POLITECNICO DI TORINO
Repository ISTITUZIONALE

Learning robust satellite attitude dynamics with physics-informed normalising flow

*Original*
Learning robust satellite attitude dynamics with physics-informed normalising flow / Cena, C., Martini, M., Chiaberge, M.. - In: ACTA ASTRONAUTICA. - ISSN 0094-5765. - 245:(2026), pp. 970-981. [10.1016/j.actaastro.2026.03.047]

*Availability:*
This version is available at: 11583/3009327 since: 2026-03-30T14:26:44Z

*Publisher:*
Elsevier

*Published*
DOI:10.1016/j.actaastro.2026.03.047

*Terms of use:*

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

*Publisher copyright*

(Article begins on next page)

27 November 2026

Acta Astronautica 245 (2026) 970–981

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Research Paper

# Learning robust satellite attitude dynamics with physics-informed normalizing flow

Carlo Cena *, Mauro Martini, Marcello Chiaberge

*Department of Electronics and Telecommunications, Politecnico di Torino, Corso Duca degli Abruzzi, 24, 10129, Torino, Italy*

ARTICLE INFO

*Keywords:*
Normalizing flow
Machine learning
Physics-informed neural network
Space vehicle control
Attitude control

ABSTRACT

Attitude control is a fundamental aspect of spacecraft operations. Model Predictive Control (MPC) has emerged as a powerful strategy for these tasks, relying on accurate models of the system dynamics to optimize control actions over a prediction horizon. In scenarios where physics models are incomplete, difficult to derive, or computationally expensive, machine learning offers a flexible alternative by learning the system behavior directly from data. However, purely data-driven models often struggle with generalization and stability, especially when applied to inputs outside their training domain. To address these limitations, we investigate the benefits of incorporating Physics-Informed Neural Networks (PINNs) into the learning of spacecraft attitude dynamics, comparing their performance with that of purely data-driven approaches. Using a Real-valued Non-Volume Preserving (Real NVP) neural network architecture with a self-attention mechanism, we trained several models on simulated data generated with the Basilisk simulator. Two training strategies were considered: a purely data-driven baseline and a physics-informed variant to improve robustness and stability. Our results demonstrate that the inclusion of physics-based information significantly enhances the performance in terms of the mean relative error with the best architectures found by 27.08%. These advantages are particularly evident when the learned models are integrated into an MPC framework, where PINN-based models consistently outperform their purely data-driven counterparts in terms of control accuracy and robustness, and achieve improved settling times when compared to traditional MPC approaches, yielding improvements of up to 62%, when subject to observation noise and RWs friction.

## 1. Introduction

The effectiveness of satellites' attitude control system (ACS) is critical to the overall result of space missions, impacting both their operational efficiency and life horizon. Accurate pointing capabilities are essential for a wide range of mission objectives, including maintaining reliable communication links, gathering precise data from onboard scientific instruments, and ensuring proper thermal regulation [1]. However, designing robust ACS for satellites remains a significant challenge because of the spacecraft's complex and non-linear dynamics, as well as the highly variable conditions of the space environment. Among common attitude control actuators, reaction wheels (RWs) are preferred for their high accuracy and moderately fast maneuvers, providing continuous and smooth control [2]. However, they also introduce significant non-linearities, such as frictions and saturation effects. In addition to the complexities introduced by actuators, spacecraft in Earth orbit are continuously subjected to various external disturbance torques, such as gravity gradient, atmospheric drag, and magnetic field torques [1]. These environmental torques increase the angular momentum of the spacecraft, making active control by the ACS essential to maintain the desired orientation. Addressing these complexities has historically relied on two main approaches: model-based methods [3–6] and, more recently, data-driven techniques [7–10]. However, both exhibit some limitations. Traditional physics-based models, while foundational and offering a structured understanding of the system behavior, often rely on simplifying assumptions. These simplifications can compromise their accuracy and make them difficult to apply effectively in highly dynamic and uncertain environments [11]. A notable example of these methods is Model Predictive Control (MPC), which relies on an internal dynamics model, whose fidelity is critical for the stability and robustness of the controller. On the other hand, purely data-driven machine learning (ML) algorithms learn directly from experience without requiring an explicit system model. Although they offer versatility in pattern recognition and scalability, these approaches present significant drawbacks in safety-critical applications [12], as demonstrated by the interest in formal methods of which we provide a few Refs. [13–15]. Therefore, improving the spacecraft's dynamics

* Corresponding author.
*E-mail addresses:* carlo.cena@polito.it (C. Cena), mauro.martini@polito.it (M. Martini), marcello.chiaberge@polito.it (M. Chiaberge).

https://doi.org/10.1016/j.actaastro.2026.03.047
Received 26 October 2025; Received in revised form 22 January 2026; Accepted 26 March 2026
Available online 27 March 2026
0094-5765/© 2026 The Authors. Published by Elsevier Ltd on behalf of IAA. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

model is a key enabler for more reliable and effective attitude control, and is the central objective of the approach proposed in this work. In response to the limitations of physics-based models and ML data-driven approaches, Physics-Informed Neural Networks (PINNs) [16] have emerged as a significant paradigm shift. PINNs integrate the governing physical laws of a system directly into the neural network's learning process. Traditional PINNs learn to predict the state of the system at a particular time instant, this however does not adapt to control and planning algorithms, which only need to explore the evolution of the actuated system in a limited future time horizon. Therefore alternative approaches have emerged: some merge physical models to AI models trained with data-driven losses [12,17], others add to the data-driven loss an unsupervised physics loss term, which ensures that the physical equations are satisfied at certain points throughout the domain [18–20]. This embedding of physical knowledge has a regularization effect, which leads PINN to be more data-efficient and robust.

In this work, we explore the application of PINNs to the learning of spacecraft attitude dynamics. Specifically, we aim to learn the transition function between consecutive control steps given the current spacecraft state and the control torque applied. We compare models trained solely with data-driven loss with those trained with both data-driven and physics-informed losses. All experiments are conducted using high-fidelity simulation data generated with the Basilisk simulator [21], ensuring realistic and reproducible evaluation conditions. Our results show that PINNs offer superior performance compared to purely model-free approaches when evaluating the model as a regressor to predict the next state of the satellite's attitude dynamics and exhibit improved stability and robustness when integrated into an MPC framework, highlighting the potential of physics-informed learning in advancing autonomous space systems. We performed these experiments using two neural network architectures: a Multilayer Perceptron (MLP) and a Real-valued Non-Volume Preserving (Real NVP) model. These choices were motivated by the need to minimize inference time under the constraints of on-board hardware, as deploying more complex models remains an active research topic [22–24]. Furthermore, to increase the model's ability to capture the correlation between control inputs and resulting dynamics, we experimented with two variants of self-attention mechanisms designed to scale the predictions by a factor between 0 and 1.

### 1.1. Contribution

The key contributions of our paper are as follows.

1. We propose a novel approach for learning spacecraft attitude dynamics using a Real NVP neural network with a self-attention mechanism.
2. We introduce a physics-informed training loss to boost the generalization and robustness properties of the learned dynamics model, optimizing the data-physics losses ratio using the Lagrangian dual approach.
3. We systematically compare purely data-driven models with their physics-informed counterparts, highlighting the benefits of incorporating physical information into the training process. Showing improvements between 90.22% and 27.08% in terms of mean relative error when predicting 10 time steps in self-loop.
4. We demonstrate the practical utility of the learned models by embedding them into an MPC framework and systematically evaluating their performance and robustness-to-noise in closed-loop attitude control tasks. The comparative analysis showed a significant reduction of the spread of the trajectories, and enabling improvements of the MPC performance with respect to traditional non-linear and linear dynamics, when the spacecraft is subject to parameters estimation and state observation errors of up to 20% and 3% respectively, and RWs friction.

### 1.2. Paper organization

The paper is organized as follows. Section 2 gives an overview of the related works. Section 3 provides a detailed explanation of the proposed model with the loss functions used. Section 4 describes the dataset and the metrics used to evaluate the framework. In Section 5, we present a comprehensive discussion of the results obtained. Finally, Section 6 offers concluding remarks on the outcomes and implications of applying our framework.

## 2. Related works

This section reviews relevant works that used machine learning, both purely data-driven and with physics-informed approaches, and the integration of neural networks with MPC, for control, state estimation, and trajectory optimization, with a focus on the aerospace field.

Neural networks have been successfully applied to various aspects of control and state estimation and are being increasingly adopted for critical tasks. [19] uses a physics-informed loss to improve resilience against disturbances when estimating the state of a power-grid with a neural network, showing significant improvements in estimation accuracy and robustness under challenging conditions like three-phase faults and data manipulation attacks. [20] introduced Physics-informed Electromagnetic Field Network (PEFNet), a neural network trained to address the path loss estimation problem by using the computational electromagnetic principles in a physics-informed loss. For what concerns the aerospace sector, in [25] a physics-informed neural network was trained to model the non-linear dynamics of quadrotors, embedding conservation laws to enhance generalization and interpretability, outperforming both traditional models and black-box neural networks. [26] showed the use of neural representations for time-optimal, constant acceleration rendezvous, highlighting the ability of neural networks to learn complex, non-linear control policies. For what concerns state estimation, [17] investigated the application of PINNs for satellite state estimation during continuous thrust maneuvers, showing how the incorporation of physical laws into neural networks can enhance the accuracy and robustness of estimations in orbital mechanics. Unlike them, we apply the physics-informed approach for attitude dynamics and aim to directly learn the state transition function through a physics-informed loss, instead of a perturbation. Furthermore, [10] explored imitation learning and generative adversarial NNs for satellite attitude control under unknown perturbations using the physical simulator MuJoCo [27], suggesting a promising avenue for robust and adaptive control in uncertain space environments. Finally, [9] uses a fully-connected deep neural network trained with a data-driven approach to estimate the inertia matrix of the final system in the context of docking and berthing.

Concerning the integration of artificial intelligence approaches in MPC in the aerospace field, [28] studied an MPC framework for aerial robots that combines an offline physics-derived model with an online machine learning correction using adaptive sparse identification. [29] proposed a MPC framework that utilizes a neural network to replace the non-linear dynamics of an MPC and primal active sets to efficiently handle complex constraints and dynamics. Extending this, [30] explored PINN-based MPC for multi-link manipulators, demonstrating the potential of combining data-driven and physics-informed models for improved control performance. A more recent advancement in this area is presented by [12], who introduced a Transformer-Based MPC approach. They used the sequence modeling capabilities of transformers to generate better initial solutions to be used as starting trajectories in a MPC, and learned the terminal cost, improving runtime and convergence. Though they used a purely data-driven loss to train the neural network. These works highlight a growing trend towards using advanced machine learning techniques, particularly transformer architectures and physics-informed approaches, to overcome challenges in traditional control and estimation problems.

