mathrm{T}}$ contains $d_x(N-1) + d_u(N+1)$ real-valued components.

To obtain the QUBO formulation, each real variable must be expressed using n_{b} binary variables. In this case, the boundary constraints in Eq. (14) should be imposed. Therefore, a fixed-point binary representation is sought such that the maximum and minimum real values that can be represented correspond to the boundaries of each state and control the real variables. When every $z_k = 0$, $y = y_{\min}$ and $z_k = 1$, $y = y_{\max}$, we express the real variable y as Eq. (30):

$$\begin{aligned} y = {} & y_{\min} + \frac{y_{\max} - y_{\min}}{2^{n_{\mathrm{b}}} - 1} \sum_{k=0}^{n_{\mathrm{b}}-1} 2^k z_k = y_{\min} \\ & + \frac{y_{\max} - y_{\min}}{2^{n_{\mathrm{b}}} - 1} \boldsymbol{g}^{\mathrm{T}} \boldsymbol{z} \quad \text{with } z_k = \{0, 1\} \end{aligned} \tag{30}$$

The complete Y_{tot} or $\tilde{Y}_{\mathrm{tot}}$ is expressed in terms of all $n_{\mathrm{b}} n_{\mathrm{real}}$ binary variables collected in the vector Z_{tot}, as Eq. (31):

$$Y_{\mathrm{tot}} = Y_{\min} + G Z_{\mathrm{tot}} \tag{31}$$

where the matrix G has a size $n_{\text{real}} \times n_{\text{b}} n_{\text{real}}$ and is defined by

$$G = \frac{1}{2^{n_{\text{b}}} - 1} \begin{bmatrix} (y_{\max_1} - y_{\min_1})\boldsymbol{g}^{\text{T}} & 0 & \cdots & 0 \\ 0 & (y_{\max_2} - y_{\min_2})\boldsymbol{g}^{\text{T}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (y_{\max_{(n_{\text{real}} n_{\text{b}})}} - y_{\min_{(n_{\text{real}} n_{\text{b}})}})\boldsymbol{g}^{\text{T}} \end{bmatrix} \tag{32}$$

By substituting Eq. (31) into Eqs. (26) and (29), an expression for the cost function in terms of the binary variables Z_{tot} is obtained as

$$J = Z_{\text{tot}}^{\text{T}} L Z_{\text{tot}} + \bar{m}^{\text{T}} Z_{\text{tot}} + N = Z_{\text{tot}}^{\text{T}} L' Z_{\text{tot}} + N \tag{33}$$

where L is the matrix of quadratic terms, $\bar{m}$ is the vector of linear terms, and N is a constant. Because, for binary variables $z_k^2 = z_k$, the linear term is included in the quadratic matrix, the formulation on the right-hand side of Eq. (33), with $L' = L + \text{diag}(\bar{m})$, and the constant N can be ignored for optimization purposes.

The methodology described above allows the linearized trajectory optimization to be formulated as a QUBO problem, where the objective is to minimize the control employed, and dynamical and boundary constraints are included. As shown in Fig. 3, a sequential approach is used to generalize the method and alleviate the linearization using a reference trajectory $[\hat{X}_{\text{tot}}^{\text{T}} \quad \hat{U}_{\text{tot}}^{\text{T}}]^{\text{T}}$ around which the dynamics are linearized. Subsequently, the QUBO problem is solved and the solution is used as a new reference trajectory, which in turn gives a new QUBO. This process is repeated until an optimal solution is achieved. Because the boundaries play an important role in guiding the sequential optimization process, they must be chosen such that the linear dynamics is a good approximation of the nonlinear dynamics. To do this, several strategies can be implemented by varying the boundaries during the process.

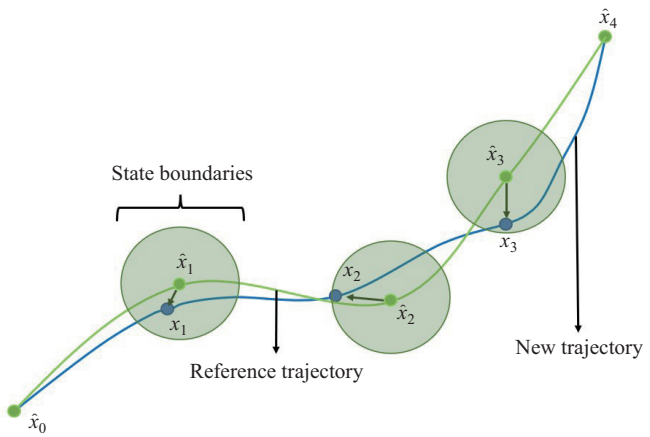


Fig. 3 Visualization of two sequential iteration during QUBO-based optimization procedure with initial and final state fixed and boundaries visualized as circles.

In the transcription procedure, the flight time Δt_{f} must be fixed because the optimization must be reduced to the QUBO form. In fact, if Δt_{f} is a variable, it is X_{tot} multiplied by U_{tot} in Eq. (22), causing $\Delta_{\text{tot}}^{\text{T}} \Delta_{\text{tot}}$ to have fourth-degree terms, which in turn causes the binary cost function to be a fourth-order binary polynomial with interacting triplets and quartets of binary variables.

4 Low-thrust Earth–Mars transfer problem

In this section, the Earth–Mars transfer problem with continuous thrust is presented as the central test case for the proposed transcription procedure and optimization by QA. The Earth–Mars transfer is a classical and well-studied trajectory optimization problem found in the literature for many different declinations and is often used to test new algorithms and techniques. In this study, the dimensions of the direct optimization problem must be sufficiently reduced so that it can be implemented on a quantum annealer. Current machines can only deal with a limited number of (binary) variables and interactions between the variables. Therefore, a series of simplifications were considered to reduce the binary variables required to represent the problem.

First, the orbits of the Earth and Mars were considered circular and coplanar, respectively, assuming that their orbital planes align precisely; therefore, the transfer problem was planar, with a state vector composed of four variables. Furthermore, two-body dynamics were considered, as they are customary in the preliminary interplanetary trajectory design. The equations of motion are expressed in Eq. (34) and expressed in polar coordinates. r, θ, v_r, and v_θ are the radial distance from the Sun, angular position, and radial and angular velocities, respectively. $\mu_{\text{S}} = GM$, where G is the universal gravitational constant and M is the mass of the Sun. u_r and u_θ are the accelerations given by the thrust and represent the controls of the optimal control problem. The variables are non-dimensional in such a way that the astronomical unit is the unit length and $\mu_{\text{S}} = 1$.

$$\dot{r} = v_r$$

$$\begin{aligned}\dot{\theta} &= \frac{v_\theta}{r}\\ \dot{v}_r &= -\frac{\mu_S}{r^2} + \frac{v_\theta^2}{r} + u_r\\ \dot{v}_\theta &= -\frac{v_r v_\theta}{r} + u_\theta\end{aligned} \tag{34}$$

The boundary conditions are expressed by

$$\begin{cases} r(t_0) = R_E, \quad v_r(t_0) = 0, \quad v_\theta(t_0) = V_E \\ r(t_f) = R_M, \quad v_r(t_f) = 0, \quad v_\theta(t_f) = V_M \end{cases} \tag{35}$$

where R_E and R_M are the radial distances of the Earth and Mars, and V_E and V_M are the orbital velocities of the Earth and Mars, respectively.

