Mechanics Based Design of Structures and Machines
An International Journal

ISSN: 1539-7734 (Print) 1539-7742 (Online) Journal homepage: www.tandfonline.com/journals/lmbd20

# Data-driven prediction of subsystem dynamics in the explicit co-simulation of multibody systems

Maciej Pikuliński, Paweł Malczyk, Antonio J. Rodríguez & Francisco González

**To cite this article:** Maciej Pikuliński, Paweł Malczyk, Antonio J. Rodríguez & Francisco González (2026) Data-driven prediction of subsystem dynamics in the explicit co-simulation of multibody systems, Mechanics Based Design of Structures and Machines, 54:1, 2541259, DOI: 10.1080/15397734.2025.2541259

**To link to this article:** https://doi.org/10.1080/15397734.2025.2541259

MECHANICS BASED DESIGN OF STRUCTURES AND MACHINES
2026, VOL. 54, NO. 1, 2541259
https://doi.org/10.1080/15397734.2025.2541259



# Data-driven prediction of subsystem dynamics in the explicit co-simulation of multibody systems

Maciej Pikuliński[a], Paweł Malczyk[a], Antonio J. Rodríguez[b], and Francisco González[b]

[a]Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology, Warsaw, Poland; [b]Laboratorio de Ingeniería Mecánica - CITENI - Industrial Campus Ferrol, Universidade da Coruña, Ferrol, Spain

**ABSTRACT**

Co-simulation is an effective way to predict the dynamics of complex engineering setups, in which the main system is divided into subsystems. Each subsystem is integrated by a particular solver, and the coordination of these simulation units is coordinated by means of the exchange of limited amounts of information at specific points in time. This enables the implementation of modular computing environments, but it often introduces numerical errors in the solution that deteriorate the quality of the results and may eventually lead to the instability of the integration. Implicit co-simulation schemes remove these errors by iterating over each step. In explicit co-simulation setups this is not possible and alternative solutions are necessary. This work presents a data-driven approach to predict subsystem dynamics in explicit co-simulation setups, aimed at mitigating the energy errors introduced by the discrete-time co-simulation interface. The proposed solution only uses information contained in the coupling variables exchanged between subsystems and does not need knowledge of their internal details. The method has been tested with nonlinear and multirate benchmark problems. Results confirmed the ability of the proposed solution to remove numerical errors caused by the coupling interface and improve the accuracy and stability of the co-simulation of the overall system dynamics.

**ARTICLE HISTORY**
Received 6 August 2024
Accepted 16 July 2025

**KEYWORDS**
Explicit co-simulation; multibody system dynamics; dynamic mode decomposition; data-driven methods

## 1. Introduction

Simulation is currently a widespread and valuable tool in product development cycles, used to reduce costs and shorten the time lapse between conceptualization and market release. Multibody system dynamics, in particular, is a field of Mechanics that has experienced sustained growth in recent decades due to improvements in computer hardware and the formulation of novel methods to address the simulation of a wide range of phenomena (such as flexibility (Bauchau 2011; Gerstmayr 2024), contacts, and friction (Anitescu and Tasora 2010; Flores, Ambrósio, and Lankarani 2023; Wojtyra, Pękal, and Frączek 2020)), to optimize and control mechanical systems (López Varela, Dopico Dopico, and Luaces Fernández 2023; Maciąg, Malczyk, and Frączek 2020; Pikuliński and Malczyk 2021), and to use computational resources to boost efficiency (Malczyk

CONTACT Maciej Pikuliński maciej.pikulinski@pw.edu.pl Institute of Aeronautics and Applied Mechanics, Warsaw University of Technology, Nowowiejska 24, 00-665 Warsaw, Poland

Supplemental data for this article can be accessed online at https://doi.org/10.1080/15397734.2025.2541259.
Communicated by Andreas Zwölfer.

et al. 2019). Nowadays, multibody dynamics software is expected to deal with complex engineering systems including multiphysics effects while performing in the most efficient way possible. Co-simulation offers a response to this need, enabling the coupling of component-level simulation from different domains and the distribution of computing tasks over a network of resources (Hu et al. 2022; Olivier, Verlinden, and Kouroussis 2022).

Co-simulation environments are simulation setups in which two or more dynamic solvers are interfaced through the exchange of a limited set of variables at specific communication points in time (González et al. 2011). The integration of the dynamics of each solver during a macro-step, i.e., between two consecutive communication points, proceeds without further input from the rest of the environment, which increases the modularity of the simulation but, at the same time, makes it necessary to coordinate the different subsystems and to ensure the consistency of their results. This coordination can be achieved iterating the integration of the dynamics of each subsystem between communication points until convergence is reached, in an implicit co-simulation scheme. In some applications, however, it is impossible to follow this approach, because some subsystems cannot reset their internal state and go back to a past instant in time to restart their integration, or because the time available to carry out numerical computations is limited. This is the case, for instance, of cyber-physical devices, in which at least one subsystem is a hardware device (Glumac et al. 2022). In these situations, explicit co-simulation schemes that evaluate each macro-step only once are mandatory.

