COSMIC: fast closed-form identification from large-scale data for LTV systems leading to optimal spacecraft attitude control

Maria Carvalho
mariaccarvalho@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2021

Abstract

The present work introduces a closed-form method for identification of discrete-time linear time-variant systems from data. We formulate the learning problem as a regularized least squares problem where the regularizer favors smooth solutions within a trajectory. Further, we develop a closed-form algorithm with guarantees of optimality and a complexity that varies linearly with the number of instants considered per trajectory. The COSMIC algorithm achieves the desired result even in the presence of large volumes of data, too large for general purpose solvers and for a specially designed coordinate descent method to reach a valid solution. To prove its applicability to real world systems, we start by performing the validation in synthetic spring-mass-damper systems and guarantee that the estimated system model can be used to find the optimal control path for such systems. Our algorithm was implemented in a Low Fidelity Simulator for a simplified version of the Comet Interceptor mission from European Space Agency, that requires precise pointing of the on-board cameras in a fast dynamics environment. Thus, we conclude that this thesis provides a new and better approach to classical system identification techniques for linear time-variant systems, while proving to be a solid base for applications in the Space industry and a step forward to the incorporation of algorithms that leverage data in such a safety-critical scientific environment.

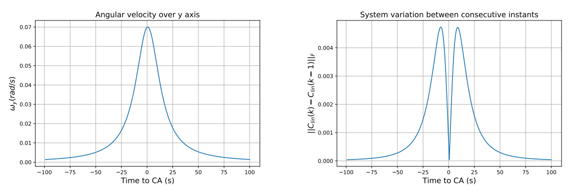
Keywords: Closed form; system identification; linear time variant; Space

1. Introduction

System modeling represents a significant cost in complex engineering projects, sometimes up to 50% of the total cost. Thus, it is essential to create practical system identification tools that adapt to a wide range of problems and achieve a solution in a time-constrained setting. Such tools can be especially useful in a Space mission project that represents huge cost efforts for entire countries and agencies [12]. The integration of Machine Learning (ML) and Guidance, Navigation and Control (GNC) can be helpful in these set of problems, from building robust control frameworks that address parameter varying systems to applying verification and validation techniques to a system in a more efficient way.

The Comet Interceptor mission aims to explore a comet entering the Solar System for the first time by obtaining precise images of the celestial body that will allow its chemical characterization, the assessment of its structure and the interaction of its components [7].

We are going to analyze a mission profile that regards the pointing as a function of the attitude of the spacecraft, without the aid of external pointing correction structures. Thus, the pointing requirement becomes an attitude control problem, where we wish to achieve a spacecraft orientation in relation to the comet that allows it to be in the main camera line of sight constantly. This requirement gets harder to meet the closer the spacecraft gets to the target, as the dynamics become faster and the actuation needed to maintain the attitude increases, as shown in Figure 1(a). Moreover, it can be observed that the variation of the system between consecutive instants, in a discrete-time setting, is constrained and does not vary indefinitely, which is clear in Figure 1(b). For controller design purposes, nonlinear systems as the attitude of the spacecraft can benefit from being modeled as linear time-variant (LTV), which is how we look at the problem.



(a) Angular velocity evolution. (b) Variation between instants.

Figure 1: Dynamics throughout the trajectory.

Given what is presented above, the present work aims to address the shortcomings of classical system identification techniques for LTV systems in order to create models that are simultaneously accurate enough to represent a wide range of phenomena while being simple enough to use with well-known and widely used control techniques, leading to a more robust control algorithm. The main challenges of this proposal are the amount of data needed to characterize one trajectory due to its changing dynamics and guaranteeing a solution for the problem in a finite-time setting.

1.1. Related work

A large body of work has been developed in order to incorporate information provided by measurements in system identification and controller synthesis. The results described in [14] establish the grounds for system identification from observed data. Earlier, the work of Kumpati et al. [11] already proposed a system identification strategy based on Neural Networks. Nevertheless, data availability and computational and algorithmic tools were limited, so performance and speed were constrained.

Recently, there has been more focus on providing guarantees that a problem can be solved to a given precision in finite time, which is crucial to bridge the gap to optimal control. A lot of work has been developed regarding linear time-invariant (LTI) systems [9], [2], [15]. Although these are significant achievements, not many real world systems can be represented as LTI and generalizing these findings to LTV systems is essential for a broader application of the results.

The literature is not as extensive regarding the direct application of data-driven system identification to linear time-variant systems. However, this problem has been a concern for many years. Dudul et al [6] apply Feedforward Neural Networks to identify an LTV system, assuming a transfer function characterization. Lin et al. [13] proposes an episodic block model for an LTV system, where parameters within a block are kept constant, followed by the exploration of a meta-learning approach for system identification divided into two steps: an offline initialization process and online adaptation. Formetin et al [8] have recently proposed a system identification procedure that takes control specifications into account in the form of regularization.

When presented with a nonlinear model, one can decide to go forward to identify the best linear approximation if it is taken into account the error associated with this assumption. We can choose to implement a linear system identification for nonlinear systems due to its simplicity and reasonably good approximation for many applications or even just as an initial estimation [16] [17]. When choosing how to proceed with the system identification, we must recognize the trade-off between the versatility of nonlinear models and the simplicity of linear ones. It is important to refer that LTV systems, in particular discrete-time, are a powerful class of models and perform well when used to approximate nonlinear dynamics and can be very useful for controller design and analysis, as we propose to do in this thesis [4].

2. Problem Statement

Let us address a discrete linear time-variant system defined as

$$\mathrm{x}(k+1) = \mathrm{A}(k)\mathrm{x}(k) + \mathrm{B}(k)\mathrm{u}(k), \tag{1}$$

such that $k \in [0,...,N-2]$, with N being the total number of instants considered to be part of one trajectory. In a given instant k, the state is $\mathrm{x}(k) \in \mathbb{R}^p$ and the control input is $\mathrm{u}(k) \in \mathbb{R}^q$. System parameters $\mathrm{A}(k) \in \mathbb{R}^{p\times p}$ and $\mathrm{B}(k) \in \mathbb{R}^{p\times q}$ are, respectively, the dynamics and control matrices that define the system's response and the unknown variables we aim to derive from the data.

Taking into account all the data available, we need to additionally define the matrices that contain the state information, $\mathrm{X}(k) \in \mathbb{R}^{p\times L}$ as $\mathrm{X}(k) = \begin{bmatrix}\mathrm{x}_1(k) & \mathrm{x}_2(k) & ... & \mathrm{x}_L(k)\end{bmatrix}$ and the control information, $\mathrm{U}(k) \in \mathbb{R}^{q\times L}$ as $\mathrm{U}(k) = \begin{bmatrix}\mathrm{u}_1(k) & \mathrm{u}_2(k) & ... & \mathrm{u}_L(k)\end{bmatrix}$, of all the L different simulations for the k-th instant of the trajectory. Moreover, we define $\mathrm{X}'(k)$ as $\mathrm{X}'(k) = \mathrm{X}(k+1)$.