Because we aimed to determine the minimum-fuel trajectory that enabled the spacecraft to transition from Earth's orbit to Mar's orbit, the cost function, in line with the transcription procedure described in Section 3, assumes the form in Eq. (36):

$$J = \int_{t_0}^{t_f} (u_r(t)^2 + u_\theta(t)^2)\mathrm{d}t = \int_{t_0}^{t_f} \boldsymbol{u}(t)^T \boldsymbol{u}(t)\mathrm{d}t \tag{36}$$

Furthermore, boundary inequality constraints on the state and control variables were added, as shown in Eq. (14):

Note that the state variable θ does not appear in the terminal constraints (the trajectory can end at any point on Mars' orbit) or in the cost function; therefore, it was excluded from the optimization problem.

Subsequently, the angular position can be computed from the optimal trajectory by integrating only $\dot{\theta} = v_\theta(t)/r(t)$ given $r(t)$ and $v_\theta(t)$. Therefore, the number of state variables was reduced to $d_x = 3$ and the number of control variables was $d_u = 2$. Once the number of nodes of the direct transcription and bits per real variable were defined, the total number of real and binary variables were computed as described in Section 3.

In addition to the planar, circular, and two-body dynamic hypotheses mentioned previously, the following simplifications were considered:

(1) The trajectory optimization was considered with a fixed time of flight t_f since, as mentioned in Section 3, considering it an optimization variable would result in a nonquadratic cost function expression.

(2) The spacecraft mass was considered constant to avoid including either an additional state variable or increasing the non-linearity of the problem. Low-thrust high specific impulse propulsion was considered, which justifies the assumption.

(3) As implied by the way in which the boundary constraints on $\boldsymbol{u}$ are expressed in Eq. (14), the control variables are u_r, u_θ and were considered individually bounded. Therefore, $\boldsymbol{u}$ was not bounded at every time but rather each component of the vector. This was done to avoid including a nonlinear inequality constraint on the norm of the control, which would be more difficult to handle.

Furthermore, the initial and final points $\boldsymbol{x}_0$ and $\boldsymbol{x}_N$ were fixed, as explained in the previous section, allowing the implicit inclusion of terminal constraints in Δ_{tot} and reducing the overall number of variables.

The particular instance of the problem considered in this study had an admissible maximum acceleration value of 10^{-4} m/s^2, which corresponds to 10 mN for every 100 kg of mass, and a fixed time of flight of 2 yr.

By employing the transcription procedure, the Earth–Mars transfer problem was translated into the QUBO form, where its size depended on the nodes and n_b values chosen, while its sparsity was determined partly by the intrinsic structure of the pseudo-spectral formulation and partly by the specific problem dynamics.

As described in Section 2, a quantum annealer is composed of a lattice of interconnected qubits based on a precise topology. When encoding a general QUBO onto a quantum annealer, the total number of available qubits, which limits the maximum number of binary variables, and the total number of interconnections, which limits the maximum number of quadratic terms or off-diagonal nonzero elements in the QUBO matrix, must be considered. The quantum annealer employed during the development of this study was the D-Wave Advantage [41], which has approximately 5000 qubits available for computation and a topology called *Pegasus*, where each qubit is connected to another 15, with approximately 35,000 interconnections.

Table 1 shows the number of nonzero quadratic elements of the QUBO matrix with varying N (degree of polynomial) and n_b (number of bits for the real variable) for the problem described. It appears that only a small number of nodes and bits are valid. Furthermore, the problem size that can be implemented practically is below the maximum number of annealer connections. In fact, there is an *embedding* process to consider because it creates chains of qubits that are natively connected in the annealer working graph, and not all 35,000 connections can be directly exploited. With the D-Wave Advantage, *embedding* can be performed using automatic heuristic

algorithms, as shown in Table 1, and the problems with embedding are highlighted in bold and blue.

Considering the value of the number of nodes that appear in Table 1 for the Earth–Mars transfer problem, the pseudospectral method allows interpolated trajectories to be obtained with sufficient precision, even with a low N, owing to the difference between the interpolated and numerically propagated trajectories, both with the same control history. Note that because the initial and final points are fixed, the interpolated trajectory always starts and ends precisely at the terminal points $\boldsymbol{x}_0$ and $\boldsymbol{x}_N$. If the problem is properly solved and the dynamical constraints, given by Δ_{tot}, are satisfied, then the numerically propagated trajectory will terminate at the final point, in general, with an error determined by how well Δ_{tot} are satisfied and how well the polynomial interpolation approximates the real dynamics. Therefore, the final error between the numerically propagated trajectory and the required final state can be considered a metric of the convergence quality of a solution.

For example, the transfer solved using traditional methods (with real variables, the full nonlinear problem, and an NLP problem commercial solver), where $N = 7$, is shown in Fig. 4). The final errors between the propagated and desired final states were $\Delta r = 60 \times 10^3$ km, $\Delta v_r = 3$ m/s, and $\Delta v_\theta = 6$ m/s. The difference between the interpolated and propagated trajectories remained in the order of 10^{-4} in nondimensional units. ΔV, computed as the integral of the norm of $\boldsymbol{u}$, was 6640 m/s.

5 Results and discussion

In this section, the Earth–Mars transfer problem with the QUBO transcription is addressed using the samplers available through the D-Wave cloud service "Leap" [42]. D-Wave systems provide a cloud-computing platform for accessing quantum computers, where users can interact with quantum processors utilizing their Python-based SDK, called Ocean. Through this service, it is possible to access the advantageous QA chip as well as hybrid solvers. Leap allows for the programming of a QUBO problem fairly easily through the Python interface. Once the QUBO is properly defined, a solver (or sampler)

Table 1 Number of off-diagonal elements of QUBO matrix for different values of nodes and bits. Values below 35,000 are highlighted in green and italics, and values with embedding are highlighted in blue and bold

	$n_b = 6$	$n_b = 7$	$n_b = 8$	$n_b = 9$	$n_b = 10$	$n_b = 11$	$n_b = 12$
$N = 4$	**6300**	**8575**	**11,200**	**14,175**	**17,500**	*21,175*	*25,200*
$N = 5$	**10,224**	**13,916**	**18,176**	*23,004*	*28,400*	*34,364*	40,896
$N = 6$	**15,084**	**20,531**	*26,816*	*33,939*	41,900	50,699	60,336
$N = 7$	*20,880*	*28,420*	37,120	46,980	58,000	70,180	83,520
$N = 8$	*27,612*	37,583	49,088	62,127	76,700	92,807	110,448
$N = 9$	35,280	48,020	62,720	79,380	98,000	118,580	141,120
$N = 10$	43,884	59,731	78,016	98,739	121,900	147,499	175,536

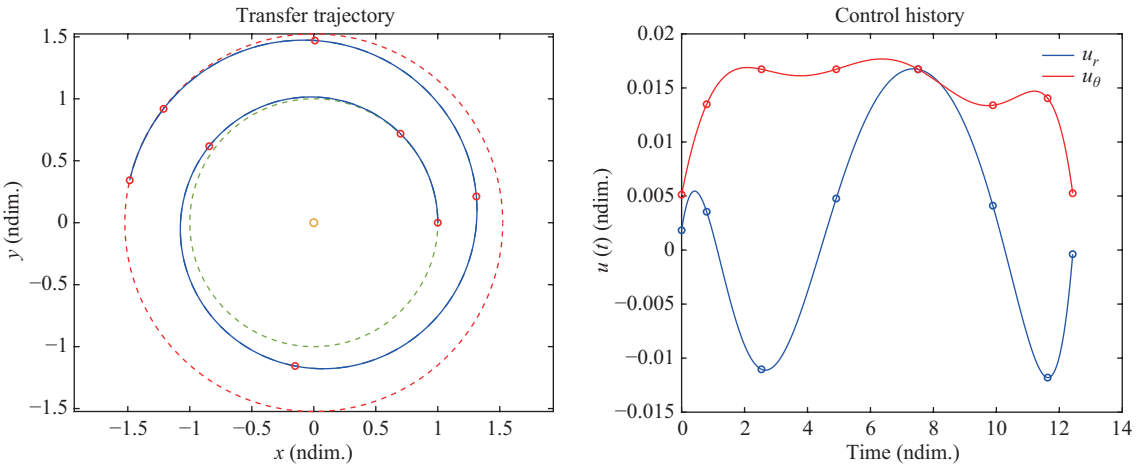


Fig. 4 Classical solution with $N = 7$. Trajectory shown on left, where red circles represent node positions. Control history of u_r and u_θ shown on right, where circles represent nodes.

sends the problem to the quantum annealer. Once the annealer finishes its computation, the result is sent back to the user and post-processed again through the Python interface. While basic access is typically free (limited to one minute per month), costs may apply for larger computations or extended usage. This platform offers developers a convenient means of harnessing QC power for optimization problems.