Implicit schemes are often more stable and accurate than their explicit counterparts (Kübler and Schiehlen 2000; Schweizer, Li, and Lu 2015). The latter frequently require that at least some subsystems perform an internal input extrapolation, which results in discontinuities at the co-simulation interface (Rodríguez et al. 2022). This gives rise to the introduction of errors in the energy balance of the overall system and degrades the performance of the numerical integration, leading to inaccurate results and instability in extreme cases. Selecting appropriate extrapolation methods is a possible way to handle this problem (Ben Khaled-El Feki et al. 2017; Rahikainen, González, and Naya 2020). In general, this is not straightforward, because the selection depends on the overall system properties and the numerical methods used to integrate the subsystems (Nopour et al. 2024). Moreover, extrapolation does not take into account the dynamics of the system under study, and it is difficult to assess how closely it represents its actual behavior. Adjusting the macro step-size to minimize the impact of energy losses at the interface is another solution (Sadjina et al. 2017), although its use is complicated if the step-size is fixed by the requirements of some components, as it may happen in Hardware-in-the-Loop environments. To overcome this limitation in real-time setups, alternative methods based on the introduction of corrections in the coupling variables have been proposed (Benedikt et al. 2013). These corrective actions can be based on direct energy measurements from the coupling variables (Rodríguez et al. 2022) or the definition of control schemes, sometimes with additional knowledge of the subsystem internals (Chen et al. 2021; Tamellin et al. 2022). The need for such explicit co-simulation corrections has been confirmed in the simulation of hydraulically actuated machinery (Peiret, González, et al. 2020), electromechanical systems (Eguillon, Lacabanne, and Tromeur-Dervout 2022), robotics problems (Peiret, González, et al. 2020; Raoofian, Dai, and Kövecses 2024), and flexible multibody dynamics (Dai et al. 2024), among other applications of industrial interest.

The availability of information about the implementation details of the subsystems enables the control and removal of errors in explicit co-simulation. It is desirable, however, to put forward solutions that rely exclusively on the coupling variables exchanged through the co-simulation interface. These can be applied without modifications to diverse kinds of problems and can still be used even if descriptions of the subsystem dynamics do not exist or are not disclosed to the co-simulation environment. This is a common situation in cyber-physical setups, or when intellectual property rights motivate the treatment of some subsystems as *black boxes*.

The critical challenge in system identification is discovering dynamics from data and finding data-driven representations that make nonlinear systems amenable to linear analysis. The dynamic mode decomposition (DMD) (Schmid 2022), originated from fluid mechanics, seems to be a highly versatile and powerful approach to discovering dynamics from time-series recordings, numerical simulations, and experimental data. The DMD is an equation-free, data-driven approach that finds dynamical models useful for short-time future state prediction and control of nonlinear system dynamics (Kaiser, Kutz, and Brunton 2021; Proctor, Brunton, and Kutz 2016). It is shown in the state–of–the–art that DMD is connected to the underlying nonlinear system dynamics through the infinite-dimensional Koopman operator, for which finite-dimensional approximations in terms of modes are numerically sought. A vital benefit of the DMD framework and its variety of extensions, as an identification approach built entirely on measurement data, is the simple formulation of techniques well-known from linear algebra. As far as multibody and robotics fields are concerned, there are some successful DMD-based applications (Berger et al. 2015; Bruder et al. 2021; Folkestad, Pastor, and Burdick 2020) to learn a robotic system's nonlinear dynamics and subsequently apply a control strategy for given tasks. To the best of the authors' knowledge, the DMD approach has not been used in the co-simulation setups where one solver evaluates multibody equations of motion. The observation highly motivates further work in this area.

In this work, we present a data-driven strategy to improve the accuracy and stability of the explicit co-simulation of multibody systems that extracts information from the coupling variables handled at the co-simulation interface. The proposed approach replaces input extrapolation within selected subsystems with a data-driven prediction of the evolution of the coupling variables that they receive as inputs during a macro step-size. The method is conceptually similar to the reduced interface model introduced in Peiret et al. (2018) and Peiret, González, et al. (2020), but does not need a description of the dynamics of the subsystem that provides the input values. Instead, the data conveyed *via* the coupling variables are processed by means of dynamic mode decomposition (DMD) to infer the behavior of that subsystem.

The proposed method has been tested in the explicit co-simulation of mechanical systems, including nonlinear multibody systems and multiphysics assemblies in multirate co-simulation setups. Results confirmed that, in subsystems subjected to direct feedthrough, the proposed method improved the accuracy of the co-simulation, bringing the outcomes closer to theoretical predictions and those obtained from monolithic simulation runs. Additionally, the method demonstrated its capability to stabilize co-simulation problems that would otherwise become unstable when constant input extrapolation was employed.

## 2. Data-driven inference of subsystem dynamics

In this paper, we put forward a data-driven framework to improve co-simulation accuracy. We first provide an introduction to the numerical issues of explicit co-simulation, and then we show how to use DMD to infer subsystem dynamics and address these issues.

### 2.1. Numerical issues in explicit co-simulation

The numerical issues that affect explicit co-simulation setups have often been illustrated with a linear oscillator in the co-simulation literature, e.g. (González et al. 2011; Schweizer, Li, and Lu 2015; Zhang et al. 2022), like the one shown in Fig. 1 following a force-displacement arrangement.