## 3. Methodology

In this section, we provide a detailed explanation of the proposed framework for physics-informed attitude dynamics learning. First, we formally frame the problem of satellite attitude dynamics. Then, the neural network model adopted is briefly described, and the physics-biased loss function used in the training is defined. Finally, the non-linear Model Predictive Control framework integrated with the learned dynamics is presented.

### 3.1. Satellite attitude dynamics

Here the necessary background on the attitude dynamics of a rigid spacecraft in Earth orbit is provided. A spacecraft is subject to both actuators control torques and external torques. The environmental torques acting on the satellite as disturbances can have different sources depending on its position and velocity: gravity gradient, atmospheric drag, and magnetic field torques. We are not providing a detailed formulation of these forces, though they are present in the simulator used to collect trajectories for training and testing, and they have an effect on the learned dynamics.

Given the actuators control torque in the body coordinate frame $N_c$, the satellite inertia matrix $I_s$, the reaction wheels inertia matrix $I_{rw}$, the angular velocity of the satellite $\omega$ and of the reaction wheels $\omega_{rw}$, the total external torque $N_e$, and defining the skew-symmetric matrix $S(\omega)$ in Eq. (1) we have that the dynamics of the satellite actuated by the Reaction Wheels (RWs) is shown in Eq. (2).

$$S(\omega) = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \tag{1}$$

$$\dot{\omega} = -I_s^{-1} S(\omega) I_s \omega - I_s^{-1} S(\omega) I_{rw} \omega_{rw} + I_s^{-1} N_c + I_s^{-1} N_e \tag{2}$$

Given the RWs torque $u_{rw}$ it is possible to calculate the torque acting on the spacecraft as shown in Eq. (3).

$$N_c = -u_{rw} + I_{rw} \dot{\omega} \tag{3}$$

The dynamics of the reaction wheels angular velocity is given in Eq. (4).

$$\dot{\omega}_{rw} = I_{rw}^{-1} u_{rw} - \dot{\omega} \tag{4}$$

It is possible to obtain the angular velocity of the spacecraft and of the reaction wheels by integrating over time Eqs. (2) and (4). This allows to compute the attitude of the spacecraft, defined by quaternion $q$, by integrating Eq. (5).

$$\dot{q} = \frac{1}{2} \Omega[\omega] q \tag{5}$$

where $\Omega[\omega]$ is given by Eq. (6).

$$\Omega[\omega] = \begin{bmatrix} 0 & -\omega_0 & -\omega_1 & -\omega_2 \\ \omega_0 & 0 & \omega_2 & -\omega_1 \\ \omega_1 & -\omega_2 & 0 & \omega_0 \\ \omega_2 & \omega_1 & -\omega_0 & 0 \end{bmatrix} \tag{6}$$

Finally, the total angular momentum of the spacecraft $h$ is composed of the angular momentum of the spacecraft and the angular momentum of the reaction wheels as shown in Eq. (7).

$$h = I_s \omega + I_{rw} \omega_{rw} \tag{7}$$

### 3.2. Problem formulation

Developing a reliable attitude control system for satellites remains exceptionally challenging given the spacecraft's inherently non-linear and intricate dynamic behavior, coupled with the unpredictable conditions of the space environment [1]. Enhancing the accuracy of the spacecraft dynamics model is therefore indispensable for achieving more precise attitude regulation, also adopting advanced model-based control techniques such as model predictive control (MPC). This becomes particularly important when the spacecraft state and parameters are subject to errors and estimation inaccuracies.

In this work, we propose a novel physics-informed neural network (PINN) approach to approximate the full dynamics model of the spacecraft. Besides the typical data-driven loss, a physics-based loss is designed to act as an inductive bias during the training process of the model, leading to enhanced generalization and robustness to noise compared to vanilla data-driven formulations. Fig. 1 describes the complete pipeline of the learning approach. The PINN receives the spacecraft state, comprising angular velocities of both satellite and reaction wheels, together with the angular acceleration of the satellite and the RWs control torque as input. It predicts the resulting changes in angular velocities of the satellite, to be directly integrated in a non-linear MPC framework usually adopted in advanced attitude control solutions.

### 3.3. Neural network architecture

A neural network model is used to learn the attitude dynamics of the spacecraft. We experimented with a MLP and a Real NVP architecture. For the MLP we experimented with varying number of Fully-Connected (FC) layers, between 2 and 8, and number of units per layer, between 16 and 256, while for the Real NVP architecture we search for a number of coupling layers between 2 and 6, each one composed by a number of FC layers between 2 and 4, and a number of units per layer between 16 and 64. Finally, we experimented with a Real NVP architecture and with self-attention layers, with the aim of limiting the computational cost of the inference by adopting an efficient architecture.

Real NVP belongs to the family of normalizing flow [31], aimed at modeling high-dimensional data distributions through a sequence of bijective transformations implemented with affine coupling layers, which operate by partitioning the input variables: one subset undergoes transformation via a scale and a translation neural networks, while the other remains unchanged. Both networks receive half of the input and apply their respective transformations to the complementary half, ensuring that the overall mapping remains invertible and computationally efficient. In particular, given an input vector for the Real NVP as shown in Eq. (8)

$$x = (x_a, x_b), \tag{8}$$

where $x_a$ and $x_b$ denote the partitions of the input, each affine coupling layer performs the transformation shown in Eq. (9).

$$y_a = x_a, \qquad y_b = x_b \odot \exp\big(s(x_a)\big) + t(x_a), \tag{9}$$

where $s(\cdot)$ and $t(\cdot)$ are scale and translation neural networks, respectively, and $\odot$ denotes element-wise multiplication. Given that the Real NVP network, $R$, consists of multiple coupling layers with alternating partitions, no variable remains unchanged across the full network. It should be noted that in this work the Real NVP network is composed only by coupling layers, making the transformation applied to the data deterministic. After the Real NVP model, a FC layer, $F$, is used to learn a mapping from the distribution learnt by the Real NVP model to the prediction space. The output $v \in \mathbb{R}^{3S\times 1}$ of this model is therefore computed as shown in Eq. (10).

$$v = F(R(x)) \tag{10}$$

With input $x \in \mathbb{R}^{30\times 1}$ and Real NVP output $R(x) \in \mathbb{R}^{30\times 1}$.

Self-Attention [32,33] is a mechanism that allows a model to weigh the importance of different elements within an input sequence when encoding contextual information. It captures long-range dependencies and enables dynamic representation learning by computing pairwise interactions between all tokens. In our framework, it is used to let the network focus on correlated outputs and scale correctly the prediction

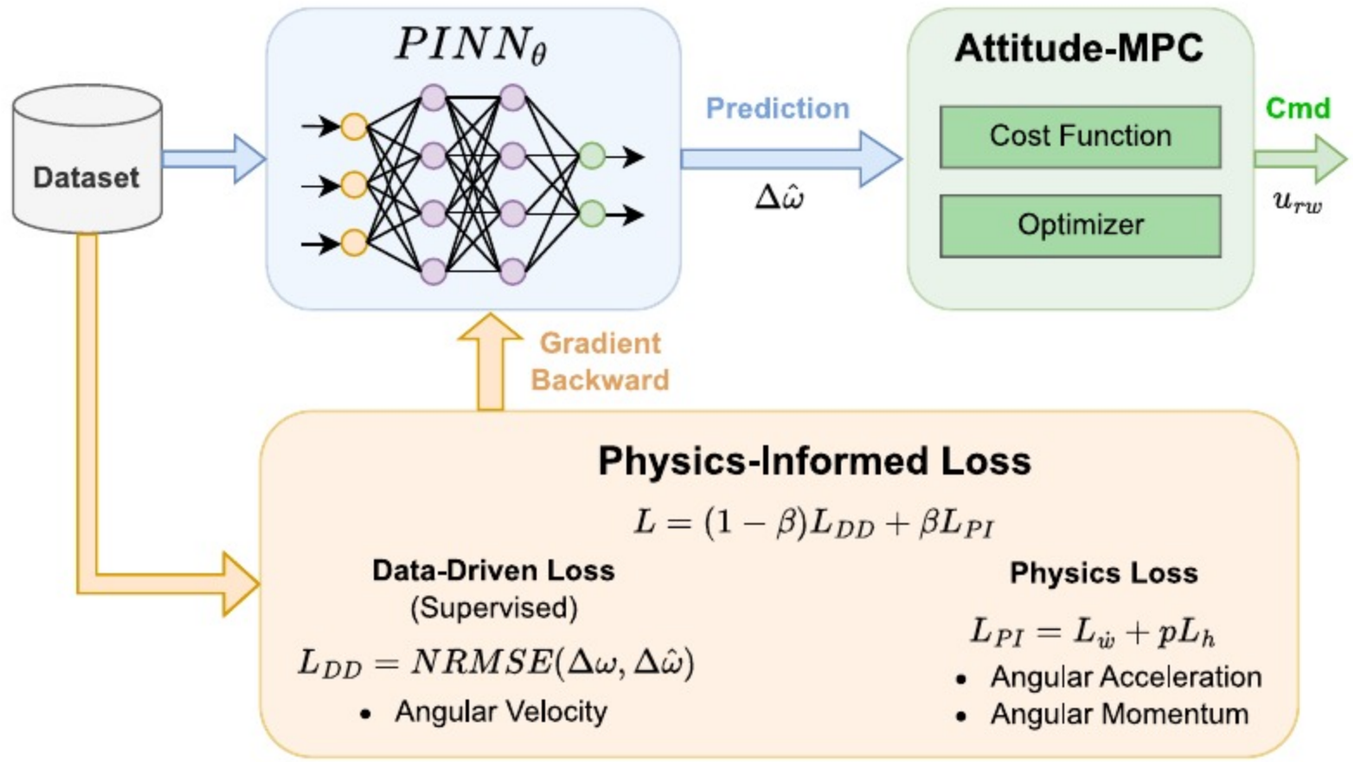



**Fig. 1.** A schematic description of the proposed PINN for satellite dynamics learning and attitude control. The PINN models, trained using a weighted sum of the data-driven and physics-informed losses, is used as dynamics model in a non-linear MPC.

when dealing with low intensity input signals. We demonstrate the benefits of this architecture in Section 5, showing that the model is more precise in predicting null changes. We implemented and experimented with two slightly different versions by changing the input of the Key and Query layers, as can be seen by looking at the differences between SA1 and SA2 in Fig. 2. In particular, in model *NVPSA1*, which uses the flow SA1, we use the output of the Real NVP model ($R(x)$) as input to compute Value, Query, and Key. Written as $a_{in}$ the input of Query and Key, we have that $a_{in} = R(x) \in \mathbb{R}^{57\times1}$ for NVPSA1. While in NVPSA2, which uses the flow SA2, the Query and Key projections receive as input the concatenation of the commanded torques, the inertia matrix of the satellite and of the RWs, and the inverse of the satellite inertia matrix. In this case $a_{in} = \{N_c, I_s, I_{rw}, I_s^{-1}\} \in R^{30\times1}$. The difference in input between the two architectures stems from the design choice to learn a weighting function that scales the predicted change in angular velocity using only the control torques and inertia matrices. This approach reduces the input dimensionality and the amount of information the network must process.