First, the results with hybrid solvers are presented, followed by a consideration of the number of iterations and computational time, and a comparison with the traditional solution. The results obtained using QPU are then presented and discussed, focusing on the current limitations of quantum annealers and comparisons with classical and hybrid solutions. The quantities of interest for classifying and assessing the solutions are the number of nodes N, number of bits per real variable n_b, and starting guess, as well as the following:

Final error: the error between the desired final state and the trajectory numerically propagated with the control obtained from the optimization process. This is a measure of the feasibility of the found solution and whether the optimization converges.

Delta V: the integrated norm of the control over the flight time. This is equivalent to the optimization objective, which is a measure of the cost of the propellant trajectory.

Number of iterations: the number of linearized QUBO problems that have been solved prior to convergence.

Optimization time: the time required for the solution of each iteration, obtained directly from the time employed by the D-Wave solver.

Wall time: the total time measured for each iteration. Different from the optimization time, it also includes the time required to re-elaborate the optimizer output and the internet delay to access the solver and retrieve the results. Furthermore, it is dependent on the computer on which the code is run. In our case, it was run on the Leap cloud.

5.1 Hybrid results

In general, it was found that through the hybrid solver, the problem was easily solved, and the solutions obtained were comparable to those of the traditional solver. The sampler employed for these cases was the "LeapHybridSampler". The first solution presented had $N = 7$ and $n_\mathrm{b} = 10$ for 340 binary variables. First, the trajectory was considered to have the following control law with purely tangential thrust: $u_r(t) = 0$, $u_\theta(t) = u_\mathrm{max}$, which is a basic estimate that produces a good starting point. A plot of the first estimated trajectory is shown in Fig. 5.

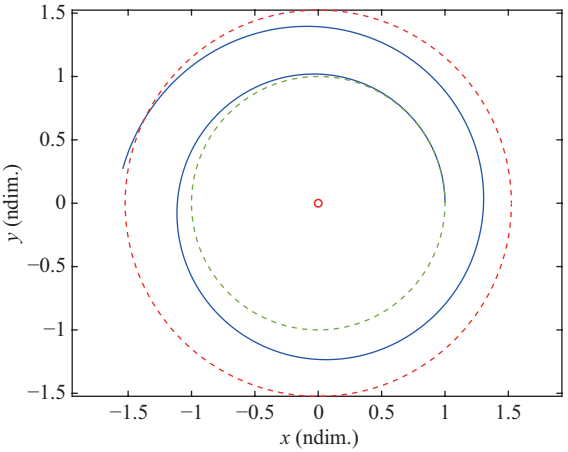


Fig. 5 First guess trajectory with $u_r(t) = 0$, $u_\theta(t) = u_\mathrm{max}$.

The hybrid solver was consistently capable of converging after a few iterations. Here, the solution presented has six iterations, $\Delta V = 6577.16$ m/s, and final errors of $\Delta r = 75 \times 10^3$ km, $\Delta v_r = 14.8$ m/s, and $\Delta v_\theta = 5.3$ m/s. The optimization time of the hybrid sampler can be considered fixed at 3 s per iteration with minimal variations, which is a parameter of the solver and, for the current problem, cannot be reduced by the user. The wall time remained almost constant at approximately 10 s/iteration. The trajectory and control histories are presented in Fig. 6.

To test the capability of the hybrid solver to converge with a more difficult starting guess, the first guess, in which the control was set to zero for the entire flight duration, was set to $u_r(t) = 0$, $u_\theta(t) = 0$. Therefore, the resulting starting trajectory remained along the Earth's orbit.

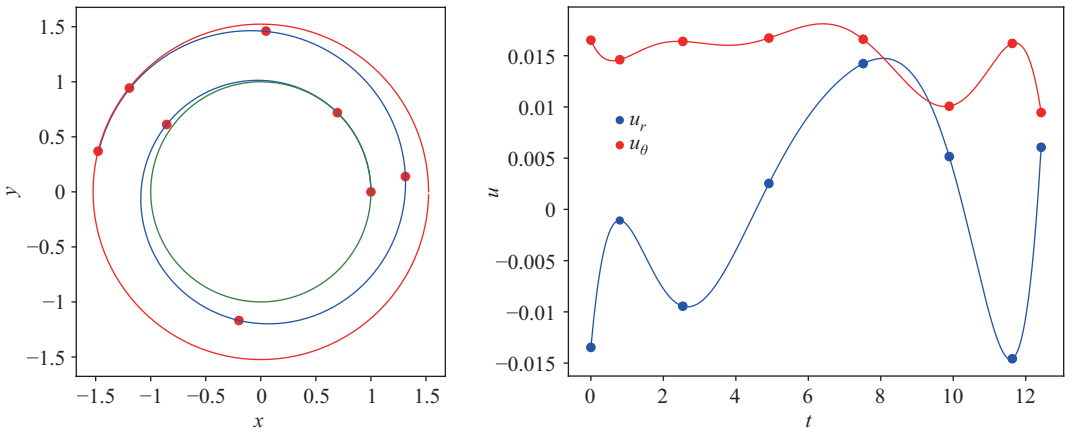
The results show that for this initial guess, which was far from the solution, the optimization procedure converged to a trajectory similar to that of the previous case. However, additional iterations were required. The presented solution required 13 iterations, $\Delta V = 6676.38$ m/s, and the final errors were $\Delta r = 27 \times 10^3$ km, $\Delta v_r = 17.7$ m/s, and $\Delta v_\theta = 1.8$ m/s. The optimization and wall times were comparable to the previous case for every

iteration. The trajectory and control histories are shown in Fig. 7.

The results of Figs. 6 and 7 highlight the importance of setting boundary constraints for the state variables and control. In fact, $\boldsymbol{x}_{\min}$, $\boldsymbol{x}_{\max}$, $\boldsymbol{u}_{\min}$, and $\boldsymbol{u}_{\max}$ are user-definable parameters with the greatest influence on the optimization procedure and resulting trajectory. For example, in the first case, they were chosen as constants among iterations, with the same upper and lower values for every state variable ($\boldsymbol{x}_{\min\,/\,\max} = 0.05$, $\boldsymbol{u}_{\min\,/\,\max} = 0.005$ in non-dimensional units). Using the same values for the second case would lead to a slower convergence because the first guess is far from the actual solution, and larger search areas allow faster transfer trajectories to be attained. While it is possible to converge with the "zero thrust" first guess by setting the boundaries as constant with adequate values, in the solution shown in Fig. 7, a boundary reduction strategy was applied. In particular, a larger value of the state variables was initially chosen, and was later reduced when the cost function no longer decreased from one iteration to the next, thus allowing a faster initial search, followed by a tighter and more accurate search. This strategy proved to be useful for QPU solutions, as discussed in Section 5.2.