With the selected coupling configuration, subsystem $\mathcal{S}_1$ is subjected to direct feedthrough. The impact of direct feedthrough on co-simulation depends on the coupling arrangement selected to interface the manager and the subsystems. In a parallel explicit co-simulation scheme, this leads to a degradation of the results (Arnold, Clauss, and Schierz 2013) and introduces artificial energy in the integration of the system dynamics (Rodríguez et al. 2022).

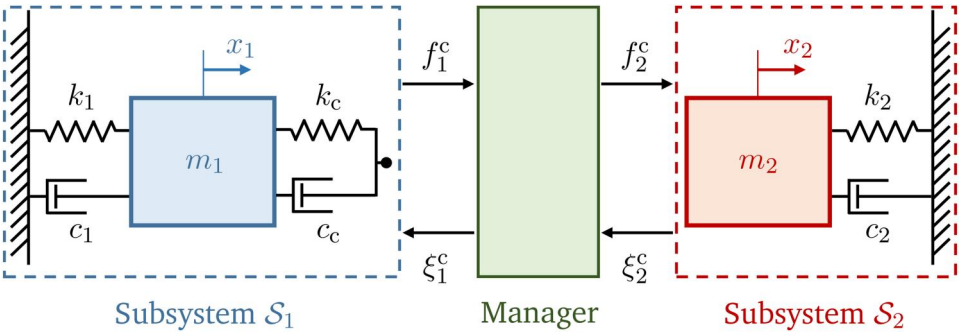


**Figure 1.** A linear oscillator co-simulated according to a force-displacement scheme.

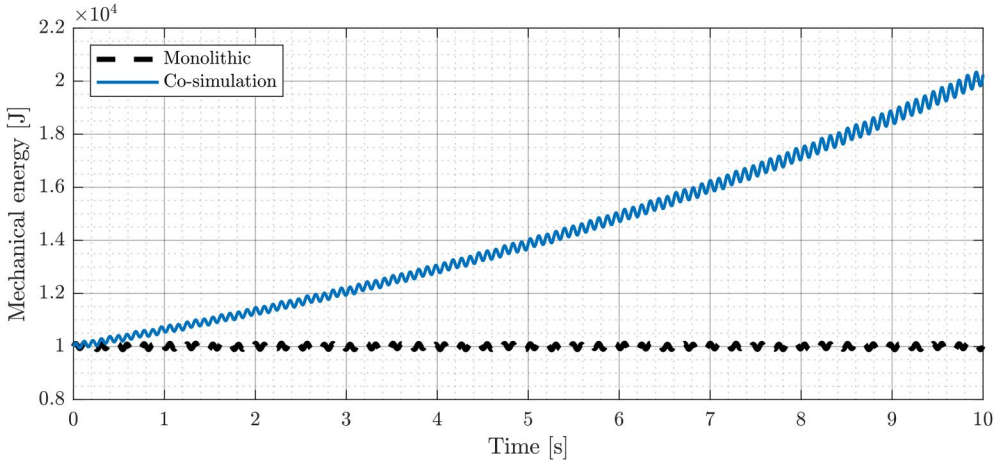


**Figure 2.** Mechanical energy during the simulation of a 10-s motion of the linear oscillator.

The double linear oscillator in Fig. 1 has been used in the literature to confirm that the use of approximated input values degrades the accuracy of the co-simulation results, which may lead to unstable behavior in extreme cases, e.g. (Rodríguez et al. 2022; Sadjina et al. 2017; Tamellin et al. 2022). Figure 2 compares the mechanical energy obtained during a 10-s co-simulation of the system motion following a single-rate explicit Jacobi scheme, to the results delivered by its monolithic counterpart. In both cases, the system damping was set to zero and a semi-implicit forward Euler integration formula was used with a step-size $h = 1$ ms. The oscillator parameters were set to $m_1 = m_2 = 1$ kg, $k_1 = 10$ N/m, $k_c = 100$ N/m, and $k_2 = 1000$ N/m. The initial conditions of the simulation were $x_1 = x_2 = 0$, $\dot{x}_1 = 100$ m/s, and $\dot{x}_2 = -100$ m/s. It can be shown that, under the same conditions, a single-rate displacement-displacement Jacobi co-simulation without direct feedthrough delivers the same results as the monolithic integration.

The errors in the mechanical energy shown in Fig. 2 translate as well into inaccurate predictions of the system motion. The use of higher-order extrapolation, e.g., first-order hold (FOH) or second-order hold (SOH) to approximate the unknown subsystem inputs may alleviate in some cases the errors introduced at the co-simulation interface. However, it is difficult to predict its impact on the accuracy of the results and it often requires a preliminary tuning stage to determine appropriate co-simulation configurations, which stems from the fact that extrapolation is not directly based on the actual system dynamics (Rahikainen, González, and Naya 2020).