To find the proper solution for system (1), we start by defining the optimization variable $\mathrm{C}(k) \in \mathbb{R}^{(p+q)\times p}$ as $\mathrm{C}(k) = \begin{bmatrix}\mathrm{A}^T(k) \\ \mathrm{B}^T(k)\end{bmatrix}$, with $\mathrm{C} = [\mathrm{C}(k)]_{k=0}^{k=N-2}$, such that $\mathrm{C} \in \mathbb{R}^{(N-1)(p+q)\times p}$. Besides, to allow for a least squares representation of the identification problem, we need to define $\mathrm{D}(k) \in \mathbb{R}^{L\times(p+q)}$ as $\mathrm{D}(k) = \begin{bmatrix}\mathrm{X}^T(k) & \mathrm{U}^T(k)\end{bmatrix}$, with $\mathrm{D} = [\mathrm{D}(k)]_{k=0}^{k=N-2}$, following $\mathrm{V} = diag(\mathrm{D}(k))$ with $\mathrm{V} \in \mathbb{R}^{(N-1)L\times(N-1)(p+q)}$.

To limit the variation of the system between instants, thus encoding the domain knowledge that a system will not change drastically from one time step to the next, as addressed in the motivation of this work, we add a term to the cost function, with regularization parameters $\lambda_k > 0$. By allowing for the λ_k to vary throughout the trajectory, we impose more flexibility to the problem formulation, covering a wider range of scenarios.

As such, we are solving

Problem 1.

$$\underset{\mathrm{C}}{minimize} \quad f(\mathrm{C}) := \frac{1}{2}\|\mathrm{VC} - \mathrm{X}'\|_F^2 + \frac{1}{2}\sum_{k=1}^{N-2} \|\lambda_k^{1/2}(\mathrm{C}(k) - \mathrm{C}(k-1))\|_F^2. \tag{2}$$

Remark 1. *The problem we are solving can be seen as a trade off between how close the optimization variable is to the data and how much we allow for it to change between instants. Thus, we can address the tuning of parameter λ_k as the tuning of the relative importance of each objective. Higher values of λ_k are congruent with little variation of the system behavior between instants and will emphasize the weight of the second term has in* *the cost function and lower values will allow for more drastic changes between k and $k+1$.*

Moreover, to aid the analysis, we can define the second term of f writing all the weighted difference equations $\lambda_k^{1/2}\left(\mathrm{C}(k)-\mathrm{C}(k-1)\right)$ as a matrix product

$$\underbrace{\begin{bmatrix}\lambda_1^{1/2} & 0 & \dots & 0 & 0\\ 0 & \lambda_2^{1/2} & \dots & 0 & 0\\ \dots & \dots & \dots & \dots & \dots\\ 0 & 0 & \dots & \lambda_{N-3}^{1/2} & 0\\ 0 & 0 & \dots & 0 & \lambda_{N-2}^{1/2}\end{bmatrix}}_{\Upsilon^{1/2}}\underbrace{\begin{bmatrix}-\mathrm{I} & \mathrm{I} & 0 & \dots & 0 & 0 & 0\\ 0 & -\mathrm{I} & \mathrm{I} & \dots & 0 & 0 & 0\\ \dots & \dots & \dots & \dots & \dots & \dots & \dots\\ 0 & 0 & 0 & \dots & -\mathrm{I} & \mathrm{I} & 0\\ 0 & 0 & 0 & \dots & 0 & -\mathrm{I} & \mathrm{I}\end{bmatrix}}_{\mathrm{F}}\begin{bmatrix}\mathrm{C}(0)\\ \mathrm{C}(1)\\ \dots\\ \mathrm{C}(N-2)\end{bmatrix}, \tag{3}$$

allowing Problem 1 to be written as

Problem 2.

$$\underset{C}{\textit{minimize}} \quad f(\mathrm{C}) := \frac{1}{2}\|\mathrm{VC}-\mathrm{X}'\|_F^2 + \frac{1}{2}\|\Upsilon^{1/2}\mathrm{FC}\|_F^2. \tag{4}$$

This statement makes explicit that we are solving an unconstrained convex optimization problem. As Problem 2 is an unconstrained minimization of a quadratic function, a solution is proven to exist and is well-defined [1]. Solving problem (4) amounts to solving the equation

$$\nabla f(\mathrm{C}) = 0. \tag{5}$$

The following linear equation solves the problem,

$$\begin{aligned}\nabla f(\mathrm{C}) &= \mathrm{V}^T(\mathrm{VC}^* - \mathrm{X}') + \mathrm{F}^T\Upsilon\mathrm{FC}^* = 0\\ &\Leftrightarrow \left(\mathrm{V}^T\mathrm{V} + \mathrm{F}^T\Upsilon\mathrm{F}\right)\mathrm{C}^* = \mathrm{V}^T\mathrm{X}'\\ &\Leftrightarrow \mathrm{C}^* = \left(\mathrm{V}^T\mathrm{V} + \mathrm{F}^T\Upsilon\mathrm{F}\right)^{-1}\mathrm{V}^T\mathrm{X}'\end{aligned} \quad . \tag{6}$$

We now derive a condition on data that guarantees that the information collected is enough to reach a solution and correctly identify the system.

Theorem 1 (***Informal***: When is the dataset large enough?)**.** *When the collected data is sufficiently varied, there is a unique solution to Problem 1. Further, after each collected trajectory, it is possible to identify if there is enough information for attaining a unique solution by computing the sum of the required trajectories covariances and testing it for positive definiteness.*

To prove Theorem 1, we will require two lemmas.

Lemma 1. *Let* $\mathrm{B} \succeq 0$, $\mathrm{B} \in \mathbb{R}^{m\times m}$ *and* $\mathrm{x} \neq 0$. *If* $\mathrm{x} \in$ *Im*(B) *then* $\mathrm{x}^T\mathrm{Bx} > 0$.

Proof. If the result of the Lemma does not hold, we would have that $\mathrm{x} \in \mathrm{Im}(\mathrm{B})$ and $\mathrm{x} \in \mathrm{Ker}(\mathrm{B})$. However, since $\mathrm{Im}(\mathrm{B}) \cap \mathrm{Ker}(\mathrm{B}) = \emptyset$, the only option would be to have $\mathrm{x} = 0$, which violates the conditions of the Lemma, thus proving that the inference holds. □

Lemma 2. *The following two expressions are equivalent: (i)* $\mathrm{A} \succeq 0$, $\mathrm{B} \succeq 0$, $\forall \mathrm{v} \in$ *Ker*$(\mathrm{B}) \setminus \{0\}$ $\quad \mathrm{v}^T\mathrm{Av} > 0$*; and (ii)* $\mathrm{A}+\mathrm{B} \succ 0$.

Proof. The proof that (ii) implies (i) follows trivially from the fact that both A and B are positive semidefinite.

To prove the other way around, we first remember that if $\mathrm{B} \succeq 0$, then $\mathrm{Im}(\mathrm{B}) \perp \mathrm{Ker}(\mathrm{B})$.