Two matrices, $W_q, W_k$, are applied to obtain respectively the Query and Key representations:

$$Q = W_q a_{in}, \qquad K = W_k a_{in} \tag{11}$$

where $W_q, W_k \in \mathbb{R}^{3S\times57}$ in the case of NVPSA1, and $W_q, W_k \in \mathbb{R}^{3S\times30}$ for NVPSA2. As value we use the output $v$ of Eq. (10).

The self-attention output $y$ is computed as

$$y = \text{sigmoid}\left(\frac{QK^\top}{\sqrt{d}}\right) v = \text{sigmoid}\left(\frac{QK^\top}{\sqrt{d}}\right) F(R(x)), \tag{12}$$

where $d$ is the dimension of the input to query and key, either 57 or 30, following [34]. The resulting output has dimension $(3S, 1)$ and represents the predicted trajectory over $S$ time steps, with three state variables per step. At time step $t$ all models aimed at predicting the resulting change in angular velocity $\Delta\hat{\omega}_{t+1}$ using as input a state composed of the current satellite angular velocity $\omega_t$, RWs velocity $\omega_{rw,t}$, the RWs commanded torque $u_{rw,t}$, and the angular acceleration of the spacecraft $\dot{\omega}_t$, estimated by computing the first-order backward difference.

### 3.4. Physics-informed training

As previously stated, we combined two loss functions. A classic data-driven loss, $L_{DD}$, shown in Eq. (13) is defined as the Normalized Root Mean Squared Error (NRMSE) of the predicted change in angular velocity $\Delta\hat{\omega}$ when compared to ground truth data obtained from the dataset. We used the standard deviation of the ground truth change in angular velocity $\sigma_{\Delta\omega}$ to normalize the Root Mean Squared Error (RMSE).

**Table 1**
Grid search results over the hyper-parameter $p$. We show the Mean Relative Error (MRE) when predicting the next $S = 10$ steps, as described in Section 4.4.

| | 1e−1 | 1e−2 | 1e−3 | 1e−4 | 0 |
|---|---|---|---|---|---|
| MRE self-loop | 1.83 | 1.72 | 2.70 | 2.37 | 1.98 |

$$L_{\text{DD}} = \frac{\sqrt{\frac{1}{B}\sum_i^B (\Delta\hat{\omega}_i - \Delta\omega_i)^2}}{\sigma_{\Delta\omega}} \tag{13}$$

A physics-informed penalty term $L_{PI}$ is designed to regularize the training of the dynamics model and avoid overfitting by embedding the physical laws of the system directly in the loss function, shown in Eq. (14). It has been derived from the satellite attitude dynamics discussed in Section 3.1, and it is composed of a weighted sum of two components: (I) the NRMSE of the predicted change in angular acceleration $L_{\dot{\omega}}$, Eq. (15), where $\dot{\omega}$ is calculated using Eq. (2); (II) the MSE of the change in angular momentum $L_h$, Eq. (16).

$$L_{PI} = L_{\dot{\omega}} + pL_h \tag{14}$$

The MSE of the change in angular momentum is weighted by a factor $p$ equal to $1e^{-2}$, found through a grid search, shown in Table 1. To limit the complexity of the physics-informed loss, the total external torque $N_e$ in Eq. (2) is set to zero, as this loss is intended to act as a regularization term that enforces general dynamical consistency. The external perturbations are indeed captured by the data-driven loss through the training data, in which such disturbances are present. This choice keeps the physics-informed loss general and robust by enforcing only intrinsic dynamical consistency, while avoiding environment-specific assumptions about external disturbances. As a trade-off, all perturbation effects are implicitly handled by the data-driven loss and under strong external torques the PI loss may hinder learning. We assume that the relative weight of the two losses will let the model learn their effect through the data-driven loss, to this end, we experimented by giving a higher weight to $L_{DD}$ rather than to $L_{PI}$.

$$L_{\dot{\omega}} = \frac{\sqrt{\frac{1}{B}\sum_i^B (\hat{\dot{\omega}} - \dot{\omega})^2}}{\sigma_{\dot{\omega}}} \tag{15}$$

$$L_h = \frac{1}{B}\sum_i^B (\| I_s \cdot (\omega + \Delta\hat{\omega}) + I_{rw} \cdot \hat{\omega}_{rw} \| - \| I_s \cdot (\omega + \Delta\omega) + I_{rw} \cdot \omega_{rw} \|)^2 \tag{16}$$

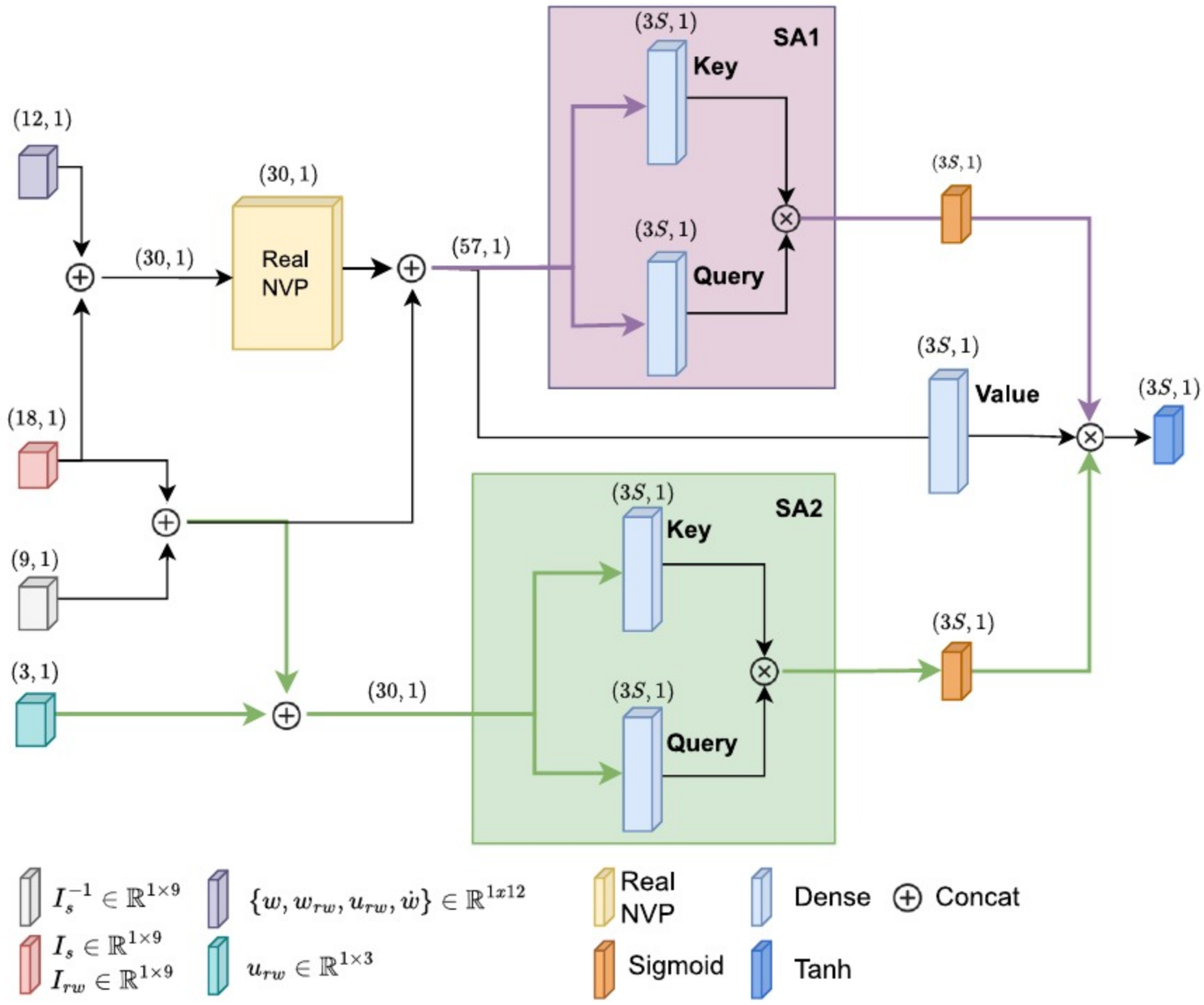



**Fig. 2.** Neural Network architecture diagrams. NVPSA1 (violet flow) uses the output of the Real NVP model to weigh the delta state predicted through a self-attention approach. NVPSA2 (green flow) instead uses the input torque and the inertia matrices of the spacecraft and of the RWs as input of the Query and Key projections. $S$ is the length of the predicted trajectory. Above each NN block we give the output shape. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In Eqs. (15) and (16), $\Delta\hat{\omega}$ and $\dot{\hat{\omega}}$ represent the predicted change in angular velocity and the predicted angular acceleration, calculated as $\dot{\hat{\omega}} = \Delta\hat{\omega}/\Delta t$. $\Delta\hat{\omega}_{rw}$ is the change in angular velocity of the RWs computed with Eq. (4) using $\dot{\hat{\omega}}$. $B$ is the batch size.

Eq. (17) shows the total loss used in our physics-informed experiments, as the weighted sum of the data-driven and physics-informed losses.

$$L = \alpha L_{DD} + \beta L_{PI} \tag{17}$$

Here $\alpha$ and $\beta$ are used to change the relative weight of the two loss functions. We experimented with fixed hand-selected values and with the Lagrangian dual approach [35], which automatically learns the best multipliers with a sub-gradient method during training, dynamically changing the relative importance of the two losses based on the performance of the model. In the latter case $\alpha = (1-\beta)$, and $\beta$ was constrained in the interval [0, 1].