### 2.2. DMD For explicit co-simulation

We provide next a data-driven framework to infer subsystem dynamics in explicit co-simulation environments to make the co-simulation much less vulnerable to numerical errors, e.g., ensuring the overall energy balance of the simulation. Specifically, we use Dynamic Mode Decomposition (DMD) (Proctor, Brunton, and Kutz 2016) to extract local-linear models within one subsystem out of another subsystem input-output characteristics. This enables a physics-based prediction of input $\mathbf{u}_{k+1}$ prior to the evaluation and return of $\mathbf{y}_{k+1}$, which can be used to address the numerical issues described in Sec. 2.1. Figure 3 illustrates the concept in its application to the linear oscillator in a force-displacement co-simulation setup. Subsystem $\mathcal{S}_1$ needs input $\mathbf{u}_1 = x_2$ to correctly evaluate its output $\mathbf{y}_1 = f^c$, the coupling force between the masses, and so is subjected to direct feedthrough. By means of DMD, the dynamics of subsystem $\mathcal{S}_2$ is inferred within $\mathcal{S}_1$ using the time-history of coupling variables exchanged between both subsystems. The availability of an inferred model of the dynamics of $\mathcal{S}_2$ makes it possible for subsystem $\mathcal{S}_1$ to predict the input that should be received at the final communication point during each macro-step, and so use it to calculate and return a more accurate value of the coupling force $f^c$.

The proposed method is similar to the use of reduced interface models (RIM) presented in Peiret et al. (2018) and Peiret, González, et al. (2020), because both approaches synthesize dynamics models that are used as physics-based auxiliary tools to improve the determination of subsystem inputs. However, there exist important differences between them. First, while RIMs were initially designed to enter the co-simulation environment as additional subsystems, the inferred models here presented are implemented inside the computational description of a subsystem with direct feedthrough, although they could also be implemented as standalone subsystems in multirate schemes. Second, and most important, building the inferred model does not require an explicit knowledge of the dynamics of the subsystem that they represent, as it is generated only from the information conveyed by the coupling variables exchanged through the co-simulation interface.

### 2.3. Windowed DMD algorithm

The DMD technique extracts the relationships between pairs of state measurement data $\widehat{\mathbf{x}}$ received from a subsystem and input (actuation) signals $\widehat{\mathbf{u}}$ submitted to the same subsystem. The discrete state-space representation of a linear system with $p$ inputs and $n$ state variables is written in the following form:

$$\widehat{\mathbf{x}}_{k+1} = \mathbf{A}_k\widehat{\mathbf{x}}_k + \mathbf{B}_k\widehat{\mathbf{u}}_k = \begin{bmatrix}\mathbf{A}_k & \mathbf{B}_k\end{bmatrix}\cdot\widehat{\mathbf{z}}_k \quad \text{and} \quad \widehat{\mathbf{z}}_k = \begin{bmatrix}\widehat{\mathbf{x}}_k \\ \widehat{\mathbf{u}}_k\end{bmatrix}, \tag{1}$$

where $\widehat{\mathbf{x}}_k \in \mathcal{R}^n$ is the state vector, $\widehat{\mathbf{u}}_k \in \mathcal{R}^p$ is the input vector. Moreover, $\mathbf{A}_k \in \mathcal{R}^{n\times n}$ is the state matrix and $\mathbf{B}_k \in \mathcal{R}^{n\times p}$ is the input matrix at step $k$. In the following paragraphs, we use the symbol $\widehat{\mathbf{y}}_k$ to denote the state $\widehat{\mathbf{x}}_{k+1}$ in the next time instant, i.e., $\widehat{\mathbf{y}}_k = \widehat{\mathbf{x}}_{k+1}$.

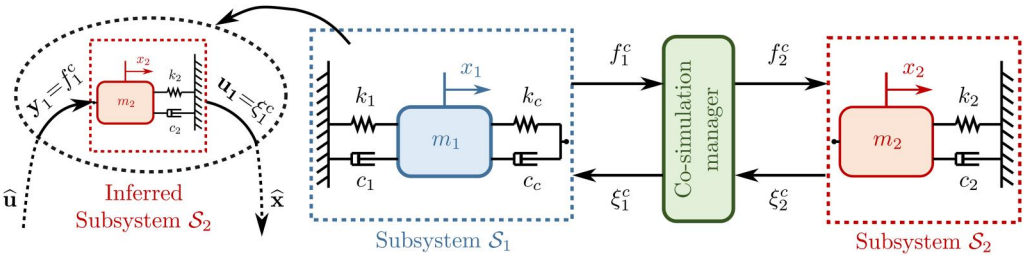


**Figure 3.** A two-degree-of-freedom linear oscillator (force-displacement coupling scheme) and inferred subsystem dynamics.