Let $\mathrm{v} = \mathrm{v_I} + \mathrm{v}_K$, $\mathrm{x} \neq 0$, with $\mathrm{v_I} \in \mathrm{Im}(\mathrm{B})$ and $\mathrm{v_K} \in \mathrm{Ker}(\mathrm{B})$.

If $\mathrm{v_I} \neq 0$, then

$$\begin{aligned}(\mathrm{v_I}+\mathrm{v}_K)^T(\mathrm{A}+\mathrm{B})(\mathrm{v_I}+\mathrm{v}_K) =\\ (\mathrm{v_I}+\mathrm{v}_K)^T\mathrm{A}(\mathrm{v_I}+\mathrm{v}_K) + (\mathrm{v_I}+\mathrm{v}_K)^T\mathrm{B}(\mathrm{v_I}+\mathrm{v}_K).\end{aligned} \tag{7}$$

From the conditions of the Lemma, we know that the first term of the sum holds, as $\mathrm{v}^T\mathrm{Av} > 0$. For the second term, we recall Lemma 1, yielding that $\mathrm{v}^T\mathrm{Bv} > 0$, thus we infer that the LHS product is greater than zero. From this, we get

$$\mathrm{A}+\mathrm{B} \succ 0.$$

If $\mathrm{v_I} = 0$, (7) is turned into

$$\mathrm{v}_K(\mathrm{A}+\mathrm{B})\mathrm{v}_K = \mathrm{v}_K\mathrm{A}\mathrm{v}_K > 0. \tag{8}$$

Thus, we prove that $\mathrm{A}+\mathrm{B} \succ 0$. □

Theorem 1. *[Existence and uniqueness of the solution of Problem 2] Let the collected dataset* D *be comprised of L trajectories drawn independently*

$$\mathrm{D} = \{(\mathrm{x}_\ell(k), \mathrm{u}_l(k)) : k \in \{0,...,N-2\}, \ell \in \{1,...,L\}\}$$

and let Σ_ℓ *be the empirical covariance of trajectory* ℓ*, i.e.,*

$$\Sigma_\ell = \frac{1}{N}\sum_{k=0}^{N-1}\begin{bmatrix}\mathrm{x}_\ell(k)\\ \mathrm{u}_\ell(k)\end{bmatrix}\begin{bmatrix}\mathrm{x}_\ell(k)\\ \mathrm{u}_\ell(k)\end{bmatrix}^T. \tag{9}$$

Then, the following statements are equivalent:

(i) the solution of Problem 1 exists and is unique;

(ii) the empirical covariance of the data, $\Sigma = \Sigma_1 + \Sigma_2 + ... + \Sigma_L$*, as in* (9) *is positive definite.*

Proof. Initially, let us define $\bar{\mathrm{x}}(k) = vec(\mathrm{X}(k))$ and $\bar{\mathrm{c}}(k) = vec(\mathrm{C}(k))$ and rewrite Problem 1

$$\begin{aligned}\underset{\bar{\mathrm{c}}}{\text{minimize}} \quad f(\bar{\mathrm{c}}) :=& \frac{1}{2}\sum_{k=0}^{N-2}\|\bar{\mathrm{x}}(k+1) - (\mathrm{D}(k)\otimes\mathrm{I})\bar{\mathrm{c}}(k)\|^2 +\\ & \frac{1}{2}\sum_{k=1}^{N-2}\|\lambda_k^{1/2}(\bar{\mathrm{c}}(k)-\bar{\mathrm{c}}(k-1))\|^2.\end{aligned} \tag{10}$$

Thus, applying the second-order condition for convex minimization problems from [1], which states that a function f is strictly convex if and only if its domain is convex and its Hessian is positive semidefinite. The Hessian of the cost function of our problem as stated in (10) is

$$\nabla^2 f(\bar{\mathrm{c}}) = \underbrace{diag(\mathrm{D}^T(k)\mathrm{D}(k)\otimes\mathrm{I})}_{\Pi} + \underbrace{\mathrm{F}^T\Upsilon\mathrm{F}}_{\Gamma} \tag{11}$$

and we must guarantee that $\nabla^2 f(\bar{c}) \succeq 0$, which is equivalent to say that the solution exists and is unique. From here on, we will use this knowledge to infer about the conditions on data.

Let us define the kernel of Γ as $\mathrm{Ker}(\Gamma) = \left\{ \begin{bmatrix} \bar{c} & \cdots & \bar{c} \end{bmatrix}^T : \bar{c} \quad \right\}$. From Lemma 2, if we have $\Pi + \Gamma \succ 0$, then we know that $\begin{bmatrix} \bar{c}^T & \cdots & \bar{c}^T \end{bmatrix} \Pi \begin{bmatrix} \bar{c} \\ \vdots \\ \bar{c} \end{bmatrix} > 0$.

Therefore,

$$
\begin{aligned}
&\bar{c}^T \left(\sum_{k=1}^{N-2} D^T(k) D(k) \otimes I \right) \bar{c} > 0 \\
\Leftrightarrow &\sum_{k=1}^{N-2} D^T(k) D(k) \succ 0 \\
\Leftrightarrow &\sum_{k=1}^{N-2} \begin{bmatrix} X^T(k) & U^T(k) \end{bmatrix}^T \begin{bmatrix} X^T(k) & U^T(k) \end{bmatrix} \succ 0 \qquad (12) \\
\Leftrightarrow &\sum_{k=1}^{N-2} \Big(\begin{bmatrix} x_1^T(k) & u_1^T(k) \end{bmatrix}^T \begin{bmatrix} x_1^T(k) & u_1^T(k) \end{bmatrix} + \\
&\cdots + \begin{bmatrix} x_\ell^T(k) & u_\ell^T(k) \end{bmatrix}^T \begin{bmatrix} x_\ell^T(k) & u_\ell^T(k) \end{bmatrix} \Big) \succ 0 \\
\Leftrightarrow &\Sigma_1 + \Sigma_2 + \cdots + \Sigma_L \succ 0
\end{aligned}
$$

Thus, we found the necessary and sufficient condition on data and proved that statements *(i)* and *(ii)* are equivalent. □

Theorem 1 allows to efficiently collect data, as we can address the covariance of the data as we collect it and verify its positive definiteness. When this condition is verified, we can stop the data collection and advance for the training.

Equation (6) makes evident that in order to find the optimal value for C, it is required to invert a matrix

$$\mathscr{A} := V^T V + F^T \Upsilon F,$$

$\mathscr{A} \in \mathbb{R}^{(N-1)(p+q) \times (N-1)(p+q)}$. This result shows that the approach requires the computation of a matrix heavily dependent on the number of instants considered in the trajectory being studied and the dimension of the system, meaning the result will become harder to obtain with an increased sampling rate and system complexity. When presented with a high sampling rate system, the size of matrices involved in the calculations can be very high, so solving directly the regularized least squares problem is not feasible. Regardless of this shortcoming, it is still important to understand the conditions in which the matrix $\mathscr{A}$ is invertible, since it will impact the numerical solution of the optimization problem. Theorem 2 addresses this point.