### 3.5. Non-linear MPC with learned dynamics

The main focus of this work is learning the attitude dynamics of a satellite through a deep neural network, which can provide substantial benefits to diverse spacecraft control operations. We demonstrate evaluating the best models obtained in a rest-to-rest spacecraft maneuvers application, coupling the learned dynamics with a MPC framework, Fig. 3. This concrete application has also been carried out with the aim of testing the effect of the learned dynamics in a model-based controller and evaluating its robustness to state estimation errors and model uncertainties. Section 4 will describe the metrics used in the two parts of the project in more detail.

The state used by the MPC is composed of the current quaternion $q$, spacecraft angular velocity $\omega$, RWs angular velocity $\omega_{rw}$, and satellite angular acceleration $\dot{\omega}$ as $x = \{q, \omega, \omega_{rw}, \dot{\omega}\}$. The PINN is used to estimate the dynamics of the spacecraft and of the RWs starting from the current state $x_t$ and the current RWs torque $u_t$, which is then integrated to obtain the new state $x_{t+1}$, as shown in Eqs. (18)–(21).

$$\Delta\hat{\omega}_{t+1} = PINN_\theta(\omega_t, \omega_{rw,t}, \dot{\omega}_t, u_t, \theta) \tag{18}$$

$$\dot{\hat{\omega}}_{t+1} = \frac{\Delta\hat{\omega}_{t+1}}{\Delta t} \tag{19}$$

$$\dot{\hat{\omega}}_{rw,t+1} = I_{rw}^{-1} u_t - \dot{\hat{\omega}}_{t+1} \tag{20}$$

$$\dot{\hat{q}}_{t+1} = \frac{1}{2}\Omega[\omega_t]q \tag{21}$$

In particular, our PINN is used to estimate the change in angular velocity $\Delta\hat{\omega}_{t+1}$, which is subsequently used to compute the angular acceleration $\dot{\hat{\omega}}_{t+1}$ (Eq. (19)), and the RWs accelerations $\dot{\hat{\omega}}_{rw}$, by substituting the estimated angular acceleration in Eq. (4). The quaternion dynamics $\dot{\hat{q}}_{t+1}$ is calculated using the angular velocity received in input, thus it depends directly on the predicted output of the PINN starting from the second step of the MPC horizon. For subsequent steps of the MPC prediction horizon, the state propagation is performed recursively. At a generic prediction step $k \in \{1, \dots, n-1\}$, the predicted state $\hat{x}_{t+k}$ is used as input to the PINN together with the corresponding control input $u_{t+k}$:

$$\Delta\hat{\omega}_{t+k+1} = PINN_\theta(\hat{\omega}_{t+k}, \hat{\omega}_{rw,t+k}, \dot{\hat{\omega}}_{t+k}, u_{t+k}, \theta), \tag{22}$$

$$\dot{\hat{\omega}}_{t+k+1} = \frac{\Delta\hat{\omega}_{t+k+1}}{\Delta t}, \tag{23}$$

$$\dot{\hat{\omega}}_{rw,t+k+1} = I_{rw}^{-1} u_{t+k} - \dot{\hat{\omega}}_{t+k+1}, \tag{24}$$

$$\dot{\hat{q}}_{t+k+1} = \frac{1}{2}\Omega[\hat{\omega}_{t+k}]\hat{q}_{t+k}, \tag{25}$$

In this work, the recursive prediction is performed over a horizon of $n = 10$ steps. It should be noted that, even though the PINN model is trained to predict a trajectory of $S$ steps, during inference only the first predicted state is used as estimated state for the MPC.

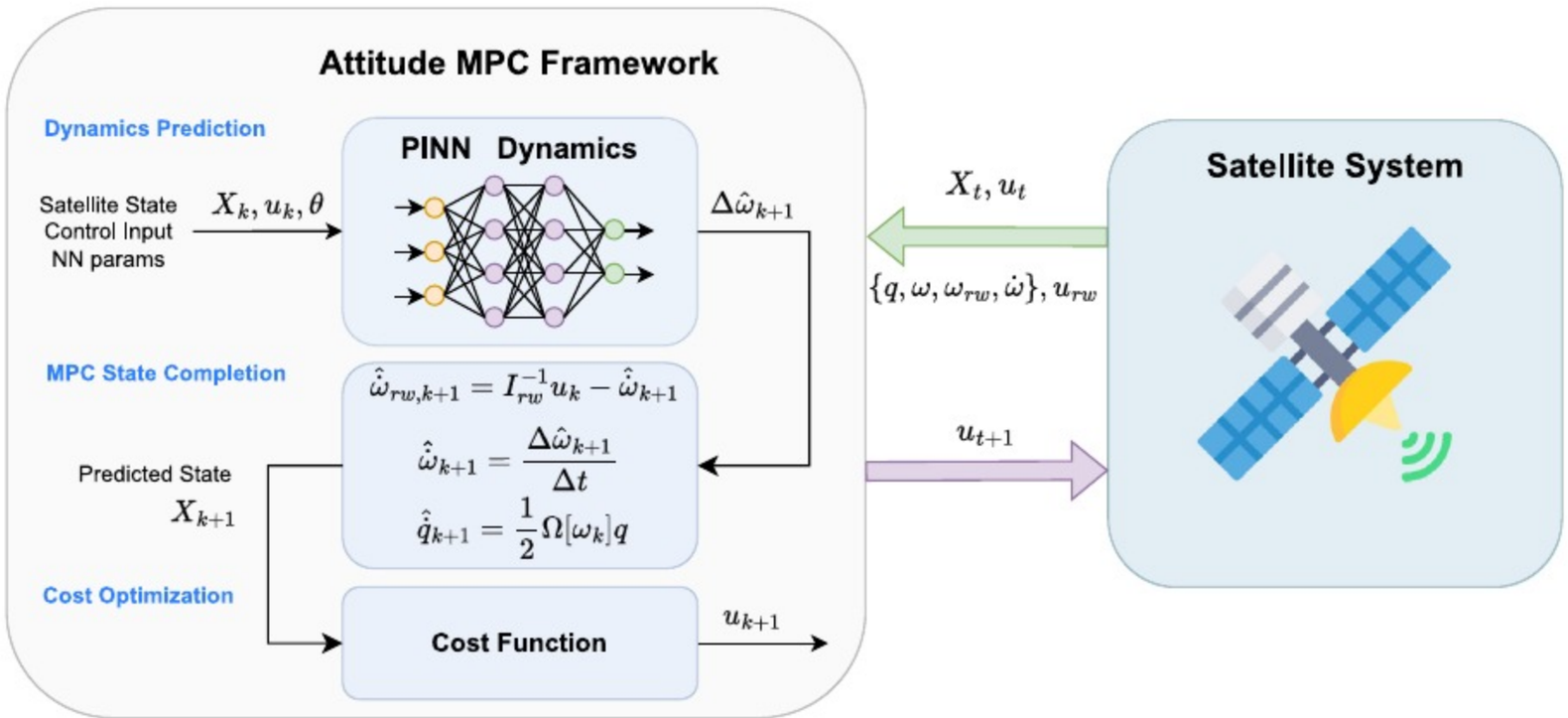



**Fig. 3.** Attitude MPC with learned dynamics, a schematic of the complete framework.

The MPC's cost function is defined in Eq. (26).

$$C = \sum_{k=0}^{n-1} \left( x_k^T Q x_k + u_k^T C u_k + \Delta u_k^T R \Delta u_k \right) + x_n^T Q x_n \tag{26}$$

where $u_k$ is an abbreviation of the commanded RWs torque $u_{rw}$ at time step $k$, and $\Delta u_k$ is the difference between the RWs torque required at the previous time step and the current one. $C$ and $R$ are diagonal cost matrices used to reduce the torques $u_{rw}$ and to limit their variation, smoothing their trajectories. Their non-zero elements are equal to $1e-1$, while $Q^{13\times13} = \text{diag}(10\,000,\ 10\,000,\ 10\,000,\ 10\,000,\ 1e^{-2}, 1e^{-2}, 1e^{-2}, 1e^{-4}, 1e^{-4}, 1e^{-4}, 1e^{-2}, 1e^{-2}, 1e^{-2})$. Finally, $n$ is the horizon of the MPC, which in our experiments is set to $n = 10$.

## 4. Experimental setups

In this section we describe in depth the dataset used, the NNs architectures, the traditional MPC implementations, the experiments and the metrics computed to evaluate the results.

### 4.1. Dataset

To train and test the models, we generated a dataset using the Basilisk simulator [21]. The simulation involved a 58 kg satellite in Low Earth Orbit actuated only by RWs which performed attitude maneuvers controlled by a MRP feedback control module available in Basilisk. In addition to the commanded torques, on the satellite acted several environmental disturbances, such as the gravity gradient, magnetic disturbances, and the atmospheric drag. The relevant satellite parameters are shown in Table 2. The rationale for this dataset design is to reproduce a representative operational environment for small satellites performing attitude control solely with RWs. By including both nominal and perturbed configurations, the dataset allows assessing not only nominal performance but also the robustness of the learned models to model uncertainties. We ran 300 runs randomly varying the initial attitude, and orbital position, to generate an initial dataset, and other 50 runs with a different satellite inertia matrix and mass, to test the robustness of the trained models to errors of approximately 10% on all axes, as these parameters are typically not known with absolute precision and may change during the operational life. These trajectories cover a broad portion of the attitude space and reaction wheel state space, providing a diverse training and testing distribution of control inputs and system responses. In each simulation, the satellite started with a random angular velocity of the reaction wheels, sampled uniformly from [−300, 300] rotations per minute. Each simulation lasted 3 min with a sampling and control time of 0.1 s, and a simulation time step of 0.001 s with Runge–Kutta 4 as integrator.