The discrete linear subsystem dynamics (1) can be generalized to take into account data snapshots:

$$\widehat{\mathbf{Y}}_k \approx \mathbf{A}_k\widehat{\mathbf{X}}_k + \mathbf{B}_k\widehat{\mathbf{U}}_k = \begin{bmatrix}\mathbf{A}_k & \mathbf{B}_k\end{bmatrix}\cdot\widehat{\mathbf{Z}}_k \quad \text{and} \quad \widehat{\mathbf{Z}}_k = \begin{bmatrix}\widehat{\mathbf{X}}_k \\ \widehat{\mathbf{U}}_k\end{bmatrix}, \tag{2}$$

where the data matrices can be arranged as follows:

$$\widehat{\mathbf{X}}_k = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{x}}_{k-w+1} & \widehat{\mathbf{x}}_{k-w+2} & \cdots\ \widehat{\mathbf{x}}_k \\ | & | & | \end{bmatrix}, \quad \widehat{\mathbf{U}}_k = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{u}}_{k-w+1} & \widehat{\mathbf{u}}_{k-w+2} & \cdots\ \widehat{\mathbf{u}}_k \\ | & | & | \end{bmatrix}$$

$$\widehat{\mathbf{Y}}_k = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{x}}_{k-w+2} & \widehat{\mathbf{x}}_{k-w+3} & \cdots\ \widehat{\mathbf{x}}_{k+1} \\ | & | & | \end{bmatrix} = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{y}}_{k-w+1} & \widehat{\mathbf{y}}_{k-w+2} & \cdots\ \widehat{\mathbf{y}}_k \\ | & | & | \end{bmatrix}. \tag{3}$$

The quantities $\widehat{\mathbf{X}}_k$, $\widehat{\mathbf{U}}_k$, and $\widehat{\mathbf{Y}}_k$ are snapshot matrices. We use a sliding window of size $w$ containing the most recent snapshots to identify the subsystem dynamics. Each iteration of the algorithm consists of two steps, namely *updating* and *downdating*. The former takes into account the current snapshot $(\widehat{\mathbf{z}}_{k+1}, \widehat{\mathbf{y}}_{k+1})$, and the latter removes the effect of the old data samples $(\widehat{\mathbf{z}}_{k-w+1}, \widehat{\mathbf{y}}_{k-w+1})$, which run out of the sliding window of size $w$. The process has been presented in Fig. 4.

The DMD is focused on finding best-fit approximations $\mathbf{S}_k = \begin{bmatrix}\mathbf{A}_k & \mathbf{B}_k\end{bmatrix}$ to the unknown state and input matrices out of a sequence of state and control inputs (Proctor, Brunton, and Kutz 2016). A least-squares solution can be found by solving the following optimization problem:

$$\mathbf{S}_k = \underset{\begin{bmatrix}\mathbf{A}_k & \mathbf{B}_k\end{bmatrix}}{\operatorname{argmin}} \left\| \widehat{\mathbf{Y}}_k - \begin{bmatrix}\mathbf{A}_k & \mathbf{B}_k\end{bmatrix}\widehat{\mathbf{Z}}_k \right\|_{\mathrm{F}} = \widehat{\mathbf{Y}}_k\widehat{\mathbf{Z}}_k^{\dagger} = \widehat{\mathbf{Y}}_k\widehat{\mathbf{Z}}_k^{\top}(\widehat{\mathbf{Z}}_k\widehat{\mathbf{Z}}_k^{\top})^{-1}, \tag{4}$$

where $||.||_{\mathrm{F}}$ is the Frobenius norm and $\widehat{\mathbf{Z}}_k^{\dagger} = \widehat{\mathbf{Z}}_k^{\top}(\widehat{\mathbf{Z}}_k\widehat{\mathbf{Z}}_k^{\top})^{-1}$ is the minimum-norm solution and the symbol † denotes pseudo-inverse. The employed data-driven method requires only snapshots of state and actuation data collected in the co-simulation framework. Having the inferred model in

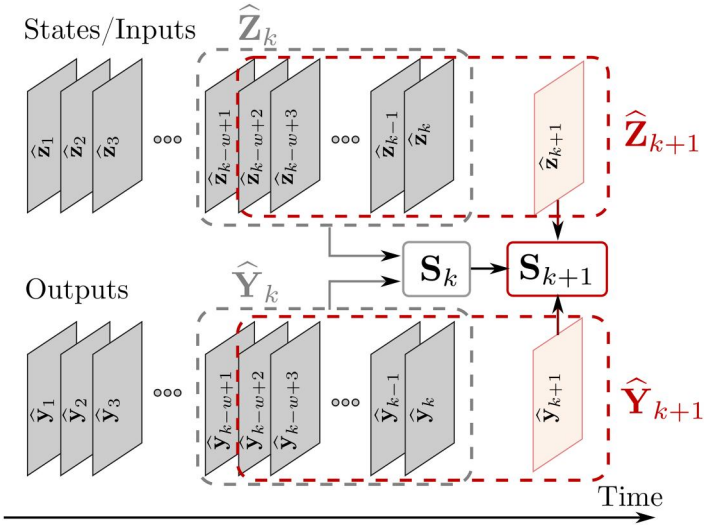


**Figure 4.** Recursive least-squares model discovery – windowed setup.

terms of $\mathbf{A}_k$ and $\mathbf{B}_k$ matrices helps to keep the numerical errors resulting from the co-simulation under control.