Theorem 2. *Let $\Upsilon \neq 0$. Then, matrix $\mathscr{A}$ is invertible if there exists a set of $p+q$ pairs $p+q$ pairs $(\ell, k) \in \{1, ..., L\} \times \{0, ..., N-2\}$ pairs such that the corresponding $\begin{bmatrix} x_\ell^T(k) & u_\ell^T(k) \end{bmatrix}$ are all linearly independent.*

Proof. The proof is made by contraposition, by proving that if the problem does not have a unique solution, then conditions of the theorem do not hold.

Let us consider then that $\mathscr{A}$ is not invertible. Then

$$
\begin{aligned}
\underset{\|\eta\|=1}{\exists \eta :} \quad \eta^T \mathscr{A} \eta = 0 &\Leftrightarrow \eta^T V^T V \eta + \eta^T F^T \Upsilon F \eta = 0 \\
&\Leftrightarrow \|V\eta\|^2 + \|\Upsilon^{1/2} F \eta\|^2 = 0 \quad . \\
&\Leftrightarrow V\eta = 0 \wedge \Upsilon^{1/2} F \eta = 0
\end{aligned} \qquad (13)
$$

Defining $\eta(k) = \begin{bmatrix} \eta_A^T(k) & \eta_B^T(k) \end{bmatrix}^T$ and $\eta = [\eta(k)]_{k=0}^{k=N-2}$, it is possible to write, for all k,

$$
\Upsilon^{1/2} F \eta = 0 \Leftrightarrow
$$
$$
\begin{cases} \eta_A(k) - \eta_A(k-1) = 0 \Leftrightarrow \eta_A(k) = \eta_A(k-1) := \bar{\eta}_A \\ \eta_B(k) - \eta_B(k-1) = 0 \Leftrightarrow \eta_B(k) = \eta_B(k-1) := \bar{\eta}_B \end{cases} \qquad (14)
$$

where the fact that $\Upsilon^{1/2}$ is invertible was used. This yields

$$\eta^* := \underbrace{\begin{bmatrix} I \\ \cdots \\ I \end{bmatrix}}_{\mathscr{N}} \underbrace{\begin{bmatrix} \bar{\eta}_A \\ \bar{\eta}_B \end{bmatrix}}_{\bar{\eta}} \qquad (15)$$

denoting the null space of F. In order for (13) to hold, η^* has to be contained by the null space of V whenever $\lambda \neq 0$, i.e.

$$V\eta^* = 0 \Leftrightarrow V\mathscr{N}\bar{\eta} = 0 \qquad (16)$$

that can be reduced to

$$\forall_k \forall_\ell \quad x_\ell^T(k)\bar{\eta}_A + u_\ell^T(k)\bar{\eta}_B = 0. \qquad (17)$$

The solution of (17) implies that either $\bar{\eta} = 0$, contradicting the hypothesis of the proof ($\|\eta\| = 1$), or that the vectors $\begin{bmatrix} x_\ell^T(k) & u_\ell^T(k) \end{bmatrix}$ in the dataset are linearly dependent, i.e., it is not possible to find $p+q$ linearly independent vectors for all $\ell = 1, ..., L$ and $k = 0, ..., N-2$. Thus, if $\lambda_k \neq 0$ for all k and the matrix $\mathscr{A}$ is singular, the condition of the Theorem cannot hold and, by contraposition, if it holds, then $\mathscr{A}$ is invertible and there exists a unique solution to Problem 2.

This concludes the proof of sufficiency of the conditions to the existence of solution. □

Remark 2. *The conditions for the existence and uniqueness of solution set forth by Theorem 1 can be clearly seen as equivalent to this new result, as both require the complete set of $\begin{bmatrix} x_\ell^T(k) & u_\ell^T(k) \end{bmatrix}$ vectors within the dataset to span $\mathbb{R}^{p+q}$.*

Solving (6) is the path adopted by classic solvers to perform its operations, which becomes a disadvantage of this technology, as it is not capable of keeping up with the complexity of the estimation. Hence, addressing this shortcoming is essential to configure the system identification problem of linear time-variant systems as a regularized least squares problem.

3. Closed form discrete LTV system identification

We can regard (4) for each instant k independently, such that it can be formulated as

Problem 3.

$$\underset{\mathrm{C}}{\textit{minimize}} \quad f(\mathrm{C}) := \underbrace{\frac{1}{2}\sum_{k=0}^{N-2} \|\mathrm{D}(k)\mathrm{C}(k) - \mathrm{X}'^T(k)\|_F^2}_{h(\mathrm{C})} + \underbrace{\frac{1}{2}\sum_{k=1}^{N-2} \|\lambda_k^{1/2}(\mathrm{C}(k) - \mathrm{C}(k-1))\|_F^2}_{g(\mathrm{C})}. \tag{18}$$

Accordingly, let us once again think of the derivative of the cost function as a sum of the derivatives of its parcels and have for the k-th instant

$$\nabla_{\mathrm{k}} f(\mathrm{C}) = \nabla_{\mathrm{k}} h(\mathrm{C}) + \nabla_{\mathrm{k}} g(\mathrm{C}), \tag{19}$$

knowing that the gradient at each instant k is $\nabla_{\mathrm{k}} f(\mathrm{C})$. The gradient of $h(\mathrm{C})$ at k is

$$\nabla_{\mathrm{k}} h(\mathrm{C}) = 2\mathrm{D}^T(k)(\mathrm{D}(k)\mathrm{C}(k) - \mathrm{X}'^T(k)). \tag{20}$$

The gradient of $g(\mathrm{C})$ is

$$\nabla g(\mathrm{C}) = 2\mathrm{F}^T\Upsilon\mathrm{FC}. \tag{21}$$

With F as in (3), we can compute $\mathrm{F}^T\Upsilon\mathrm{F}$

$$\mathrm{F}^T\Upsilon\mathrm{F} = \begin{bmatrix} \lambda_1\mathrm{I} & -\lambda_1\mathrm{I} & 0 & \dots & 0 & 0 & 0 \\ -\lambda_1\mathrm{I} & (\lambda_1+\lambda_2)\mathrm{I} & -\lambda_2\mathrm{I} & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -\lambda_{N-3}\mathrm{I} & (\lambda_{N-3}+\lambda_{N-2})\mathrm{I} & -\lambda_{N-2}\mathrm{I} \\ 0 & 0 & 0 & \dots & 0 & -\lambda_{N-2}\mathrm{I} & \lambda_{N-2}\mathrm{I} \end{bmatrix}. \tag{22}$$

Replacing (20) in (19) and addressing each element of (21), we get $\nabla_{\mathrm{k}} f(\mathrm{C}) =$

$$\nabla_k f(\mathrm{C}) = \mathrm{D}^T(k)(\mathrm{D}(k)\mathrm{C}(k) - \mathrm{X}'^T(k)) + [\mathrm{F}^T\Upsilon\mathrm{FC}](k) = 0. \tag{23}$$

We have, thus, to solve a linear system of equations.