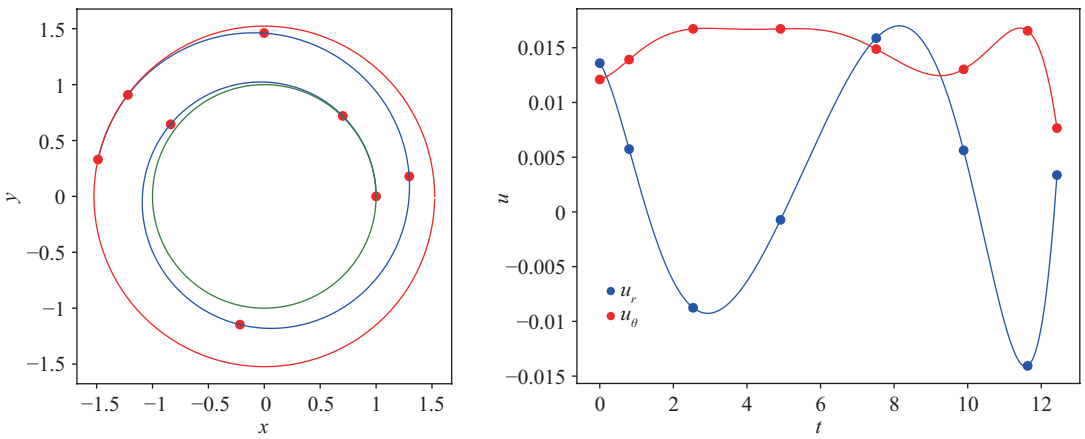
5.2 QPU results

As shown in Table 1, when using the QPU sampler of D-Wave, the dimensions of the implementable problem are significantly constrained. Furthermore, the found embeddings are characterized by long chains; for example, for $N = 4$, $n_{\mathrm{b}} = 7$ and $N = 5$, $n_{\mathrm{b}} = 8$, the maximum chain lengths are 22 and 29, respectively. Subsequently, the problem quickly becomes too large, and it is no longer possible to find an embedding.



(a) Trajectory (b) Control history

Fig. 6 Hybrid solution with $N = 7$, $n_{\mathrm{b}} = 10$. Circles represent nodes positions.



(a) Trajectory (b) Control history

Fig. 7 Hybrid solution with $N = 7$, $n_{\mathrm{b}} = 10$ and "zero thrust" starting guess. Circles represent nodes positions.

With the allowed values of N, the polynomial approximation of the trajectory starts to lose accuracy, while the value of n_b requires the real variables to be bounded quite tightly to obtain a sufficiently precise binary representation, while long chain lengths are more likely to break and produce inaccurate solutions.

Considering these limitations due to the capability of currently available machines, we attempted to solve the Earth–Mars transfer problem. All results were obtained using the Advantage 4.1 QPU chip.

First, for $N = 4$, $n_\mathrm{b} = 7$, the structure of the ensuing QUBO problem is shown in Fig. 8, where the y- and x-axes coordinates represent respectively the columns and rows of the Q matrix, the nonzero elements are highlighted by blue dots, and the null elements are on the left in white. The sparsity of the matrix, computed as the number of nonzero elements divided by the total number of elements, is equal to $s_Q = 0.48$, and for the range of nodes and bits allowed, values of approximately 0.5 are always obtained. The nonzero elements are not uniformly distributed: the top-left submatrix is dense, whereas the bottom-right submatrix is sparse with a block diagonal structure. This can be related to the structure of the polynomial interpolation by a term appearing in $\bar{Q}_\Delta$ that depends on D_tot and A_tot, and the $\bar{C}_U$ term given by the minimization of the control. The top-right and bottom-left submatrices are given by mixed terms that depend on D_tot, A_tot, and B_tot. The matrix structure depends on the reference trajectory around which linearization is performed; therefore, its numerical values change from iteration to iteration, and, in principle, its structure can vary, although the simulations show that these structural variations can be ignored.

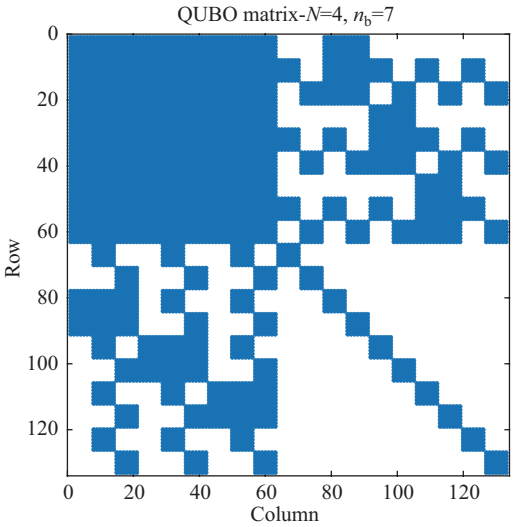


Fig. 8 Example of QUBO matrix structure for the Earth–Mars transfer problem.

A solution with $N = 4$, $n_\mathrm{b} = 7$ is shown in Fig. 9, where the tangential thrust-first estimate is employed. The number of iterations required in this QPU case was significantly higher than that of the hybrid solutions, with 55 sequential QUBO problems. The QPU solutions were unable to handle the same boundaries of the hybrid solutions well, and they behaved better with tighter boundary values. Therefore, the cost function improved less for each iteration and more iterations were required. Additionally, a boundary-reduction strategy had to be applied to improve the solution after a certain point, and the results show that the boundaries were reduced every time the cost function did not improve for two iterations

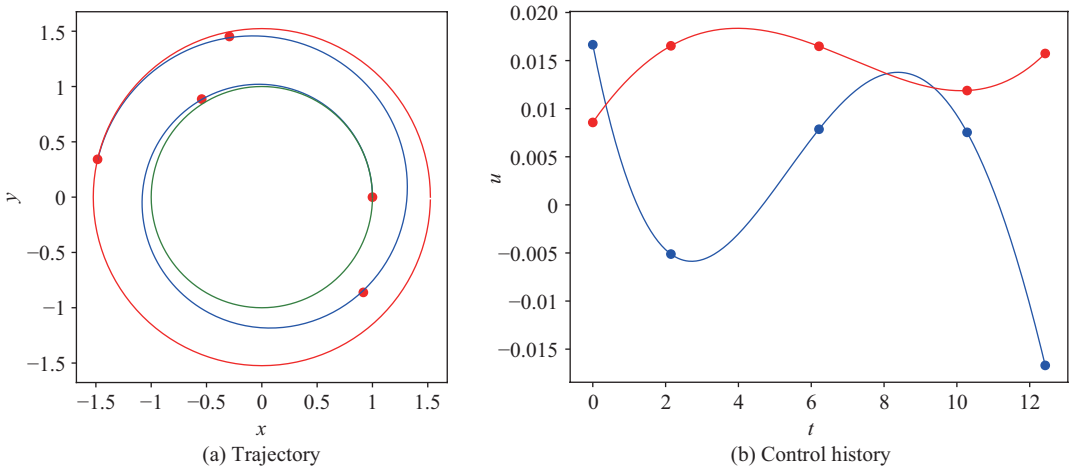


Fig. 9 Solution by QPU with $N = 4$, $n_\mathrm{b} = 7$. First guess is as in Fig. 5. Circles represent node positions.

(not necessarily consecutive). The trajectories are shown in Fig. 9, where two boundary reductions are applied.