### 2.4. Recursive formulation

The algorithm for the inference of subsystem dynamics in the co-simulation framework can be implemented in the global form expressed in eq. (4). The idea is to slide a "window" containing only the most recent data measurements and recover subsystem dynamics in the form of $\mathbf{A}_k$ and $\mathbf{B}_k$ matrices. In this section, we discuss an alternative method (adapted from (Zhang et al. 2019)) that heavily exploits the recursive least-squares approach to reduce memory requirements in the solution process. Let us assume that at time $k$ we have access to past snapshot pairs $\left\{(\widehat{\mathbf{z}}_j, \widehat{\mathbf{y}}_j)\right\}_{j=k-w+1}^{k}$ organized in a manner shown in Fig. 4:

$$\widehat{\mathbf{Y}}_k = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{y}}_{k-w+1} & \widehat{\mathbf{y}}_{k-w+2} & \cdots \ \widehat{\mathbf{y}}_k \\ | & | & | \end{bmatrix}, \quad \widehat{\mathbf{Z}}_k = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{z}}_{k-w+1} & \widehat{\mathbf{z}}_{k-w+2} & \cdots \ \widehat{\mathbf{z}}_k \\ | & | & | \end{bmatrix}. \tag{5}$$

We consider minimizing the following cost function $J_k$ (cf. eq. (4)):

$$J_k(\mathbf{S}_k) = \sum_{i=k-w+1}^{k} ||\widehat{\mathbf{y}}_i - \mathbf{S}_k \widehat{\mathbf{z}}_i||^2 = ||\widehat{\mathbf{Y}}_k - \mathbf{S}_k \widehat{\mathbf{Z}}_k||_{\mathrm{F}}^2. \tag{6}$$

The objective function (6) can be extended in several ways (Pikuliński, Malczyk, and Aarts 2025), e.g., by adding appropriate penalty parameters that improve the numerical robustness and conditioning of the formulation or by performing sparsity-promoting regression on a library of carefully selected basis functions (Pikuliński and Malczyk 2024). The process of updating takes new information at time $k+1$ to generate a matrix $\mathbf{S}'_{k+1}$ that maps the dataset $\widehat{\mathbf{Z}}'_{k+1} = \begin{bmatrix} \widehat{\mathbf{Z}}_k & \widehat{\mathbf{z}}_{k+1} \end{bmatrix}$ into $\widehat{\mathbf{Y}}'_{k+1} = \begin{bmatrix} \widehat{\mathbf{Y}}_k & \widehat{\mathbf{y}}_{k+1} \end{bmatrix}$. The strategy starts with the observation that $\mathbf{S}_k$ in eq. (4) can be decomposed into a product of two matrices $\mathbf{Q}_k$ and $\mathbf{P}_k$ given by

$$\mathbf{Q}_k = \widehat{\mathbf{Y}}_k \widehat{\mathbf{Z}}_k^{\top}, \tag{7}$$

$$\mathbf{P}_k = (\widehat{\mathbf{Z}}_k \widehat{\mathbf{Z}}_k^{\top})^{-1}, \tag{8}$$

The matrix $\widehat{\mathbf{Z}}_k$ should be of full rank to guarantee the existence of $\mathbf{P}_k$. In this case $\mathbf{P}_k$ is symmetric and strictly positive definite. We wish to compute $\mathbf{S}'_{k+1} = \mathbf{Q}'_{k+1}\mathbf{P}'_{k+1}$. The quantities $\mathbf{Q}'_{k+1}$, $\mathbf{P}'_{k+1}$ correspond to the matrices $\mathbf{Q}_k$ and $\mathbf{P}_k$.

$$\mathbf{Q}'_{k+1} = \widehat{\mathbf{Y}}'_{k+1}\widehat{\mathbf{Z}}'^{\top}_{k+1} = \begin{bmatrix} \widehat{\mathbf{Y}}_k & \widehat{\mathbf{y}}_{k+1} \end{bmatrix} \cdot \begin{bmatrix} \widehat{\mathbf{Z}}_k & \widehat{\mathbf{z}}_{k+1} \end{bmatrix}^{\top} = \mathbf{Q}_k + \widehat{\mathbf{y}}_{k+1}\widehat{\mathbf{z}}_{k+1}^{\top}. \tag{9}$$

Similarly, we came up with the following formula:

$$\left(\mathbf{P}'_{k+1}\right)^{-1} = \widehat{\mathbf{Z}}'_{k+1}\widehat{\mathbf{Z}}'^{T}_{k+1} = \begin{bmatrix} \widehat{\mathbf{Z}}_k & \widehat{\mathbf{z}}_{k+1} \end{bmatrix} \cdot \begin{bmatrix} \widehat{\mathbf{Z}}_k & \widehat{\mathbf{z}}_{k+1} \end{bmatrix}^{T} = \mathbf{P}_k^{-1} + \widehat{\mathbf{z}}_{k+1}\widehat{\mathbf{z}}_{k+1}^{T}. \tag{10}$$

The formulas (9) and (10) allow to calculate a subsystem model $\mathbf{S}_{k+1}$ at time $k+1$ in the form:

$$\mathbf{S}_{k+1} = \mathbf{Q}'_{k+1}\mathbf{P}'_{k+1} = \left(\mathbf{Q}_k + \widehat{\mathbf{y}}_{k+1}\widehat{\mathbf{z}}_{k+1}^{\top}\right) \cdot \left(\mathbf{P}_k^{-1} + \widehat{\mathbf{z}}_{k+1}\widehat{\mathbf{z}}_{k+1}^{T}\right)^{-1}. \tag{11}$$