Given the previously defined D, let us have the following auxiliary variables

$$\begin{cases} \mathrm{S}_{00} = \mathrm{D}^T(0)\mathrm{D}(0) + \lambda_1\mathrm{I} \\ \mathrm{S}_{N-2,N-2} = \mathrm{D}^T(N-2)\mathrm{D}(N-2) + \lambda_{N-2}\mathrm{I} \\ \mathrm{S}_{kk} = \mathrm{D}^T(k)\mathrm{D}(k) + (\lambda_k + \lambda_{k+1})\mathrm{I} \\ \mathrm{S}_{k,k-1} = \mathrm{S}_{k-1,k} = -\lambda_k\mathrm{I} \\ \mathrm{S}_{k,k+1} = -\lambda_{k+1}\mathrm{I} \\ \Theta_k = \mathrm{D}^T(k)\mathrm{X}'^T(k) \end{cases}. \tag{24}$$

Remark 3. *Our closed form solution, with a complexity that scales linearly with the number of time steps considered stems, from the key observation that the linear system of equations to be solved in* (23) *is in fact a block tridiagonal linear system. By considering the LU factorization method, our problem has a closed form solution with linear complexity [10].*

In the next section, we apply the *LU* factorization to (23). In general, it consists on a forward pass and then a backward pass, in this case with a complexity that scales linearly with the number of time steps considered in the system model.

Algorithm 1 COSMIC - Closed form system identification

Input: D, X′, $[\lambda_k]_{k=1}^{k=N-2}$
Output: C^*

```
 1: $\mathrm{S}_{00} \leftarrow \mathrm{D}^T(0)\mathrm{D}(0) + \lambda_1\mathrm{I}$
 2: $\mathrm{S}_{N-2,N-2} \leftarrow \mathrm{D}^T(N-2)\mathrm{D}(N-2) + \lambda_{N-2}\mathrm{I}$
 3: for $k \in [1,...,N-3]$ do
 4:     $\mathrm{S}_{kk} \leftarrow \mathrm{D}^T(k)\mathrm{D}(k) + (\lambda_k + \lambda_{k+1})\mathrm{I}$
 5: end for

 6: for $k \in [0,...,N-2]$ do
 7:     $\Theta_k \leftarrow \mathrm{D}^T(k)\mathrm{X}'^T(k)$
 8: end for

 9: $\Lambda_{00} \leftarrow \mathrm{S}_0$
10: $\mathrm{Y}_0 \leftarrow \Lambda_0^{-1}\Theta_0$

11: for $k \in [1,...,N-2]$ do                          ▷ forward pass
12:     $\Lambda_k \leftarrow \mathrm{S}_{kk} - \lambda_k^2\Lambda_{k-1}^{-1}$
13:     $\mathrm{Y}_k \leftarrow \Lambda_k^{-1}(\Theta_k + \lambda_k\mathrm{Y}_{k-1})$
14: end for

15: $\mathrm{C}(N-2) \leftarrow \mathrm{Y}_{N-2}$

16: for $k \in [N-3,...,0]$ do                         ▷ backward pass
17:     $\mathrm{C}(k) \leftarrow \mathrm{Y}_k + \lambda_{k+1}\Lambda_k^{-1}\mathrm{C}(k+1)$
18: end for
        return $\mathrm{C}^* \leftarrow \mathrm{C}$
```

Forward pass

For the forward pass, we solve $\mathrm{LY} = \Theta$, where L is

$$\mathrm{L} = \begin{bmatrix} \Lambda_0 & 0 & \cdots & 0 & 0 \\ \mathrm{S}_{1,0} & \Lambda_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \Lambda_{N-3} & 0 \\ 0 & 0 & \cdots & \mathrm{S}_{N-2,N-3} & \Lambda_{N-2} \end{bmatrix},$$

and Λ_k and $\Theta_{k,\ell}$ are square submatrices of size $(p+q)\times(p+q)$. To start the process, the unknown variables are initialized with

$$\begin{cases} \Lambda_0 = \mathrm{S}_{00} \\ \mathrm{Y}_0 = \Lambda_0^{-1}\Theta_0 \end{cases},$$

and for the subsequent time steps $k = 1, ..., N-2$, we incrementally compute

$$\begin{cases} \Omega_k = \mathrm{S}_{k,k-1}\Lambda_{k-1}^{-1}\mathrm{S}_{k-1,k} \\ \Lambda_k = \mathrm{S}_{kk} - \Omega_k \\ \mathrm{Y}_k = \Lambda_k^{-1}(\Theta_k - \mathrm{S}_{k,k-1}\mathrm{Y}_{k-1}). \end{cases} \tag{25}$$

Backward pass

Given the results of the forward pass, we can address the backward propagation by solving $\mathrm{MC} = \mathrm{Y}$, with

$$\mathrm{M} = \begin{bmatrix} \mathrm{I} & \Lambda_0^{-1}\mathrm{S}_{0,1} & \cdots & 0 & 0 \\ 0 & \mathrm{I} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathrm{I} & \Lambda_{N-3}^{-1}\mathrm{S}_{N-3,N-2} \\ 0 & 0 & \cdots & 0 & \mathrm{I} \end{bmatrix}. \tag{26}$$

The optimization variable in the last instant is initialized with $\mathrm{C}(N-2) = \mathrm{Y}_{N-2}$. For $i = N-3, ..., 0$,

$$\mathrm{C}(k) = \mathrm{Y}_k - \Lambda_k^{-1}\mathrm{S}_{k,k+1}\mathrm{C}(k+1). \tag{27}$$

Next, we state the main result of this work, about the convergence in a fixed, finite number of algebraic operations, i.e., as a closed form.

Theorem 3. *[Algorithm 1 solves* (18) *in closed form with linear complexity]*

Algorithm 1 yields a closed form solution for the system identification problem of the linear time-variant system from data, with linear complexity on the number of time steps. Namely, the number of multiplication operations to be performed by the Algorithm 1 is $c = (N-1)\left((p+q)^3 + (2p+3)(p+q)^2\right)$*, which is linear on the number of time steps N, and cubic on the size of state space and control space, and does not depend on L, the size of the dataset used for learning matrices* $[\mathrm{A}(k), \mathrm{B}(k)]_{k=0}^{N-2}$.

Proof. We must prove that solving $\nabla f(\mathrm{C}) = 0$ is equivalent to solving

$$\mathrm{LMC} = \Theta, \tag{28}$$

where L is a lower triangular matrix with non zero diagonal and first subdiagonal blocks, and M is an upper triangular matrix with diagonal identity submatrices and first superdiagonal non-zero.