Regarding the final errors, acceptable values and full convergence were more difficult to obtain. In Fig. 9, the following values are obtained: $\Delta r = 3.2 \times 10^5$ km, $\Delta v_r = 98.7$ m/s, $\Delta v_\theta = 88.9$ m/s, and $\Delta V = 6333.98$ m/s. The optimization time coincides with the time the Advantage chip was accessed, and it was equal to 54.19 ms for each iteration. The wall time varied in the range of 2.2–3 s.

Another striking difference from the previous hybrid cases is the values of the optimization and wall times, which were approximately 55 and 3 times shorter than those in the hybrid results, respectively. The rapid optimization time is strictly related to the annealing process, which is an important parameter. For the hybrid solver, there was a comparably short time given by the access of the solver compared to that of the QPU solver; however, the classical elaboration and processing of the problem performed by the solver increased the total optimization time. The faster optimization per iteration of the QPU-only solver was partly sacrificed by the higher number of iterations required, as discussed later.

5.3 Discussion

The differences between the first and third solutions obtained with the hybrid and full-quantum solvers, respectively, appear especially in the final error, which is linked to the satisfaction of the dynamical constraints, number of iterations, and optimization time required for each iteration. These two solutions were compared in terms of the decrease in the cost function value as the optimization process advanced. A comparison of the results is presented in Fig. 10, where the three plots, from left to right, compare the solutions based on the number of iterations, cumulative optimization time, and wall time, respectively.

A large difference in the final cost function value between the two solutions appears immediately, which reflects their difference in quality and constraint satisfaction. Although several iterations were executed from the QPU solution compared to only a few in the hybrid solution based on the optimization time, the QPU solution was faster. However, considering the total wall time, even though it was shorter for a single QPU iteration, a higher number of iterations resulted in an longer overall time required to obtain the QPU solution compared to that of the hybrid solution.

Notably, Fig. 10 compares two QUBO representations of the Earth–Mars transfer problem with different sizes; the hybrid solver had 340 binary variables, while the QPU solver had 133. Therefore, although the plots in Fig. 10 can be a visual demonstration of the difference in the current capabilities of the two solvers, the hybrid method provides a more accurate representation of the trajectory owing to the higher N and n_b.

A comparison of the same number of variables is shown in Fig. 11. With respect to Fig. 10, the final values of the cost function are similar, and the final error of the hybrid solution is comparable to that of the QPU solution. Figure 11 shows that a reduction in the dimensions of the problem has an impact on the solution quality.

Based on the considerations shown in Figs. 10 and 11, we attempted to solve the problem of the intermediate dimensions between the two solvers, with $N = 5$, $n_\mathrm{b} = 8$ (192 binary variables). In this case, the hybrid solver was able to achieve better constraint satisfaction in a few iterations, although it was worse than that shown in

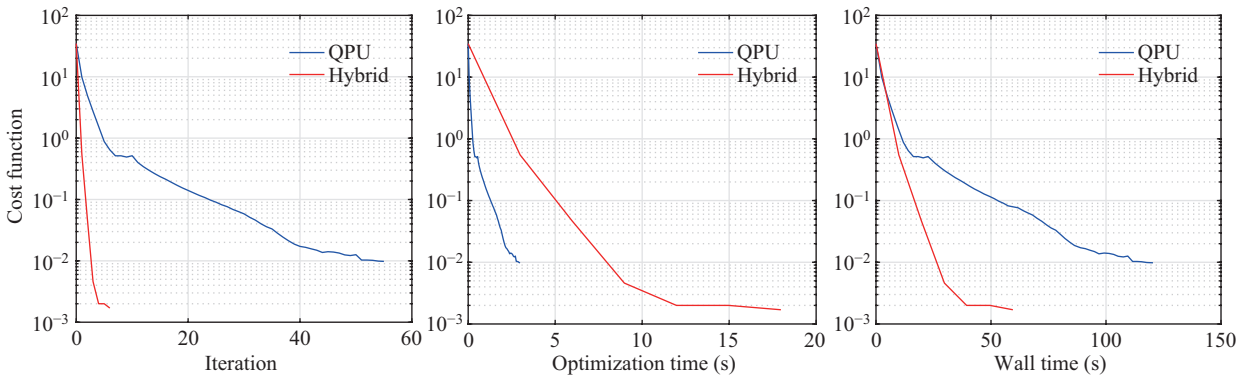


Fig. 10 Comparison between hybrid solution of Fig. 6 and QPU solution of Fig. 9.

Fig. 6 and had a higher ΔV with respect to the previous results. The QPU solver, based on the final error, did not improve in the $N = 4$, $n_b = 7$ case; it also had an increased ΔV, and the cost function continued to decrease for a very long number of iterations, but with diminishing returns. The history of the cost function is presented in Fig. 12 (the wall time plot was omitted) and the final cost function values are slightly different in favor of the hybrid solution.

Figure 13 shows the hybrid and QPU solutions for the last case. The two plots are similar; however, the QPU solver does not reach the same trajectory as the hybrid solver owing to a local search issue. The difference between the two solutions could be because the full-quantum solver faced greater difficulty when handling all the variables as embedding requires long chains (a maximum length of 29) or the overall QUBO problem was excessively sensitive.

The results are summarized in Table 2, where all the main features are reported: error, ΔV cost, iterations, and optimization and wall times. The table includes the hybrid and QPU results specifying the number of nodes and bits. The first two rows differ in the initial guess. In the last two rows, the full-classical solutions are reported. In the second-to-last row, the traditional nonlinear and real-valued solutions are presented, as shown in Section 4; in this case, the optimization was performed using MATLAB, so the times reported may not be immediately comparable with the others because a different computing environment and computer were used. The last row shows a solution to the QUBO problem obtained with the simulated annealing (SA) algorithm, which is available as a solver in the D-Wave Leap. The results were obtained with an initial point equal to the reference trajectory and 300 runs of the algorithm for each iteration. The SA was capable of solving the QUBO problem, with a performance comparable to that of the hybrid solver, but with longer optimization and wall times.

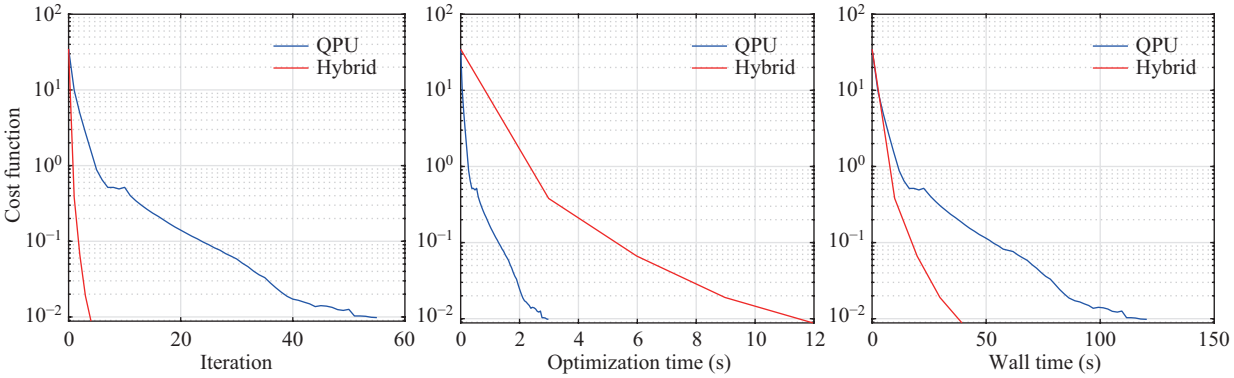


Fig. 11 Comparison between hybrid solution with $N = 4$, $n_b = 7$ and QPU solution of Fig. 9.