The recursive formula that updates $\mathbf{Q}_{k+1}$ from $\mathbf{Q}_k$ is directly available. Finding $\mathbf{P}_{k+1}$ from $\mathbf{P}_k$ requires computing the inverse directly. To do that, we employ the Sherman-Morrison matrix inversion formula (Golub and van Loan 2013) (i.e., $(\mathbf{A}+\mathbf{bc}^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{bc}^T\mathbf{A}^{-1}}{1+\mathbf{c}^T\mathbf{A}^{-1}\mathbf{b}}$ for outer product of two column vectors $\mathbf{b}$ and $\mathbf{c}$ and $1+\mathbf{c}^T\mathbf{Ab} \neq 0$), to get:

$$\mathbf{P}'_{k+1} = \left(\widehat{\mathbf{Z}}'_{k+1}\widehat{\mathbf{Z}}'^{\top}_{k+1}\right)^{-1} = \left(\mathbf{P}_k^{-1} + \widehat{\mathbf{z}}_{k+1}\widehat{\mathbf{z}}^{\top}_{k+1}\right)^{-1} = \mathbf{P}_k - \gamma'_{k+1}\mathbf{P}_k\widehat{\mathbf{z}}_{k+1}\widehat{\mathbf{z}}^{\top}_{k+1}\mathbf{P}_k, \tag{12}$$

where

$$\gamma'_{k+1} = \frac{1}{1+\widehat{\mathbf{z}}^T_{k+1}\mathbf{P}_k\widehat{\mathbf{z}}_{k+1}}. \tag{13}$$

Ultimately, the update formula can be viewed as:

$$\mathbf{S}'_{k+1} = \mathbf{S}_k + \gamma'_{k+1}\left(\widehat{\mathbf{y}}_{k+1} - \mathbf{S}_k\widehat{\mathbf{z}}_{k+1}\right)\widehat{\mathbf{z}}^{\top}_{k+1}\mathbf{P}_k. \tag{14}$$

The formula (14) shows the updated model $\mathbf{S}_{k+1}$ as a sum of the previous model $\mathbf{S}_k$ and a correction term. Now, we start the downdating process, which removes the oldest snapshot pair $(\widehat{\mathbf{y}}_{k-w+1}, \widehat{\mathbf{z}}_{k-w+1})$ from the sliding window of length $w$. At step $k+1$, we need to compute a matrix $\mathbf{S}_{k+1} = \mathbf{Q}_{k+1}\mathbf{P}_{k+1}$ that corresponds to the matrices $\mathbf{Q}'_{k+1}$ and $\mathbf{P}'_{k+1}$ evaluated in the updating step. Let us define the matrices based on scaled snapshots that exclude the oldest snapshot pair:

$$\widehat{\mathbf{Y}}_{k+1} = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{y}}_{k-w+2} & \widehat{\mathbf{y}}_{k-w+3} & \cdots\ \widehat{\mathbf{y}}_{k+1} \\ | & | & | \end{bmatrix}, \quad \widehat{\mathbf{Z}}_{k+1} = \begin{bmatrix} | & | & | \\ \widehat{\mathbf{z}}_{k-w+2} & \widehat{\mathbf{z}}_{k-w+2} & \cdots\ \widehat{\mathbf{z}}_{k+1} \\ | & | & | \end{bmatrix}. \tag{15}$$

Then, we should write the following relation for $\mathbf{Q}_{k+1}$:

$$\mathbf{Q}_{k+1} = \widehat{\mathbf{Y}}_{k+1}\widehat{\mathbf{Z}}^T_{k+1} = \mathbf{Q}'_{k+1} - \widehat{\mathbf{y}}_{k-w+1}\widehat{\mathbf{z}}^T_{k-w+1}. \tag{16}$$

The matrix $\mathbf{P}_{k+1}$ can be evaluated by using the mentioned Sherman-Morrison formula, to get:

$$\begin{aligned} \mathbf{P}_{k+1} &= \left(\widehat{\mathbf{Z}}_{k+1}\widehat{\mathbf{Z}}^T_{k+1}\right)^{-1} = \left((\mathbf{P}'_{k+1})^{-1} - \widehat{\mathbf{z}}_{k-w+1}\widehat{\mathbf{z}}^T_{k-w+1}\right)^{-1} \\ &= \mathbf{P}'_{k+1} - \gamma_{k+1}\mathbf{P}'_{k+1}\widehat{\mathbf{z}}_{k-w+1}\widehat{\mathbf{z}}^T_{k-w+1}\mathbf{P}'_{k+1}, \end{aligned} \tag{17}$$

and

$$\gamma_{k+1} = \frac{-1}{1-\widehat{\mathbf{z}}^T_{k-w+1}\mathbf{P}'_{k+1}\widehat{\mathbf{z}}_{k-w+1}}. \tag{18}$$

Finally, the derivation leads to the following update formula:

$$\mathbf{S}_{k+1} = \mathbf{S}'_{k+1} + \gamma_{k+1}(\widehat{\mathbf{y}}_{k+1} - \mathbf{S}'_{k+1}\widehat{\mathbf{z}}_{k+1})\widehat{\mathbf{z}}^T_{k+1}\mathbf{P}'_{k+1}. \tag{19}$$

An algorithmic summary of how the windowed version of DMD is used in co-simulation is provided below. Its application within the Jacobi scheme is illustrated in Fig. 5 and discussed in detail further.