Let us assume a system starting from the results in (23). We can rearrange the system of linear equations to address it as

$$\underbrace{\begin{bmatrix} \mathrm{D}^T(0)\mathrm{D}(0)\mathrm{C}(0) + \lambda(-\mathrm{C}(0) + \mathrm{C}(1)) \\ \vdots \\ \mathrm{D}^T(k)\mathrm{D}(k)\mathrm{C}(k) + \lambda(-\mathrm{C}(k-1) + 2\mathrm{C}(k) - \mathrm{C}(k+1)) \\ \vdots \\ \mathrm{D}^T(N-2)\mathrm{D}(N-2)\mathrm{C}(N-2) + \lambda(-\mathrm{C}(N-3) + \mathrm{C}(N-2)) \end{bmatrix}}_{\mathrm{LMC}}$$

$$= \underbrace{\begin{bmatrix} \mathrm{D}^T(0)\mathrm{X}'^T(0) \\ \vdots \\ \mathrm{D}^T(k)\mathrm{X}'^T(k) \\ \vdots \\ \mathrm{D}^T(N-2)\mathrm{X}'^T(N-2) \end{bmatrix}}_{\Theta},$$

which directly yields the definition of Θ, just like it was defined in (24). Decomposing the LHS of the previous definition, we can identify the dependence on C and single it out. Thus, we get

$$\underbrace{\begin{bmatrix} \mathrm{S}_{00} & -\lambda_1\mathrm{I} & 0 & 0 & 0 \\ \ddots & \ddots & \vdots & & \\ \ddots & -\lambda_k\mathrm{I} & \mathrm{S}_{kk} & -\lambda_{k+1}\mathrm{I} & \ddots \\ & & \vdots & & \\ 0 & 0 & 0 & -\lambda_{N-2}\mathrm{I} & \mathrm{S}_{N-2,N-2} \end{bmatrix}}_{\mathrm{LM}} \begin{bmatrix} \mathrm{C}(0) \\ \vdots \\ \mathrm{C}(k) \\ \vdots \\ \mathrm{C}(N-2) \end{bmatrix}. \tag{29}$$

Additionally, from the assumption of the form of matrices L and M, as previously defined, we get LM =

$$\begin{bmatrix} \Lambda_0 & \mathrm{S}_{01} & 0 & 0 & 0 & 0 & 0 \\ \mathrm{S}_{01} & \Lambda_1 + \Omega_1 & \mathrm{S}_{12} & \vdots & \vdots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \mathrm{S}_{k,k-1} & \Lambda_k + \Omega_k & \mathrm{S}_{k,k+1} & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \mathrm{S}_{N-4,N-3} & \Lambda_{N-3} + \Omega_{N-3} & \mathrm{S}_{N-3,N-2} \\ 0 & 0 & 0 & 0 & 0 & \mathrm{S}_{N-2,N-3} & \Lambda_{N-2} + \Omega_{N-2} \end{bmatrix} \tag{30}$$

yielding the results we stated in the formulation of the problem by making it equal to (29). Hence, we prove that the problem can be solved by a *LU* factorization and that solving equation (5), taking into account the nuances of the k-th instants, is equivalent to solving (28). For the complexity result, we analyze the multiplication operations involved in the presented algorithm as follows.

For the *forward pass* we need to invert all $N-1$ Λ_k matrices. If we use traditional inversion methods, the number of operations for each inversion is $(p+q)^3$. Multiplications by a scalar involve $2(p+q)^2$. The multiplication of the two matrices in Step 11 of Algorithm 1, Λ_k^{-1}, with a matrix $(p+q) \times p$ costs $p(p+q)^2$, gives a total number of operations for all $N-1$ steps of

$$c_{\mathrm{fwd}} = (N-1)\left((p+q)^3 + (p+2)(p+q)^2\right),$$

while the *backward pass* demands multiplication of a scalar by a matrix of size $(p+q)\times(p+q)$ accounting for $(p+q)^2$ operations, and a multiplication between a matrix of size $(p+q)\times(p+q)$, and one of size $(p+q)\times p$, yielding $p(p+q)^2$ operations. Thus, the total number of backward pass operations is

$$c_{\text{bwd}} = (N-1)\left((p+1)(p+q)^2\right).$$

$\square$

3.1. COSMIC validation

To demonstrate previous stated theoretical results, a simulation and testing environment was developed. A classical spring-mass-damper system is first simulated and excited in a Simulink model to allow for data collection both in LTI and LTV setups, and to guarantee sufficient data we input a sine function with different amplitude for each run of the simulation. It is also worth noting that in each simulation run, the initial conditions take a different value.

The validation is performed with parameter λ_k constant throughout the trajectory, thus being represented by λ. Afterwards, this data is used as input into COSMIC and multiple models are trained, in a Python environment, to understand the behavior of the proposed algorithm in different settings, by varying the λ or the noise that disturbs the state.

Finally, the estimated model is excited with previously selected inputs from which we know the expected result, and we are able to evaluate the performance. The metrics used are the estimation error and the prediction error.

Taking into account the spring-mass-damper system, considered the ground truth, the estimation error, i.e. how close the model is to the true system, is defined as $\|\hat{\text{C}}-\text{C}_{\text{gt}}\|_F$. Regarding the prediction power of COSMIC, we propose the evaluation of the predicted state error when compared to the true state from a simulated trajectory that was not part of the training data set. As such, we address the metric by defining the norm of the error as $\|\hat{\text{x}}(k+1)-\text{x}(k+1)\|_2$, allowing for the assessment of the error propagation. Moreover, for the following validation study, we assumed that the measurements were disturbed by noise, which can be characterized by $\omega \sim \mathcal{N}(\mu,\sigma^2)$ with $\mu = 0$ and variance σ^2.

To infer how the performance of the algorithm depends on the measurement noise and parameter λ, we opted to develop a parametric study where, for each noise variance, we tested the estimation for a reasonable range of values for parameter λ. Figure 2 exposes the variation of the estimation error with the noise added to the measurements for the LTV system and it is clear that for a noise standard deviation of about 10% of the maximum initial conditions, the algorithm is heavily affected. Reducing this value, the estimation error drops significantly. It is evident that smaller values for the parameter λ lead to a worse performance of the algorithm, as the variation between instants is not significant

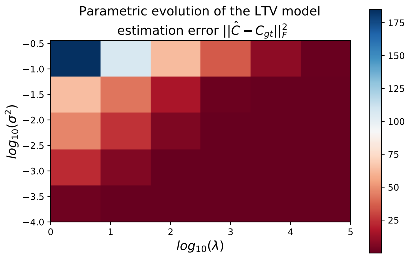
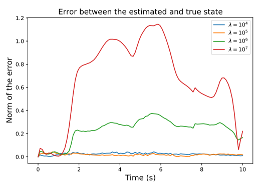


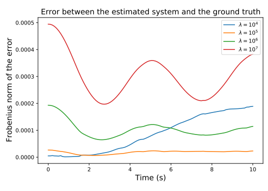
Figure 2: State estimation performance of the multiple methods tested, compared with the true state.

enough to allow such values, resulting much better with higher values of λ.

As part of the validation process, we can also evaluate what is the λ that best adapts the estimated model to the true system characteristics. To do so, we evaluate the system in the previous conditions, choosing an appropriate value for the noise, in this case a standard deviation of about 10% of the initial. From a simple analysis of the plots in Figure 3, we can say that the λ that best fits the system we are trying to estimate is $\lambda = 10^5$, as the lowest errors occur at this value.



(a) Estimation error.



(b) Prediction error.

Figure 3: Evaluation metrics evolution with time to infer the optimal $\lambda = 10^5$.