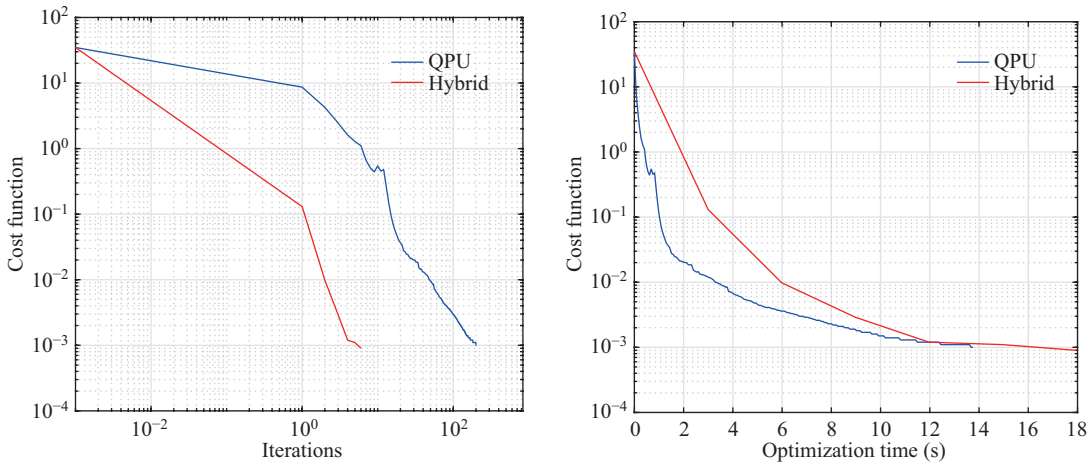
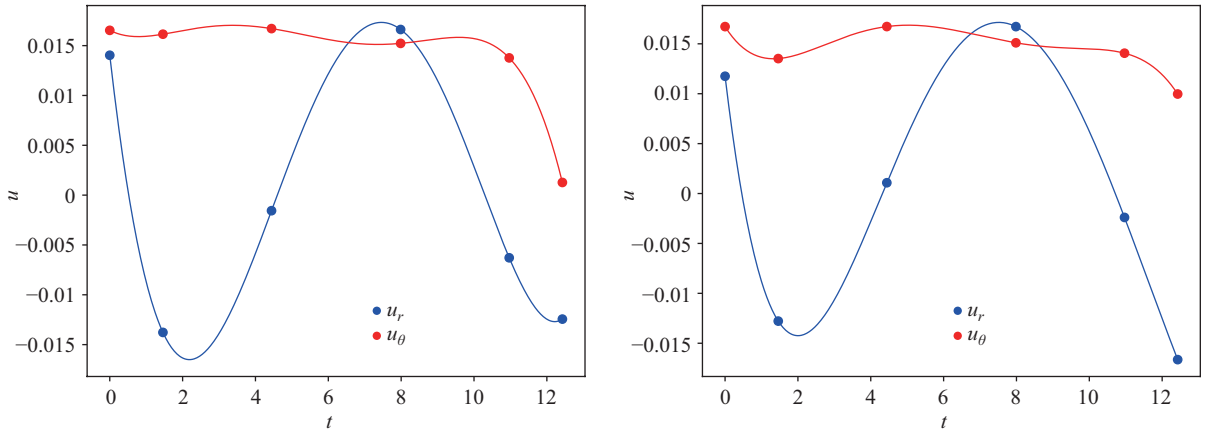


Fig. 12 Comparison between hybrid and QPU solutions with $N = 5$, $n_b = 8$. Left plot has a logarithmic scale on x-axis.



(a) Hybrid solution (a) QPU solution

Fig. 13 Control histories for solutions with $N = 5$, $n_b = 8$

Table 2 Summary of results.

	Δr (km)	Δv_r (m/s)	Δv_θ (m/s)	ΔV (m/s)	N_{iter}	T_{opt} (s)	WT (s)
Hybrid $N = 7$, $n_b = 10$, Fig. 6	7.5×10^4	14.8	5.3	6577.16	6	18.0	59.5
Hybrid $N = 7$, $n_b = 10$, Fig. 7	2.7×10^4	17.7	1.8	6676.38	13	38.9	117.0
Hybrid $N = 4$, $n_b = 7$	5.7×10^5	65.7	92.6	6369.35	4	12.0	39.4
Hybrid $N = 5$, $n_b = 8$	1.9×10^5	4.9	36.9	7025.64	6	17.9	54.0
QPU $N = 4$, $n_b = 7$	3.2×10^5	98.7	88.9	6333.98	55	3.0	120.42
QPU $N = 5$, $n_b = 8$	1.2×10^6	6.3	63.4	6853.91	201	13.7	612.1
Classical $N = 7$	6.0×10^4	3.0	6.0	6640.37	—	2.46	2.46
SA $N = 7$, $n_b = 10$	1.1×10^5	7.8	0.9	6434.09	5	57.83	65.11

6 Conclusions

In this study, QA was explored for optimizing spacecraft trajectories. Because a specific formulation is required to employ QA, a transcription procedure was proposed to transform a general trajectory problem into the QUBO form. The transcription procedure uses the pseudospectral method, linearization, and the binary representation of real variables. Using a sequential approach and starting a reference trajectory, nonlinear problems can be tackled using this method.

An Earth–Mars low-thrust transfer was considered as a case study. Employing the D-Wave cloud computing platform, Leap, to access real quantum annealers, hybrid and QPU solvers were tested. The hybrid solver used classical and quantum resources and alleviated the severe restrictions of full-quantum solvers for the case study. The QPU solver required the problem dimensions to be reduced and performed an embedding of the QUBO problem.

The solutions obtained using the hybrid solver were promising, returning trajectories comparable to the classical nonlinear solution of the problem. The hybrid solver can converge after a few iterations and is robust to initial guesses that are far from the solution. Although, in this case study, more iterations were required.

As mentioned previously, QPU solvers are constrained by the maximum number of variables. With $N = 4$, $n_b = 7$, equivalent to 133 binary variables, the QPU solver achieved trajectories that were close to convergence, but did not reach them entirely, presenting higher final state errors. Many more iterations were required compared the hybrid solver, and the cost function decreased more slowly for each iteration. However, the single iterations were much faster, and the total optimization time was shorter. The total wall time, including the communication time within the cloud and internet latency as well as the waiting time to access the solvers, was longer. Both time measures can be considered important in testing the QA optimization, and the optimization time was more indicative of the QA/hybrid solver speed, whereas the wall time provided a measure of the entire process but was affected by the cloud computing platform characteristics.

A comparison of the hybrid and QPU results shows that the QPU solver did not achieve complete convergence

partly because of the reduced problem dimensions as the hybrid solver reached a similar solution with the same number of nodes and bits. However, the test results presented in Fig. 12 indicate that other factors may also be involved. With 192 binary variables, the hybrid solver achieved a better solution than the QPU solver, but it was still worse than that obtained with more variables, and both presented higher ΔV costs. Because the hybrid solver only preprocessed the problem before submitting smaller instances to the QPU solver, the difference in behavior can be caused by the difficulties the QPU solver had when handling the problem as a whole.

In conclusion, the obtained results prove that the proposed transcription method is valid because it yields trajectories similar to those of traditional methods. Furthermore, QA was tested for trajectory optimization and it was found that the hybrid solver was capable of reliably solving the test problem. In contrast, the QPU solver results show that quantum annealers must evolve further before being considered for this task. However, the fast optimization time of the single QUBO problem, in the order of milliseconds, is promising.