**Table 2**
We show the satellite inertia matrix $I_S$, the RWs inertia matrix $I_{rw}$, the satellite mass, the maximum torque acting on the RWs $u_{rw}$, the maximum RWs speed $\omega_{rw}$, and the controller timestep $\Delta t$.

| $I_s$ | $I_{rw}$ | Mass [kg] | Max $u_{rw}$ [N m] | Max $\omega_{rw}$ [rpm] | $\Delta t$ [s] |
|---|---|---|---|---|---|
| 5.700, 0.045, 0.002<br>0.045, 3.300, 0.012<br>0.002, 0.012, 6.100 | 0.001, 0., 0.<br>0., 0.001, 0.<br>0., 0., 0.001 | 58 | 0.05 | 6000 | 0.1 |

**Table 3**
Results obtained during the Dynamics Learning Experiments. Experiments with DD are performed using only the data-driven loss, those with LD used a Lagrangian dual approach. In bold are shown the models that will be further analyzed in the MPC and robustness-to-noise experiments.

| Experiment | MRE | Physics error | MRE self-loop | Physics error self-loop |
|---|---|---|---|---|
| RealNVP-DD | $3.3 \times 10^{-3}$ | 0.25 | 28.84 | 91.68 |
| MLP-4-16-DD | 19.67 | 24.38 | 22.63 | 137.34 |
| MLP-9-64-DD | 9.85 | 14.55 | 10.52 | 93.11 |
| NVPSA1-DD | $3.5 \times 10^{-3}$ | 0.25 | 3.07 | 6.65 |
| **NVPSA2-DD** | $\mathbf{3.2 \times 10^{-3}}$ | **0.25** | **2.40** | **5.83** |
| RealNVP-LD | $3.6 \times 10^{-3}$ | 0.19 | 2.82 | 0.84 |
| MLP-4-16-LD | 6.26 | 17.18 | 6.96 | 95.86 |
| MLP-9-64-LD | 8.26 | 10.13 | 10.02 | 65.63 |
| NVPSA1-LD | $3.2 \times 10^{-3}$ | 0.18 | 2.15 | 0.40 |
| **NVPSA2-LD** | $\mathbf{3.2 \times 10^{-3}}$ | **0.16** | **1.75** | **0.31** |

We construct the dataset by creating input–output pairs as follows: the input $x_i$ has shape (1, 12) and is defined as $x_i = \{\omega, \omega_{rw}, u_{rw}, \dot{\omega}\}$, while the target $y_i$ is equal to the change in satellite angular velocity, i.e. $y_i = \Delta\omega_i$. The training dataset was split, with random sampling, 67%–33% respectively for training and validation. This method was selected to perform training and validation on all phases of a trajectory, as the objective is to have a model that performs well irrespectively of the control phase. The length of the dataset episodes is such that potential problems linked to an under-sampling of certain control phases can be disregarded.

### 4.2. Models

In our experiments, we tested several models with varying architectures. The objectives were to obtain good performances in estimating the next attitude state given the current one and the commanded torque, and to maintain a low inference time. We experimented with a Real NVP whose scale and translation networks are composed only of FC layers, and two combinations of Real NVP with a self-attention

mechanism to decouple the predicted outputs. This architecture was selected as initial experiments showed that the Real NVP led to an improvement of 26.43% in terms of MRE and 30.59% in terms of physics-informed loss, when compared against a MLP, on the validation set for the best pair of architectures found by a grid search. Notably, for both models, the identified architecture was considerably smaller than the maximum permitted by the defined search space. Moreover, the Real NVP remains computationally efficient and fast, making it a practical choice for deployment despite its enhanced expressiveness. A comparison on the test set between these two architectures can be found in Table 3.

In addition to the number of FC and coupling layers, and units in each FC layer, we experimented with the introduction of the satellite and RWs inertia information into the network. In particular, we concatenated it to the input of the model and before the final layer. In the baseline, without self-attention mechanism, a skip connection was used to concatenate the input to the output of the Real NVP before the final FC layer. The best configuration found, shown in Fig. 2, concatenates $I_s$ and $I_{rw}$ with the input of the model and $I_s$, $I_{rw}$ and $I_s^{-1}$ to the features generated by the model before the last layer.

Finally, with the best models found, we experimented with the number of predicted steps, $S$ in Fig. 2. The model was trained to predict with a single inference all future $S$ steps, and the two losses were computed by iteratively propagating the state while keeping the torques constant. The best results were obtained with $S = 10$. During inference only the first predicted state was used to evaluate the model or as estimated state for the MPC.

### 4.3. Traditional MPC

In this work we compare our framework to two different MPC without AI, one uses a state space form (linear MPC) and another one uses a non-linear approach. Given the MPC context introduced in Section 3.5, we use the same input state and cost matrices with the non-linear dynamics, implemented following Section 3.1, and change them when using the state space form. In particular, we used as input state $x = \{q, w\}$ and all terms of matrix $Q^{6\times6}$ were set to 1000, all other parameters were left unchanged. To implement the discrete state space model, given in Eq. (27), we followed [5].

$$\dot{x}_{t+1} = A_d x_t + B_d u_t \tag{27}$$

In Eq. (27) matrix $A_d$ and matrix $B_d$ are computed from Eqs. (28) and (29) using the zero-order hold control implementation, i.e. keeping the control torque constant according to the control frequency, as shown in Eq. (31). Given $w_s$, which is the scalar angular velocity of the spacecraft about the Earth's center, and $I$, which is the satellite inertia matrix $I_s$.

$$A = \begin{bmatrix} 0^{3\times3} & 0.5I^{3\times3} \\ M^{3\times6} & \end{bmatrix} \tag{28}$$

$$B = \begin{bmatrix} 0^{3\times3} \\ \mathrm{diag}\left(\frac{1}{I_0}, \frac{1}{I_1}, \frac{1}{I_2}\right) \end{bmatrix} \tag{29}$$

where $M$ is defined in Eq. (30).

$$M = \begin{bmatrix} -8w_s^2\frac{I_1-I_2}{I_0} & 0 & 0 & 0 & 0 & w_s\frac{I_0+I_2-I_1}{I_0} \\ 0 & -6w_s^2\frac{I_0-I_2}{I_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & -2w_s^2\frac{I_1-I_0}{I_2} & -w_s\frac{I_2+I_0-I_1}{I_2} & 0 & 0 \end{bmatrix} \tag{30}$$

$$A_d = e^{A\delta t}, B_d = \int_0^{\delta t} e^{A\sigma} d\sigma \tag{31}$$

### 4.4. Evaluation metrics & experiments

To evaluate our models, we used a combination of metrics to evaluate their performance both as standalone regressors and to compare the performances of the resulting MPC controllers. These two evaluations represent the two steps that we followed during our experiments.

Firstly, we trained several neural networks with varying architectures and hyper-parameters to find the best performing framework to estimate the next attitude, given the current one and the commanded RWs torque. We will refer to this step as Dynamics Learning Experiments. The metrics used for this step are the Mean Relative Error (MRE) and the physics error, i.e. the physics-informed loss. The results were averaged over all time steps and simulations. For both metrics we will show both the results on the single-step prediction and the results when predicting the next 10 steps under the assumption of constant command torque. This was done because preliminary results showed that the models were unable to learn from the small changes associated with a single-step prediction. This will also be shown in Section 5.

Secondly, in the experiments named MPC Experiments, we used the best performing neural networks to test various non-linear MPCs evaluated with the Stability Performance Error, which is the difference between the instantaneous performance error at a given time $t$ and the error value at an earlier time $t - \Delta t$, as defined in "ECSS-E-HB-60-10A" [36]. This has been computed using $\Delta t = 10$. The performance error is defined as the difference between the desired state and the current one.

We performed 300 Monte Carlo simulations for each model to test their robustness-to-noise, adding a random error of up to 3% from a Gaussian distribution to each state variable, and a continuously distributed random error of up to 10% to the inertia matrix, and up to 20% to the satellite mass. Moreover, friction was inserted in the RWs following a linear dynamics up to $50\% \pm 12,5\%$ of the RWs maximum velocity. The initial attitude was varied randomly in the range $(\frac{\pi}{8}, \frac{\pi}{2})$. It should be noted that the neural networks were trained with data generated in simulations without noise and RWs friction. To evaluate these experiments, we will show the trajectories distribution and the stability performance error, along with the average steady-state error over the last minute of simulation and the time required to converge to a attitude error below 1 degree (Settling Time).

### 4.5. Implementation details

The experiments were performed on a computer with an Intel i7-9700K 3.60 GHz CPU, 64 GB of RAM, and a GeForce RTX 2080 Ti. The models and loss functions have been implemented in torch and we used the "do-mpc" library to implement the MPC with AI as dynamics estimator. The models have been trained on the GPU using a batch size of 16 384, found during the hyper-parameter search, and tested on CPU with an inference time of $0.66 \pm 0.03$ ms for the biggest models, i.e. the Real NVP with self-attention mechanism. Finally, the models were tested on relevant edge devices to acquire data about their computational cost. These results are shown in Section 5.3.

## 5. Experiments and results

Here we show and discuss the results obtained in the two groups of experiments. Each section presents the results obtained in the relative experiments using the metrics described in Section 4.

### 5.1. Dynamics learning experiments

We performed several experiments to identify the best architecture, and the best performing model found has 4 coupling layers, whose scale and translation network have 2 FC layers with 64 units each. This model was used to study the impact of concatenating the inertia parameters of the satellite and the RWs of the training set, respectively