3.2. Benchmark

The novelty of the algorithm presented comes from its efficiency, mainly when compared to other approaches to the same problem. As we are solving a convex optimization problem, we can obtain a solution through `cvxpy`, a domain specific language that allows for easy problem syntax to express convex optimization problems in Python and is widely used [3]. For this specific use case, the ECOS solver [5], an interior-point solver for second-order cone programming, is automatically invoked. Hence, we evaluate the problem agnostic approach, as in (6) with this strategy.

In addition, we are also going to be evaluating the solution of (18) through a Stochastic Block Coordinate Descent (SBCD) algorithm. This approach allows for a simplification of `cvxpy` perspective, as it treats each C(k) as an independent optimization variable for an iteration of the algorithm, while keeping all the other values constant, simplifying the calculations and allowing for an easier derivative verification. By evaluating the value of the derivative at the new C, we can infer if the optimality condition was reached or not. However, this approach is iterative, which means that it may not be able to reach a solution if the proper limits are not found. Nevertheless, the SBCD approach enables different loss functions for the data fidelity term, like, e.g., the Huber M-estimator, robust to outlier measurements. Finally, we run again the COSMIC algorithm. These simulations, as all the ones run in a Python environment, were made in Google Colab development setting, with 13GB of RAM available and the Intel(R) Xeon(R) CPU @ 2.20GHz processor.

Table 1: Performance comparison of different system identification solutions.

Instants	**cvxpy** **Time**	***SBCD*** **Time**	***Closed form*** **Time**
100	2.929	0.031	0.014
1000	42.588	0.273	0.106
10000	*	3.531	1.017
100000	*	21.055	9.696

*Session crashed after using all available RAM.

To perform the comparison in Table 1, as we are studying the performance of the methods against cost and time, we simulated a 1/10 sampling rate trajectory and varied its length between 10 s, 100 s, 1000 s and 10000 s with $\lambda = 0.001$ and stopping the SBCD at 1 million iterations, while using the setting presented previously.

The cvxpy approach tries to solve the problem for all time steps simultaneously, leading to a solution that requires the machine to invert a $(N-1)(p+q) \times (N-1)(p+q)$ matrix, as stated in (6). As the number of instances per trajectory increases, the complexity also increases and the cvxpy approach can no longer compute a solution in due course, while both SBCD and closed form approaches continue to perform as expected. Moreover, for more disturbed data, the computation needed to reach this solution becomes more difficult. Regarding the cost function, no significant changes are observed from one method to another

This leads to the conclusion that the closed form algorithm is the best option to solve (18), as it can achieve minimal cost values with an efficient use of computational resources, reaching a solution for all cases tested, performing significantly better than the other approaches. Moreover, analyzing Table 1, we can experimentally observe the linearity with the number of instances per trajectory that is stated in Theorem 3.

4. Controller design for estimated model

To follow up on the system identification from data, we derive an optimal control law for the LTV estimated model based on a dynamic programming setup, as detailed in [18] for discrete-time LTV LQR. Algorithm 2 summarizes the approach.

Algorithm 2 Dynamic Programming for LTV controller design

```
Input: H, Q, R, A, B, x
Output: K

1: $P_{N-1} \leftarrow H$

2: for $k \in [N-2, ..., 0]$ do
3:     $K_k \leftarrow \left(R_k + B^T(k)P_{k+1}B(k)\right)^{-1} B^T(k)P_{k+1}A(k)$
4:     $P_k \leftarrow Q_k + K_k^T R_k K_k + (A(k) - B(k)K_k)^T P_{k+1} (A(k) - B(k)K_k)$
5: end for
       return K
```

Regarding the estimated model, we used the parameters found to be the most suiting to develop and save a model, i.e. $\lambda = 10^5$, and the measurements were disturbed by noise with standard deviation $\sigma = 0.06$. Then, we load the new model into another Python environment to allow for it to be used in the dynamic programming algorithm that results in an optimal control path. Afterwards, the solution is returned to Simulink and tested in the original spring-mass-damper system.

For the design phase of the controller, we develop a tuning framework taking into account the LQR needs, keeping matrices Q_k and R_k constants throughout the trajectory and depending only on the elements from their diagonals, entailing that the most suiting design parameters are $\frac{q_v}{q_x} = 10^{-1}$, $\frac{r}{q_x} = 10^{-3}$ and $q_x = 1$.

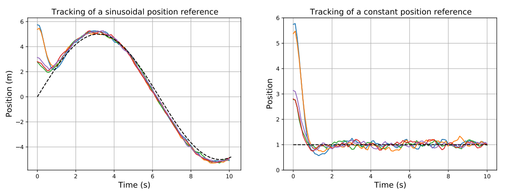


Figure 4: System position response from different initial conditions.

Figure 4 shows the system response to different inputs when controlled by the LTV LQR drawn from the estimated model. After performing a statistical analysis of the tracking error, we can confidently say that the estimated model is a good approximation of the system, as shown by the close values of the statistical metrics in Table 2. Moreover, it allows for a better controller synthesis than previously used techniques, performing better than the LTI LQR, presenting a smaller mean and sum of squared error, once again from Table 2. Thus, we can be stated that the estimated model from COSMIC is a good approximation of the spring-mass-damper system used throughout this work and the dynamic programming strategy worked well for the problem posed.

Hence, this system identification and controller design framework is validated and we can further test it in more complex environments.

Table 2: Statistical comparison of the controller design methods.

Control	**Mean**	σ	$\sum$error^2
Estimated model	0.1050	0.7346	0.5506
Ground truth	0.0994	0.7441	0.5635
Time invariant	0.1218	0.7425	0.5661

5. Comet Interceptor as a Case Study

5.1. Attitude dynamics and kinematics

The attitude dynamics and kinematics of the spacecraft are described by the general representation of these phenomenons, i.e.

$$\begin{cases} \dot{\mathrm{q}}(t) = \frac{1}{2}\mathrm{q}(t) \otimes \begin{bmatrix} 0 & \omega(t) \end{bmatrix}^T \\ \mathrm{J}\dot{\omega}(t) = [\mathrm{J}\omega(t)] \times \omega(t) - \mathrm{u}(t) + \mathrm{T}(t) \end{cases} \tag{31}$$

The attitude is represented by the quaternion $\mathrm{q} = \begin{bmatrix} q_0 & q_x & q_y & q_z \end{bmatrix}^T$, with norm 1 and $\{q_0, q_x, q_y, q_z\} \in \mathbb{R}$, and it transforms a vector represented in the the body fixed frame, centered in the spacecraft, in a vector in the inertial frame centered in the target. The angular velocity is $\omega \in \mathbb{R}^3$. Then, to complete the attitude dynamics, we also define torque input as $\mathrm{u} \in \mathbb{R}^3$ and the external disturbances $\mathrm{T} \in \mathbb{R}^3$, which can be caused by multiple factors but are not going to be considered for the first stage of this application. J represents the inertia in the spacecraft body frame. This system is nonlinear. The pointing error is defined as the angle between the direction that the camera is truly pointing at, r_ℓ, and the direction of the comet, d_c, both in the body fixed frame. Thus, the cosine of the pointing error is $\cos(\theta_{PE}) = \mathrm{r}_\ell^T \mathrm{r}_c$.