Further development of this work can proceed in several ways. In the transcription procedure, alternatives with higher QUBO sparsity could be beneficial, and mixed *hp* methods could be considered [43]. Direct collocation methods, such as the Hermite–Simpson and Gauss–Lobatto methods [36] are also known to lead to sparser problems; however, the linearized QUBO formulation is less obvious. In the solvers and application fields, more complex problems will likely be required to surpass the linearization need, and a straightforward way to do so is to test the constrained quadratic model solver by D-Wave, which is a hybrid solver that allows the inclusion of a set of quadratic constraints in the binary optimization problem. Therefore, the dynamical constraints may possibly be expressed as a second-order expansion, increasing accuracy. Additionally, a more general QUBO formulation that avoids linearization can be provided by an arbitrary high-order expansion of the dynamics, resulting in a high-order binary polynomial that can be reduced to a QUBO formulation using *quadratization* techniques [28]. This can reduce the overall order of a binary optimization problem to the second order; however, this requires adding more binary variables.

Funding note

Open access funding provided by Università degli Studi di Roma La Sapienza within the CRUI-CARE Agreement.

Acknowledgements

This research was part of an ISCRA Type C project supported by CINECA, through which the computing time on the D-Wave Leap platform was obtained.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article. The author Christian Circi is the Associate Editor of this journal.

References

[1] Betts, J. T. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, **1998**, 21(2): 193–207.

[2] Conway, B. A. A survey of methods available for the numerical optimization of continuous dynamic systems. *Journal of Optimization Theory and Applications*, **2012**, 152(2): 271–306.

[3] Pontryagin, L. S. *Mathematical Theory of Optimal Processes*. Routledge, **2018.**

[4] Izzo, D., Märtens, M., Pan, B. F. A survey on artificial intelligence trends in spacecraft guidance dynamics and control. *Astrodynamics*, **2019**, 3(4): 287–299.

[5] De Grossi, F., Circi, C. Quantum-inspired diffusion Monte Carlo optimization algorithm applied to space trajectories and attitude maneuvers. *Advances in Space Research*, **2022**, 69(1): 592–608.

[6] De Grossi, F., Circi, C. Nonlinear model predictive control leveraging quantum-inspired optimization in the three body problem with uncertainty. *Acta Astronautica*, **2023**, 205: 68–79.

[7] Federici, L., Scorsoglio, A., Ghilardi, L., D'Ambrosio, A., Benedikter, B., Zavoli, A., Furfaro, R. Image-based meta-reinforcement learning for autonomous guidance of an asteroid impactor. *Journal of Guidance, Control, and Dynamics*, **2022**, 45(11): 2013–2028.

[8] Scorsoglio, A., D'Ambrosio, A., Ghilardi, L., Furfaro, R., Gaudet, B., Linares, R., Curti, F. Safe lunar landing via images: A reinforcement meta-learning application to autonomous hazard avoidance and landing. In: Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, Virtual, **2020**: 9–12.

[9] Jiang, J. X., Zeng, X. Y., Guzzetti, D., You, Y. Y. Path planning for asteroid hopping rovers with pre-trained deep reinforcement learning architectures. *Acta Astronautica*, **2020**, 171: 265–279.

[10] Gaudet, B., Linares, R., Furfaro, R. Six degree-of-freedom hovering over an asteroid with unknown environmental dynamics via reinforcement learning. In: Proceedings of the AIAA Scitech Forum, **2020**: AIAA 2020-0953.

[11] D'Ambrosio, A., Carbone, A., Curti, F. Optimal maneuvers around binary asteroids using particle swarm optimization and machine learning. *Journal of Spacecraft and Rockets*, **2023**, 60(5): 1458–1472.

[12] D'Ambrosio, A., Carbone, A., Mastrofini, M., Curti, F. Optimal reference orbit tracking around asteroids via particle swarm optimization and inverse dynamics technique. In: Proceedings of the 31st AAS/AIAA Space Flight Mechanics Meeting, **2021**: AAS 21-260.

[13] Nielsen, M., Chuang, I. *Quantum Computation and Quantum Information*. Cambridge University Press, **2000**.

[14] Shor, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, **1997**, 26(5): 1484–1509.

[15] Grover, L. K. A fast quantum mechanical algorithm for database search. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing, **1996**: 212–219.

[16] Harrow, A. W., Hassidim, A., Lloyd, S. Quantum algorithm for linear systems of equations. *Physical Review Letters*, **2009**, 103(15): 150502.

[17] Feynman, R. P. Simulating physics with computers. In: *Feynman and Computation*. CRC Press, **2002**.

[18] Lloyd, S. Universal quantum simulators. *Science*, **1996**, 273(5278): 1073–1078.

[19] Daley, A. J., Bloch, I., Kokail, C., Flannigan, S., Pearson, N., Troyer, M., Zoller, P. Practical quantum advantage in quantum simulation. *Nature*, **2022**, 607(7920): 667–676.

[20] Lloyd, S., De Palma, G., Gokler, C., Kiani, B., Liu, Z. W., Marvian, M., Tennie, F., Palmer, T. Quantum algorithm for nonlinear differential equations. *arXiv preprint*, **2020**, arXiv:2011.06571.

[21] Dunjko, V., Briegel, H. J. Machine learning & artificial intelligence in the quantum domain: A review of recent progress. *Reports on Progress in Physics*, **2018**, 81(7): 074001.

[22] Cerezo, M., Verdon, G., Huang, H. Y., Cincio, L., Coles, P. J. Challenges and opportunities in quantum machine learning. *Nature Computational Science*, **2022**, 2(9): 567–576.

[23] Farhi, E., Goldstone, J., Gutmann, S. A quantum approximate optimization algorithm. *arXiv preprint*, **2014**, arXiv:1411.4028.

[24] Hadfield, S., Wang, Z. H., O'Gorman, B., Rieffel, E. G., Venturelli, D., Biswas, R. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, **2019**, 12(2): 34.

[25] Farhi, E., Goldstone, J., Gutmann, S., Sipser, M. Quantum computation by adiabatic evolution. *arXiv preprint*, **2000**, arXiv:quant-ph/0001106.

[26] Kadowaki, T., Nishimori, H. Quantum annealing in the transverse Ising model. *Physical Review E*, **1998**, 58(5): 5355–5363.

[27] Peng, X. H., Liao, Z. Y., Xu, N. Y., Qin, G., Zhou, X. Y., Suter, D., Du, J. F. Quantum adiabatic algorithm for factorization and its experimental implementation. *Physical Review Letters*, **2008**, 101(22): 220405.

[28] Chang, T. H., Lux, T. C. H., Tipirneni, S. S. Least-squares solutions to polynomial systems of equations with quantum annealing. *Quantum Information Processing*, **2019**, 18(12): 374.

[29] Srivastava, S., Sundararaghavan, V. Box algorithm for the solution of differential equations on a quantum annealer. *Physical Review A*, **2019**, 99(5): 052355.

[30] Zanger, B., Mendl, C. B., Schulz, M., Schreiber, M. Quantum algorithms for solving ordinary differential equations via classical integration methods. *Quantum*, **2021**, 5: 502.

[31] Yarkoni, S., Raponi, E., Bäck, T., Schmitt, S. Quantum annealing for industry applications: Introduction and review. *Reports on Progress in Physics*, **2022**, 85(10): 104001.

[32] Smelyanskiy, V. N., Rieffel, E. G., Knysh, S. I., Williams, C. P., Johnson, M. W., Thom, M. C., Macready, W. G., Pudenz, K. L. A near-term quantum computing approach for hard computational problems in space exploration. *arXiv preprint*, **2012**, arXiv:1204.2821.