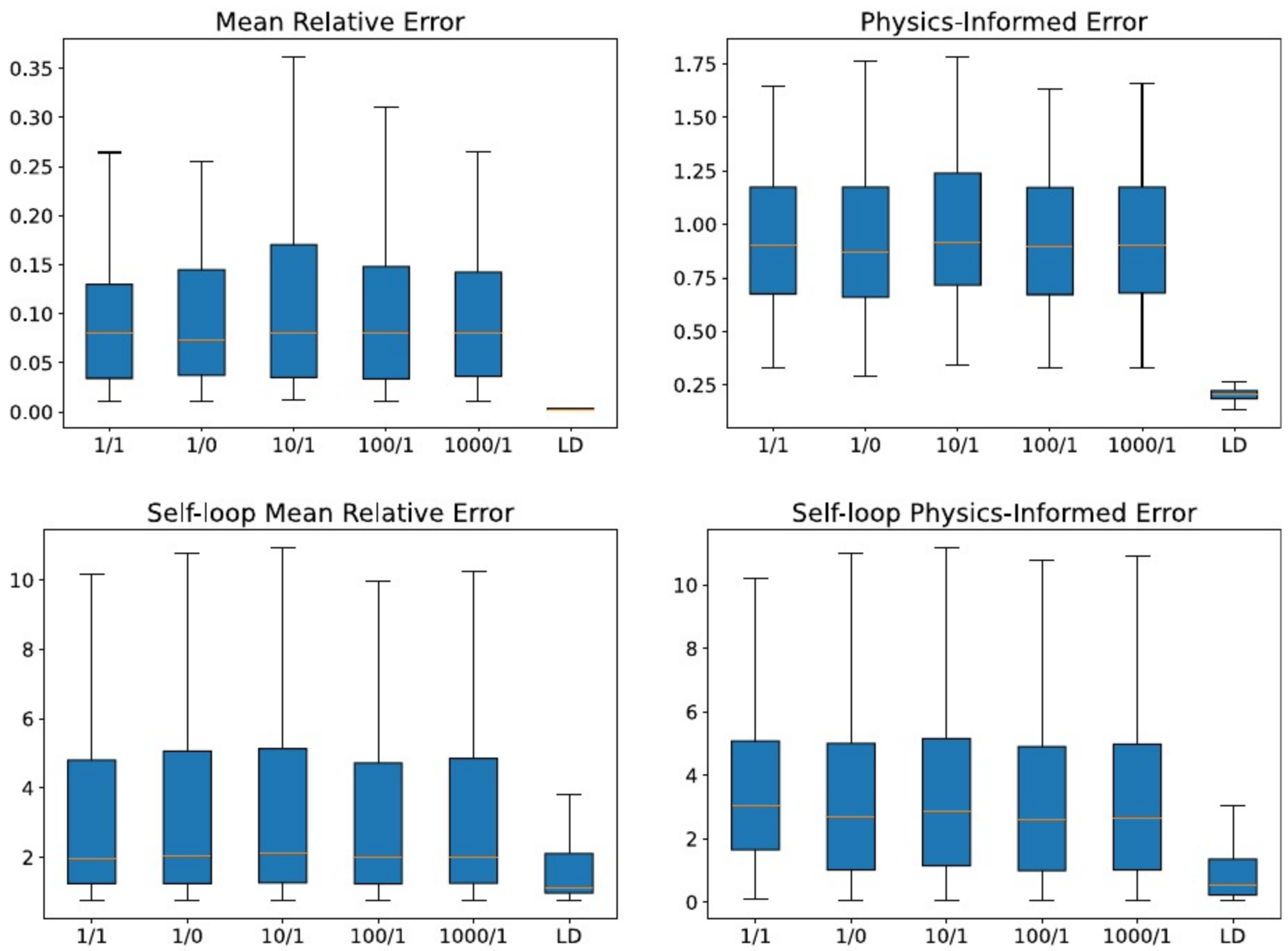



**Fig. 4.** Distribution of the results obtained during the experiments to evaluate the effect of different relative weights $\alpha$ and $\beta$ in the loss. We show the results for various ratio of fixed weights ($\alpha/\beta$) and for the Lagrangian dual approach LD.

**Table 4**
Correlation analysis between $u_{rw}$ and $\Delta\hat{\omega}$, and of $\Delta\hat{\omega}$. We show the MAE between expected and actual correlation, along with the p-values of the results. Experiments with a $p$-value above 0.05 are not significant. "p-value baseline" corresponds to the significance of the differences with respect to the models without self-attention mechanisms, while "p-value SA1" to the difference with respect to the models NVPSA1.

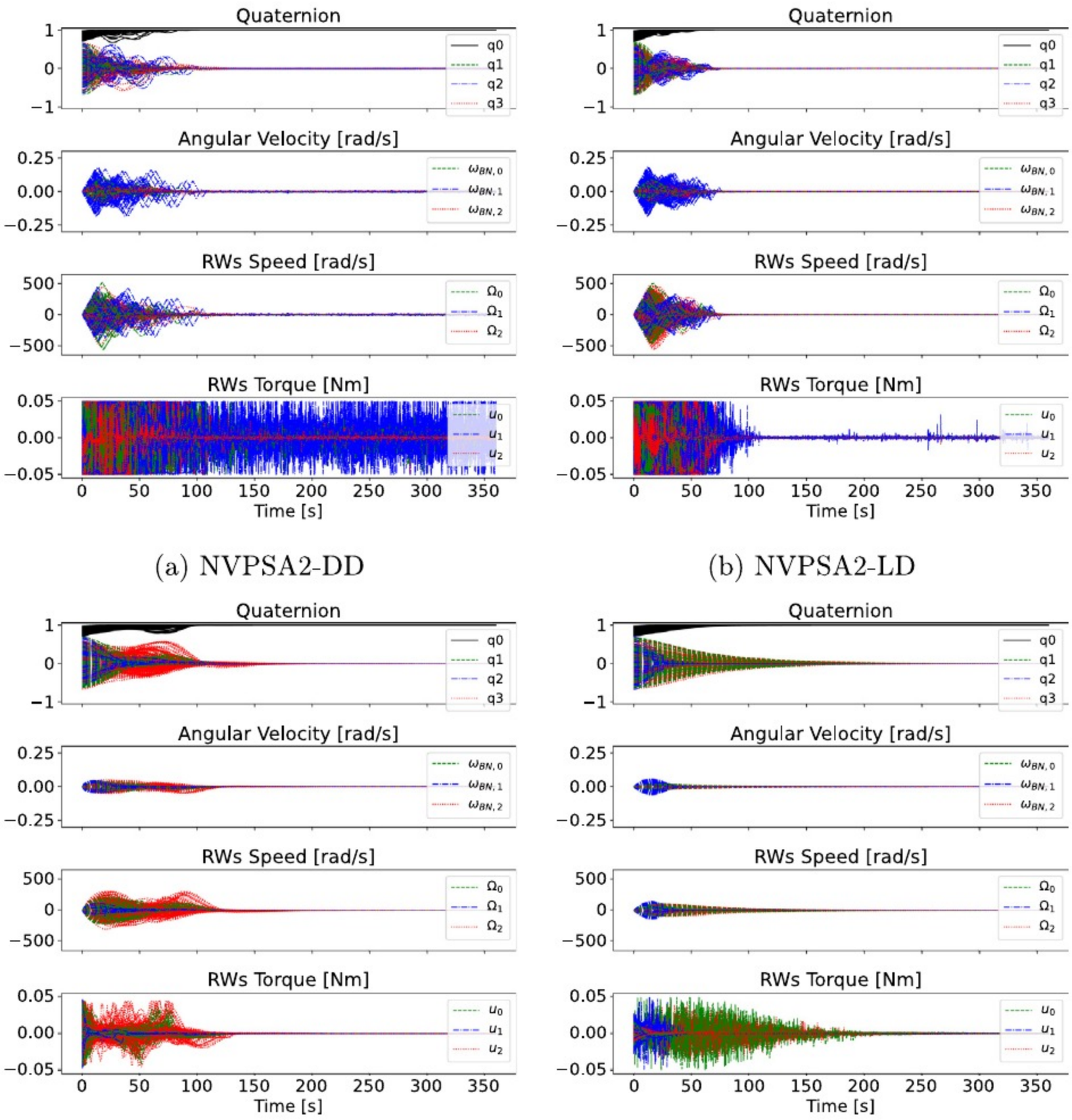
| Experiment | MAE $u_{rw}/\Delta\hat{\omega}$ | MAE $\Delta\hat{\omega}$ | p-value baseline | p-value SA1 |
|---|---|---|---|---|
| DD | 0.0435 | 0.0190 | – | – |
| NVPSA1-DD | 0.0399 | 0.0191 | 0.0162/0.6638 | – |
| NVPSA2-DD | 0.0341 | 0.0143 | 0.0005/0.0005 | 0.0417/0.0008 |
| LD | 0.0368 | 0.0207 | – | – |
| NVPSA1-LD | 0.0283 | 0.0165 | 0.0005/0.0188 | – |
| NVPSA2-LD | 0.0250 | 0.0141 | 0.0000/0.0002 | 0.0143/0.0043 |

$I_s$ and $I_{rw}$, to the input state, and the use of different pairs of weights for the data-driven and physics-informed loss, along with the use of the Lagrangian dual method [35]. The results showed that in all cases the introduction of the inertia parameters led to improved results. It should be noted that this addition helped the models even when the $I_s$ parameters were subject to an error of approximately 10%, that is, the test set.

Moreover, as shown in Fig. 4, we found that the use of the Lagrangian dual method led to improved results with respect to the use of fixed weights. Therefore, the following experiments have been performed using an input composed of the concatenation of state, $I_s$ and $I_{rw}$ from the training set, and the Lagrangian dual method to change the relative weight of the data-driven and physics-informed losses during training, starting from the same initial weight and with the constraint that the data-driven loss is assigned a higher weight, as stated in Section 3.

Subsequently, we proceeded to evaluate the performances of the models trained only with the data-driven approaches and those trained with both losses, along with the introduction of the two implementations of the self-attention mechanism introduced in Section 3. In Table 3 are shown the results obtained by these experiments. In particular, the table shows for each experiment the average of the MRE and of the physics error, both for the next step prediction and when using the model in self-loop mode for 10 time steps, i.e. when using the current output of the model as state for the input of the following inference. The experiments labeled with a DD are those in which the model has been trained using only the data-driven loss, while those with LD used the Lagrangian dual method. We also report the results obtained with the best MLP model identified during the initial grid search (*MLP-4-16*), which consists of 4 fully connected (FC) layers with 16 units each, and the results from an MLP model (*MLP-9-64*) that matches the number of FC layers used in the Real NVP model without self-attention (*RealNVP*), with each layer having the same number of units as those in the FC layers of the coupling blocks. The results show that the self-attention mechanism that, in addition to the inertia matrices, used as input for the key and query networks the commanded torque, rather than the output of the Real NVP network, *NVPSA2*, led to the best performances both when using only the data-driven loss and when using both losses. Moreover, the experiments revealed that the model trained with both losses achieved consistently better results both in terms of physics error and in terms of MRE. Both MLP models underperform *RealNVP* in all metrics, but the "MRP self-loop" when using only the data-driven loss.

To better evaluate the results and confirm their significance, in addition to the average, we analyzed the standard deviation and computed the p-values with the Wilcoxon test. The standard deviation analysis showed that the models trained only with the data-driven loss had a standard deviation higher of an order of magnitude, and that the same difference held between models trained with the same loss with and without self-attention mechanism. When considering the significance test, a $p$-value is defined as not significant when it is higher than 0.05. We obtained not significant results only in the single-step evaluation. In particular, the differences between the Physics Error of the data-driven




(a) NVPSA2-DD

(b) NVPSA2-LD

(c) Linear MPC

(d) Non-Linear MPC

**Fig. 5.** 300 MC simulation with parameter estimation errors, state estimation noise and RWs friction for NVPSA2-DD (upper left), NVPSA2-LD (upper right), traditional MPC with linear (lower left) and non-linear (lower right) dynamics.

models and of RealNVP-LD and NVPSA2-LD were not significant, nor was the difference between the MRE of the NVPSA1-LD and NVPSA2-LD. We underlined these results in Table 3. It should be noted that the MRE of NVPSA2-LD is not significant only with respect to the result found for NVPSA1-LD. We argue that this is due to the small changes in angular velocity in a single step, as the p-values obtained when comparing the predictions over 10 time steps are all close 0.