The latter direction can be obtained from the position of the spacecraft represented in the inertial frame, using the attitude information that can be retrieved from the attitude kinematics and the translation information, which entails $\begin{bmatrix} 0 & \mathrm{d}_c^T \end{bmatrix}^T = \mathrm{q}^* \otimes \begin{bmatrix} 0 & {}^I\mathrm{d}_c^T \end{bmatrix}^T \otimes \mathrm{q}$, where ${}^I\mathrm{r}_c$ a the unitary vector.

As we wish to drive the error to zero, the system that is taken into account for the controller present in the data collection environment is

$$\begin{cases} \dot{\delta\theta}(t) = -[\bar{\omega}(t)]_\times \delta\theta(t) + \delta\omega(t) \\ \dot{\delta\omega}(t) = \mathrm{J}^{-1}\mathrm{A}_\omega \delta\omega(t) - \mathrm{J}^{-1}\delta\mathrm{u}(t) \\ \dot{\delta\iota}(t) = \delta\theta(t), \end{cases} \tag{32}$$

here representing the integral of the attitude error as $\int \delta\theta = \delta\iota$.

We are performing this implementation in a Low Fidelity Simulator for the Comet Interceptor mission, with representations of separate dynamical properties of the system, which result in first solutions for the problems encountered that can then be tuned and adapted to the full overview of the system. Thus, the work presented here is an approach to the first step of the design process, where the nonlinear system is represented, without external perturbations, to allow for a deeper knowledge of the system behavior to the control inputs that may be given to it and then possibly use the solutions found as the first approach to more complicated scenarios. We follow the work flow of the previous validation, starting with the learning phase, then deriving an optimal controller and finalizing with the testing in the original simulation.

For the purpose of this proof of concept, and considering the motivation of this work, we opt to address the problem with varying λ_k. Taking into account the previously knowledge about the system, i.e. the angular velocity over the y axis that is desired, we establish that the work can be performed considering two different values for λ_k and three zones of the trajectory where this value will be maintained constant.

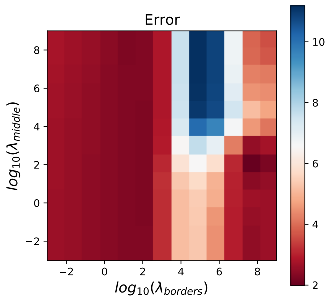


Figure 5: Parametric results for the variation of λ_k along the trajectory.

Analyzing Figure 5 (a), we can infer that the lowest estimation error from the previous estimated state occurs for $\lambda_{middle} = 10^2$ and $\lambda_{borders} = 10^8$. For the instantaneous estimation error, the error decreases as the λ_k increases. We choose the best pair to be the one with the lowest mean estimation error, from estimated state, as it also appears to have a low error when calculated with the true state.

This result is in line with the analysis of the system and its expected behavior. For the zones further away from the closest approach point, the system dynamics is much slower and the variations between instants are less disruptive, which is in line with a higher value of λ_k, that imposes a narrower difference between the optimal variable variation in consecutive instants. In the middle of the trajectory, the spacecraft is at its closest point to the target and its attitude needs to change much faster to maintain a pointing error small enough, thus the system changes much faster and the difference between to instants needs to be larger, which is allowed by the smaller value of λ_k.

Applying the same dynamic programming framework, we also developed a new controller and we are now able to input the optimal gains calculated for each distinct instant into the Low Fidelity Simulator, instead of the constant gain that had been used to collect the data. Figure 6 shows the results from varying initial conditions and it is clear that the controller from the linear system identification performed by COSMIC is able to control the nonlinear system and is a good option for the first approach to the GNC framework of the mission.

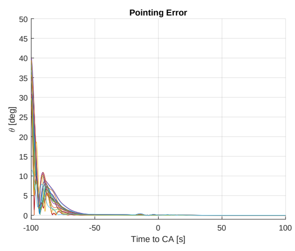


Figure 6: Pointing error resulting from the designed controller application.

6. Conclusions

The main breakthrough of this work was the development of COSMIC, a closed form system identification algorithm from data for linear time-variant systems, formulating the identification problem as a regularized least squares, with a regularization term that imposes a constrained variation between the solution consecutive instances. Moreover, we derived a condition on data to guarantee a solution. COSMIC performs considerably better than oher conventional solutions. The validation and testing, including in a Low Fidelity Simulator of Comet Interceptor, proved that this approach is a fair solution for the attitude control problem presented as motivation and should be accounted for in the initial phases of engineering projects to derive valid controllers and ease the design process. In the future, COSMIC can be tested in simulators with more detailed environments and can be taken advantage of for possible online learning and adaptive control solutions.

References

[1] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[2] S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics.*, 2018.

[3] S. Diamond and S. Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

[4] R. Dobbe, S. Liu, Y. Yuan, and C. Tomlin. Blind identification of fully observed linear time-varying systems via sparse recovery. *Automatica*, 100, 2019.

[5] A. Domahidi, E. Chu, and S. Boyd. Ecos: An SOCP solver for embedded systems. In *2013 European Control Conference (ECC)*. IEEE, 2013.

[6] S. Dudul and A. Ghatol. Identification of linear dynamical time-variant systems using feedforward neural network. *IE (I) Journal*, 2004.

[7] ESA. Assessment of Mission to Intercept a Long Period Comet or Interplanetary Object. CDF Study Report, ESA, Dec. 2019.

[8] S. Formentin and A. Chiuso. Control-oriented regularization for linear system identification. *Automatica*, 127, 2021.

[9] M. Hardt, T. Ma, and B. Recht. Gradient descent learns linear dynamical systems. *Journal of Machine Learning Research*, 2016.

[10] E. Isaacson and H. B. Keller. *Analysis of Numerical Methods*. Dover Publications, New York, 06 1994.

[11] S. N. Kumpati, P. Kannan, et al. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27, 1990.

[12] F. Lamnabhi-Lagarrigue, A. Annaswamy, S. Engell, A. Isaksson, P. Khargonekar, R. M. Murray, H. Nijmeijer, T. Samad, D. Tilbury, and P. Van den Hof. Systems & control for the future of humanity, research agenda: Current and future roles, impact and grand challenges. *Annual Reviews in Control*, 43, 2017.

[13] S. Lin, H. Wang, and J. Zhang. System identification via meta-learning in linear time-varying environments. *arXiv:2010.14664*, 2020.

[14] L. Ljung. System identification. In *Signal Analysis and Prediction*, pages 163–173. Birkhäuser Boston, 1998.

[15] T. Sarkar and A. Rakhlin. Near optimal finite time identification of arbitrary linear dynamical systems. In *International Conference on Machine Learning*, pages 5610–5618. PMLR, 2019.

[16] J. Schoukens and L. Ljung. Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, 39(6):28–99, 2019.

[17] L. Vanbeylen, E. Louarroudi, and R. Pintelon. How nonlinear system identification can benefit from recent time-varying tools: the time-varying best linear approximation. In *52nd IEEE Conference on Decision and Control*. IEEE, 2013.

[18] S. H. Zak. *Systems and control*, volume 198. Oxford University Press, 2003.