[33] Stollenwerk, T., O'Gorman, B., Venturelli, D., Mandrà, S., Rodionova, O., Ng, H., Sridhar, B., Rieffel, E. G., Biswas, R. Quantum annealing applied to de-conflicting optimal trajectories for air traffic management. *IEEE Transactions on Intelligent Transportation Systems*, **2020**, 21(1): 285–297.

[34] Stollenwerk, T., Michaud, V., Lobe, E., Picard, M., Basermann, A., Botter, T. Agile earth observation satellite scheduling with a quantum annealer. *IEEE Transactions on Aerospace and Electronic Systems*, **2021**, 57(5): 3520–3528.

[35] Elnagar, G., Kazemi, M. A., Razzaghi, M. The pseudospectral Legendre method for discretizing optimal control problems. *IEEE Transactions on Automatic Control*, **1995**, 40(10): 1793–1796.

[36] Topputo, F., Zhang, C. Survey of direct transcription for low-thrust space trajectory optimization with applications. *Abstract and Applied Analysis*, **2014**, 2014: 851720.

[37] Hofmann, C., Topputo, F. Pseudospectral convex low-thrust trajectory optimization in a high-fidelity model. In: Proceedings of the AAS/AIAA Astrodynamics Specialist Conference, **2022**: 1–19.

[38] Glover, F., Kochenberger, G., Hennig, R., Du, Y.

Quantum bridge analytics I: A tutorial on formulating and using QUBO models. *Annals of Operations Research*, **2022**, 314(1): 141–183.

[39] D-Wave Quantum Inc. First to demonstrate quantum supremacy on useful, real-world problem. **2023**. Available at https://www.dwavesys.com/

[40] Garg, D., Patterson, M. A., Hager, W. W., Rao, A. V., Benson, D. A., Huntington, G. T. An overview of three pseudospectral methods for the numerical solution of optimal control problems. *Advances in the Astronautical Sciences*, **2010**, 135: 475–487.

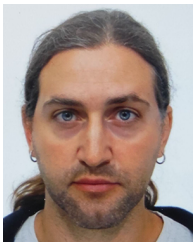
[41] McGeoch, C., Farré, P. The D-Wave advantage system: An overview. D-Wave Technical Report Series 14-1049A-A. D-Wave Systems Inc., **2020**.

[42] D-Wave Quantum Inc. Hundreds of quantum applications. **2023**. Available at https://www.dwavequantum.com/learn/featured-applications/?items=all&thirdParty=-1

[43] Darby, C. L., Hager, W. W., Rao, A. V. Direct trajectory optimization using a variable low-order adaptive pseudospectral method. *Journal of Spacecraft and Rockets*, **2011**, 48(3): 433–445.

Federico De Grossi obtained his master degree in space and astronautical engineering in 2019 at Sapienza University of Rome, and then continued his studies pursuing the Ph.D. in aeronautical and space engineering which he received in 2023 at Sapienza University of Rome, with a thesis on "Quantum-inspired meta-heuristic algorithm for trajectory optimization and spacecraft control". His research encompassed spacecraft trajectory optimization, space mission analysis, meta-heuristic optimization in conjunction with quantum computing, quantum cryptography for satellite constellations.

Andrea Carbone obtained in 2019 his master degree cum laude in space and astronautics engineering at University of Rome "La Sapienza". In 2020, he then integrated his studies by obtaining the Special Master's Degree at School of Aerospace Engineering at University of Rome "La Sapienza". Meanwhile he was three times winner of a scholarship as a junior/senior researcher at ARCALab (Automation Robotics and Control for Aerospace Lab), School of Aerospace Engineering at University of Rome "La Sapienza", for collaborating at the project SPOT "Star sensor image on-board Processing for Orbiting objects deTection", stipulated with the Italian Space Agency. The research consists in coding the on-board unit software of the SPOT for the detection of objects with stellar sensors. Later, in November 2022, he secured a scholarship for the National Ph.D. in Earth Observation and assumed the role of Ph.D. researcher at the EOSIAL ("Earth Observation and Satellite Image Applications Laboratory"), School of Aerospace Engineering, Sapienza University of Rome. Furthermore, since September 2022, he has been collaborating with ϕ-lab at the European Space Agency (ESA), situated in the ESRIN Establishment, as a visiting researcher. In this capacity, his contributions encompass the development of equivariant quantum and quantum convolutional neural networks for the analysis of remote sensing images.



Dario Spiller is a researcher affiliated with the School of Aerospace Engineering. He holds his Ph.D. degree in aerospace engineering, with a specialization in optimal control problems pertaining to trajectory and attitude maneuver planning, using innovative swarm intelligence methodologies. Following the successful completion of his doctoral studies, he embarked on a postdoctoral research program at the European Space Agency (ESA) financed by the Italian Space Agency (ASI). During his tenure at ESA, he made significant contributions to the field of remote sensing through the application of cutting-edge artificial intelligence techniques, particularly in the context of multi- and hyper-spectral imagery analysis. Notably, Dario also delved into the emerging field of quantum computing, exploring its potential applications in image analysis. Currently, Dario's research pursuits encompass several domains. He focuses on the intersection of quantum computing with machine learning and optimization problems, pushing the boundaries of computational possibilities. Additionally, his expertise extends to remote sensing, where he continues to advance the field, and on-board computing for aerospace applications. Beyond his academic and research endeavors, Dario serves as a consultant for the Food and Agriculture Organization of the United Nations. In this capacity, he leverages his expertise in machine learning technologies to make substantial contributions to crop and land cover mapping, aiding in global food security initiatives.

Daniele Ottaviani obtained his master degree cum laude in applied mathematics at University of Rome "Sapienza". He completed his studies with a II level master degree in "scientific computing", where he deepened high performance computing (HPC), and with his Ph.D. degree in "mathematics and models". Since 2013 he has worked as research fellow at the Astronomical Observatory of Rome "Monte Porzio Catone". Since 2018 he has been working in CINECA as HPC Software Developer, with

a special active monitoring role for the nascent quantum technologies. Since 2020 he is the coordinator of the CINECA Quantum Computing Lab.

Riccardo Mengoni obtained his master degree in theoretical physics at University of Camerino. He then moved to University of Verona for his Ph.D., where he worked on quantum machine learning. In summer 2018 he was selected for the USRA Quantum Academy that gave him the opportunity to be a NASA intern working on quantum annealing applications. Currently he is a quantum computing specialist at CINECA High Performance Computing (HPC) Center working on QC and its integration with HPC.

Christian Circi is currently associate professor in flight mechanics in the Department of Astronautic, Electrical and Energy Engineering at "Sapienza University of Rome". He graduated in aeronautical (B.Sc.+M.Sc.) and aerospace engineering (M.Sc.) and pursued his Ph.D. degree in aerospace engineering at Sapienza University of Rome. He worked as a researcher at the GMV (Grupo de Mecanica of Vuelo-Madrid), and is engaged in intensive consultancy activities for public companies and private industries. He is principal investigator for the Italian Space Agency project "Solar sail propulsion system". He is lecturer in "interplanetary trajectories" (since 2007), "mathematical methods for aerospace engineering" (since 2009), and "flight mechanics of launchers" (since 2016), in the master degree course of astronautical engineering at Sapienza University of Rome. He is a lecturer in the second level master course in "space transportation systems" (since 2004). His principal research fields are: orbital mechanics, third-body and solar perturbations, interplanetary and lunar trajectories, solar sail, orbits for planetary observation and ascent trajectory of launchers. C. Circi is the author of 164 scientific papers and Associate Editor for the scientific journals: *Aerospace Science and Technology*, *International Journal of Aerospace Engineering*, *Astrodynamics*, and *Space: Science & Technology*.