As anticipated in Section 3, we inserted the self-attention mechanisms to help the model learn the correlation between the commanded torques and the change in angular velocity, and to decouple the prediction of the models along the 3 axes. To verify its impact towards this end, we computed the correlations between the predictions and the input torques and checked whether they were similar to the correlations between expected change in angular velocity and the input torque. We performed the same experiment with the correlations of the predictions, i.e. by looking at the correlations between the angular velocity in the 3 axes. The average values, along with the Wilcoxon test results for statistical significance, are shown in Table 4. All results are considered significant as their p-values are below 0.05, and, together with those shown in Table 3, show that using self-attention it is possible to improve the performance of the models by increasing the correlation between the commanded torques and the estimated change in angular velocity.

### 5.2. MPC experiments

Given the results presented above, we used the models NVPSA2-LD and NVPSA2-DD as dynamics predictor in a MPC controller to compare the performances of the best performing model trained with and without the physics-informed loss. Comparing them with the two traditional MPC controllers introduced in Section 4.3. In Fig. 5 we present the trajectories of 300 Monte Carlo simulations when performing rest-to-rest maneuvers over 360 s with the model trained only with the data-driven loss (NVPSA2-DD), both the data-driven and the physics-informed loss (NVPSA2-LD), and the two traditional MPC controllers. In these simulations we randomly changed the orbital position, the initial satellite attitude, as well as introduce parameter estimation errors for the satellite weight (up to 20%) and for the satellite inertia matrix

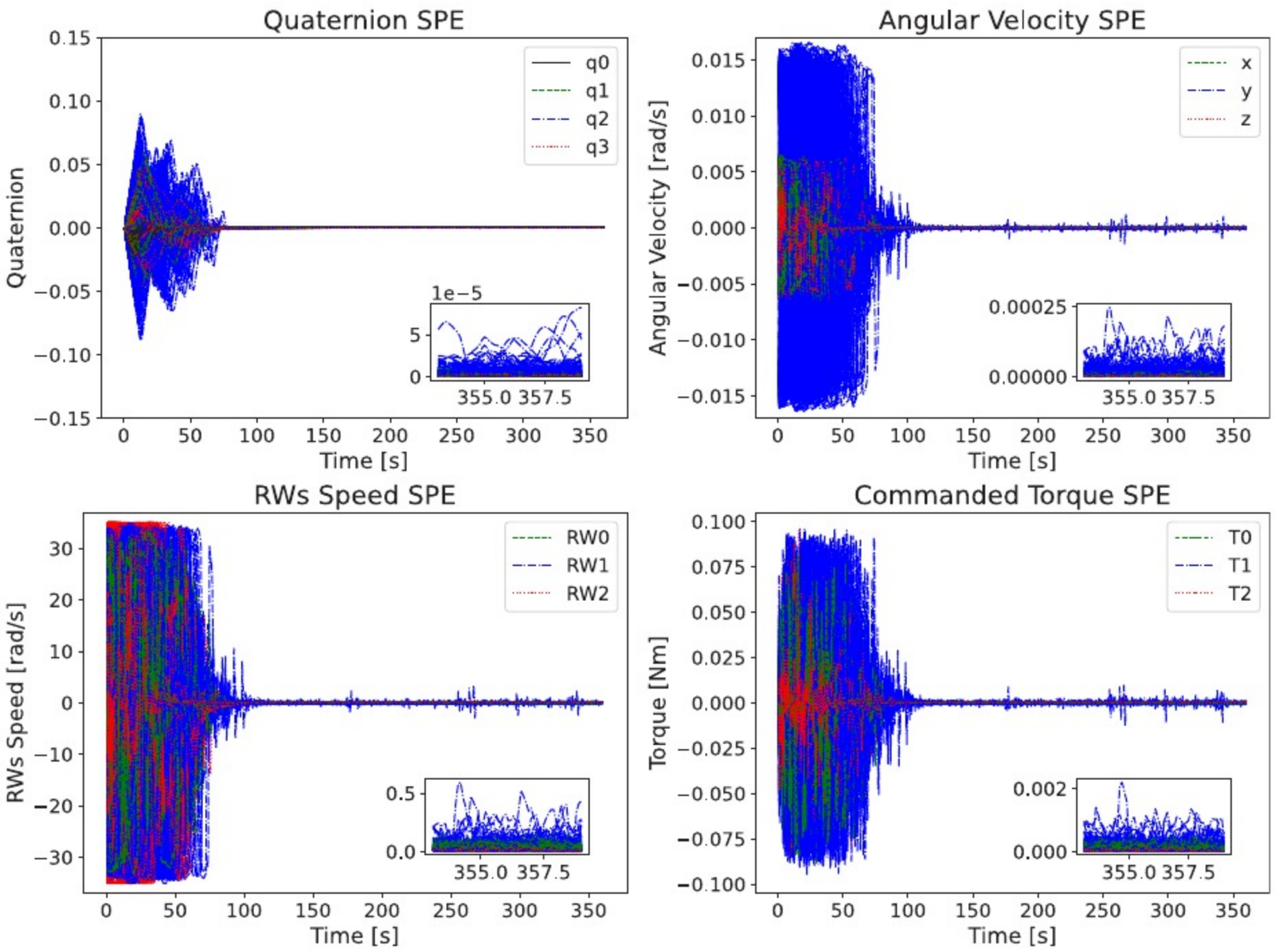



**Fig. 6.** Stability Performance Error (SPE) for the MPC with NVPSA2-LD under parameter estimation inaccuracies, RWs friction and state estimation noise.

**Table 5**
Results of the experiments performed to study the robustness-to-noise of the models when used in a MPC with several sources of noise. We show the median over 300 MC simulations.

| Experiment | Steady-state error [degrees] | Settling time [s] |
|---|---|---|
| NVPSA2-DD | 0.5865 | 65.0 |
| NVPSA2-LD | 0.0464 | **42.8** |
| Non-Linear MPC | 0.1215 | 186.2 |
| Linear MPC | **0.0024** | 114.1 |

(up to 10%). Moreover, state estimation was subject to a Gaussian random noise of up to 3% and the reaction wheels were subject to friction, as described in Section 4.4. The results show that all models are able to reach the required state with a steady-state error below 1 degree, but the model trained only with the data-driven loss is less stable. Additionally, it can be seen that the MPC with the physics-informed model (NVPSA2-LD) is able to reach a good convergence error, given the noise present in the environment, in the minimum amount of time among all controller tested. Though, it tends to produce higher torques even in steady-state. In Table 5 we show the median settling time, computed as the time required to reach a pointing error below 1 degree, and the median of the steady-state error over the last minute of the trajectory, i.e. the angle in degrees between the current and target quaternion. The results show that the MPC with state space model requires considerably more time to reach convergence, while being able to reach a more accurate attitude, with respect to the MPC with NVPSA2-LD, which is the fastest, i.e. 114 s for the state space model against the 43 s required by NVPSA2-LD. Moreover, the median settling time for the traditional non-linear MPC is approximately 3 min. Finally, Fig. 6 shows the stability performance error for the quaternion, the angular velocity of the satellite, the velocities of the RWs and torque applied to the RWs when using the physics-informed model NVPSA2-LD.

**Table 6**
Inference latency tests on different edge devices and architectures. For each device and architecture, we show the average and standard deviation of the latency, in milliseconds, over 1000 inferences.

| Device | RealNVP [ms] | NVPSA1 [ms] | NVPSA2 [ms] |
|---|---|---|---|
| AMD GX-412HC | 2.30 ± 0.63 | 2.69 ± 0.68 | 2.62 ± 0.09 |
| Raspberry PI 4B | 7.91 ± 1.55 | 8.92 ± 1.59 | 9.12 ± 1.23 |
| Jetson Nano CPU 5 W | 5.07 ± 0.28 | 6.13 ± 0.30 | 6.19 ± 0.45 |
| Jetson Nano CPU 10 W | 3.26 ± 1.62 | 3.93 ± 0.19 | 3.98 ± 0.35 |
| Jetson Nano GPU 5 W | 15.21 ± 58.26 | 19.44 ± 57.74 | 18.86 ± 61.17 |
| Jetson Nano GPU 10 W | 9.19 ± 35.77 | 10.65 ± 37.76 | 11.84 ± 71.73 |

### 5.3. Edge inference

In Table 6 we show the mean and standard deviation of the inference latency for the three neural network architectures in 3 edge devices: Jetson Nano, Raspberry PI 4B and AMD GX-412HC. In the Jetson Nano we tested 4 different combinations of maximum power consumption and target hardware, i.e. CPU or GPU. Given the small size of the model, the results suggest that it is better to perform inference only in the CPU, because the overhead associated to the data movement to the GPU is higher than the improvement in performance. When considering only the CPUs it can be seen that the bigger model (NVPSA2) could be used with a frequency between 100 Hz and 380 Hz.

## 6. Conclusions

In this work, we introduced a novel framework for learning spacecraft attitude dynamics using a Real NVP neural network augmented with a self-attention mechanism and trained via a hybrid loss that combines data-driven supervision and physics-informed penalty. The relative weight of the two loss terms has been dynamically changed during training via the Lagrangian dual approach. Our analysis demonstrated

that incorporating physical knowledge in the training as an inductive bias significantly enhances model performance, with improvements in terms of Mean Relative Error (MRE) ranging from 27.08% to 90.22% when forecasting 10 time steps. Embedding the learned models into an MPC framework further validated their practical utility, showing robust closed-loop control performance and resilience to observational noise and RWs friction. This evaluation was carried out by showing the trajectory distributions of 300 Monte Carlo simulations and by evaluating their average steady-state errors, along with the settling time, for both purely data-driven and physics-informed models, and comparing their performances with traditional MPC solutions. Achieving an improvement in terms of settling time of about 62% compared to the best performing traditional MPC. Additionally, the Stability Performance Error (SPE) was computed to evaluate the stability of the proposed controller. Across all metrics, the physics-informed model consistently achieved the best results compared to the data-driven approach, demonstrating superior robustness. Although the best results were achieved with 10-step predictions, we believe that larger networks and longer horizons could yield even greater accuracy, though at increased computational cost. We believe that the proposed framework has promising generalization capabilities and could be used to address complex non-linear dynamics in future works, such as those involved in satellite berthing or docking with non-cooperative targets, or for the control of under-actuated spacecraft. These scenarios present greater modeling and control challenges due to factors like discontinuous dynamics, partial observability, and limited actuation, which align well with the strengths of combined data-driven and physics-informed learning. Finally, we envision the integration of traditional and AI-based approaches to leverage the strengths of both, enabling faster and more accurate attitude control, even under non-nominal conditions. In the context of this work, a potential robust solution may see the integration of the MPC with physics-informed dynamics for the highly dynamic maneuvers and steady-state dynamics when close to the target attitude.

## CRediT authorship contribution statement

**Carlo Cena:** Writing – review & editing, Writing – original draft, Software, Investigation, Conceptualization. **Mauro Martini:** Writing – review & editing, Conceptualization. **Marcello Chiaberge:** Project administration, Funding acquisition.

## Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used Gemini and Copilot in order to avoid syntactical errors and improve readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work has been developed with the contribution of the Politecnico di Torino Interdepartmental Centre for Service Robotics (PIC4SeR) (https://pic4ser.polito.it) and Argotec (https://argotecgroup.com). Thanks to Manuel Pecorilla from Argotec who helped with valuable suggestions. Funding: This work was supported by the project PNRR-NGEU which has received funding from the MUR – DM 117/2023.

## References

[1] J.K. Wayer, J.-F. Castet, J.H. Saleh, Spacecraft attitude control subsystem: Reliability, multi-state analyses, and comparative failure behavior in LEO and GEO, Acta Astronaut. 85 (2013) 83–92, http://dx.doi.org/10.1016/j.actaastro.2012.12.003.

[2] X. Xia, G. Sun, K. Zhang, S. Wu, T. Wang, L. Xia, S. Liu, NanoSats/CubeSats ADCS survey, in: 2017 29th Chinese Control and Decision Conference, CCDC, 2017, pp. 5151–5158, http://dx.doi.org/10.1109/CCDC.2017.7979410.

[3] P. Iannelli, F. Angeletti, P. Gasbarri, A model predictive control for attitude stabilization and spin control of a spacecraft with a flexible rotating payload, Acta Astronaut. 199 (2022) 401–411, http://dx.doi.org/10.1016/j.actaastro.2022.07.024.

[4] J. Narkiewicz, M. Sochacki, B. Zakrzewski, Generic model of a satellite attitude control system, Int. J. Aerosp. Eng. 2020 (1) (2020) 5352019, http://dx.doi.org/10.1155/2020/5352019.

[5] M. Pecorilla, F. Stesina, Novel extended Kalman filter + $\mathbf{H}_\infty$ optimal output feedback control configuration for small satellites with high pointing stability requirements, Internat. J. Robust Nonlinear Control 34 (5) (2024) 2991–3010, http://dx.doi.org/10.1002/rnc.7119.

[6] A.G. Romero, L.C.G. de Souza, Satellite controller system based on reaction wheels using the State-Dependent Riccati Equation (SDRE) on Java, in: K.L. Cavalca, H.I. Weber (Eds.), Proceedings of the 10th International Conference on Rotor Dynamics – IFToMM, Springer International Publishing, Cham, 2019, pp. 547–561.

[7] C. Cena, S. Bucci, A. Balossino, Deep reinforcement learning for under-actuated satellite attitude control and reaction wheel desaturation using solar radiation pressure, in: Proceedings of the International Astronautical Congress, IAC, 2023.

[8] W. Retagne, J. Dauer, G. Waxenegger-Wilfing, Adaptive satellite attitude control for varying masses using deep reinforcement learning, Front. Robot. AI Volume 11 - 2024 (2024) http://dx.doi.org/10.3389/frobt.2024.1402846.

[9] S. Wu, W. Chu, Z. Wu, W. Chen, W. Wang, Inertia matrix identification of combined spacecraft using a deep neural network with optimized network structure, Adv. Space Res. 73 (3) (2024) 1979–1991, http://dx.doi.org/10.1016/j.asr.2023.11.024.

[10] Z. Zhang, H. Peng, X. Bai, Imitation learning for satellite attitude control under unknown perturbations, 2025, http://dx.doi.org/10.48550/arXiv.2507.01161, Pre-print techrxiv.

[11] M. Moldabekov, A. Sukhenko, Y. Orazaly, A. Aden, Dynamics analysis of a non-linear satellite attitude control system using an exact linear model, Mathematics 11 (12) (2023) http://dx.doi.org/10.3390/math11122614.

[12] D. Celestini, D. Gammelli, T. Guffanti, S. D'Amico, E. Capello, M. Pavone, Transformer-Based model predictive control: Trajectory optimization via sequence modeling, IEEE Robot. Autom. Lett. 9 (11) (2024) 9820–9827, http://dx.doi.org/10.1109/LRA.2024.3466069.

[13] A. Baninajjar, A. Rezine, A. Aminifar, VNN: verification-friendly neural networks with hard robustness guarantees, in: Proceedings of the 41st International Conference on Machine Learning, ICML '24, JMLR.org, 2024.

[14] M. Zhang, Safeguarding neural network-controlled systems via formal methods: From Safety-by-Design to runtime assurance (Invited Talk), in: P. Rümmer, Z. Wu (Eds.), Theoretical Aspects of Software Engineering, Springer Nature Switzerland, Cham, 2026, pp. 3–10.

[15] A. Pereira, C. Thomas, Challenges of machine learning applied to Safety-Critical Cyber-Physical systems, Mach. Learn. Knowl. Extr. 2 (4) (2020) 579–602, http://dx.doi.org/10.3390/make2040031, URL https://www.mdpi.com/2504-4990/2/4/31.

[16] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707, http://dx.doi.org/10.1016/j.jcp.2018.10.045.

[17] J. Varey, J.D. Ruprecht, M. Tierney, R. Sullenberger, Physics-Informed neural networks for satellite state estimation, in: 2024 IEEE Aerospace Conference, 2024, pp. 1–8, http://dx.doi.org/10.1109/AERO58975.2024.10521414.

[18] E.A. Antonelo, E. Camponogara, L.O. Seman, J.P. Jordanou, E.R. de Souza, J.F. Hübner, Physics-informed neural nets for control of dynamical systems, Neurocomputing 579 (2024) 127419, http://dx.doi.org/10.1016/j.neucom.2024.127419.

[19] S. Falas, M. Asprou, C. Konstntinou, M.K. Michael, Robust power system state estimation using physics-informed neural networks, IEEE Trans. Ind. Inform. pre-print (2025).

[20] F. Jiang, T. Li, X. Lv, H. Rui, D. Jin, Physics-Informed neural networks for path loss estimation by solving electromagnetic integral equations, IEEE Trans. Wirel. Commun. 23 (10) (2024) 15380–15393, http://dx.doi.org/10.1109/TWC.2024.3429196.

[21] P.W. Kenneally, S. Piggott, H. Schaub, Basilisk: A flexible, scalable and modular astrodynamics simulation framework, J. Aerosp. Inf. Syst. 17 (9) (2020) 496–507, http://dx.doi.org/10.2514/1.I010762.

[22] G.N. Kiprit, A. Koch, M. Petry, M. Werner, Graph neural networks for anomaly detection in spacecraft, in: Proceedings of the 1st SPAICE Conference on AI in and for Space, Zenodo, 2024, pp. 105–109, http://dx.doi.org/10.5281/zenodo.13885527.

[23] C. Cena, U. Albertin, S. Bucci, M. Martini, A. Balossino, M. Chiaberge, Self-supervised Real NVP for on-board satellite fault detection, Acta Astronaut. 239 (2026) 49–60, http://dx.doi.org/10.1016/j.actaastro.2025.10.068.

[24] J. Lee, I. Bahk, H. Kim, S. Jeong, S. Lee, D. Min, An autonomous parallelization of transformer model inference on heterogeneous edge devices, in: Proceedings of the 38th ACM International Conference on Supercomputing, ICS '24, Association for Computing Machinery, New York, NY, USA, 2024, pp. 50–61, http://dx.doi.org/10.1145/3650200.3656628.

[25] W. Gu, S. Primatesta, A. Rizzo, Physics-informed neural network for quadrotor dynamical modeling, Robot. Auton. Syst. 171 (2024) 104569, http://dx.doi.org/10.1016/j.robot.2023.104569.

[26] D. Izzo, S. Origer, Neural representation of a time optimal, constant acceleration rendezvous, Acta Astronaut. 204 (2023) 510–517, http://dx.doi.org/10.1016/j.actaastro.2022.08.045.

[27] E. Todorov, T. Erez, Y. Tassa, MuJoCo: A physics engine for model-based control, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 5026–5033, http://dx.doi.org/10.1109/IROS.2012.6386109.

[28] T. Manzoor, H. Pei, Z. Sun, Z. Cheng, Model predictive control technique for ducted fan aerial vehicles using Physics-Informed machine learning, Drones 7 (1) (2023) http://dx.doi.org/10.3390/drones7010004.

[29] S.W. Chen, T. Wang, N. Atanasov, V. Kumar, M. Morari, Large scale model predictive control with neural networks and primal active sets, Automatica 135 (2022) 109947, http://dx.doi.org/10.1016/j.automatica.2021.109947.

[30] J. Nicodemus, J. Kneifl, J. Fehr, B. Unger, Physics-informed neural networks-based model predictive control for multi-link manipulators, IFAC-PapersOnLine 55 (20) (2022) 331–336, http://dx.doi.org/10.1016/j.ifacol.2022.09.117, 10th Vienna International Conference on Mathematical Modelling MATHMOD 2022.

[31] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real NVP, in: 5th International Conference on Learning Representations, ICLR, OpenReview.net, 2017.

[32] J. Cheng, L. Dong, M. Lapata, Long Short-Term Memory-Networks for machine reading, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2016, pp. 551–561, http://dx.doi.org/10.18653/v1/D16-1053.

[33] Z. Lin, M. Feng, C.N. dos Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured Self-Attentive sentence embedding, in: 5th International Conference on Learning Representations, ICLR, OpenReview.net, 2017.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS '17, Curran Associates Inc., Red Hook, NY, USA, 2017, pp. 6000–6010.

[35] F. Fioretto, P. Van Hentenryck, T.W.K. Mak, C. Tran, F. Baldo, M. Lombardi, Lagrangian duality for constrained deep learning, in: Y. Dong, G. Ifrim, D. Mladenić, C. Saunders, S. Van Hoecke (Eds.), Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track, Springer International Publishing, Cham, 2021, pp. 118–135.

[36] European Cooperation for Space Standardization, ECSS-E-HB-60-10a - Control performance guidelines, 2010.