1. Collect $w$ snapshot pairs $(\widehat{\mathbf{z}}_j, \widehat{\mathbf{y}}_j)$, $j = 1, \ldots, w$ (where window length is large enough; until then use, e.g., ZOH extrapolation) and $\widehat{\mathbf{Z}}_k$ defined in (5) is full row rank.
2. Initialize $\mathbf{P}_k$ and $\mathbf{S}_k$ in a batch processing mode (see Sec. 2.3).
3. When a new snapshot pair $(\widehat{\mathbf{z}}_{k+1}, \widehat{\mathbf{y}}_{k+1})$ becomes available execute the following steps:

a. Perform the updating step and use (9), (12) to calculate $\mathbf{Q}'_{k+1}$ and $\mathbf{P}'_{k+1}$.
b. Calculate the update $\mathbf{S}'_{k+1}$ by taking (14).

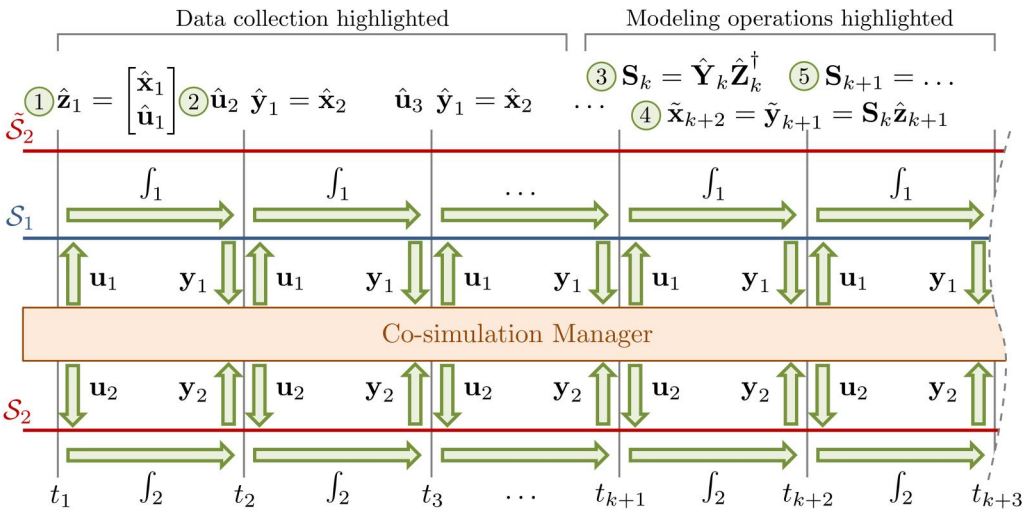


**Figure 5.** Data collection and modeling operations involved integrating two systems using the Jacobi scheme. Within the $\mathcal{S}_1$ system solver, the model of the $\mathcal{S}_2$ system is constructed ($\widetilde{\mathcal{S}}_2$).

c. If $w$ is increasing, $\mathbf{Q}_{k+1} = \mathbf{Q}'_{k+1}$, $\mathbf{P}_{k+1} = \mathbf{P}'_{k+1}$, $\mathbf{S}_{k+1} = \mathbf{S}'_{k+1}$ and skip to (3f).
d. Perform the downdating step and use (16), (17) to calculate $\mathbf{Q}_{k+1}$ and $\mathbf{P}_{k+1}$.
e. Calculate the update $\mathbf{S}_{k+1}$ by taking (19).
f. Predict $\widetilde{\mathbf{x}}_{k+3} = \widetilde{\mathbf{y}}_{k+2} \approx \mathbf{S}_{k+1}\widehat{\mathbf{z}}_{k+2}$, where $\widehat{\mathbf{z}}_{k+2} = \left[\widehat{\mathbf{x}}_{k+2}^{\mathrm{T}} \ \widehat{\mathbf{u}}_{k+2}^{\mathrm{T}}\right]^{\mathrm{T}}$.

Figure 5 illustrates the co-simulation based on the explicit Jacobi scheme involving two systems, $\mathcal{S}_1$ and $\mathcal{S}_2$. The co-simulation begins at time $t_1$, with subsequent time steps depicted by grey vertical lines to aid referencing. To briefly recap the integration scheme – at each step, the co-simulation manager sends inputs $(\mathbf{u}_1, \mathbf{u}_2)$ to the subsystems, which then perform local integration and return outputs ($\mathbf{y}_1$, $\mathbf{y}_2$) back to the manager as designed in the interface. We assume the presence of a direct feedthrough, such that $\mathbf{y}_1$ at $\forall_{j\in\mathbb{N}}\ t_j$ depends on the information $\mathbf{u}_1$ that is received at $t_{j+1}$ block.

Within subsystem $\mathcal{S}_1$, a windowed DMD is employed to infer the dynamics of $\mathcal{S}_2$ and predict $\mathbf{u}_1$ before sending back $\mathbf{y}_1$ – similar as in the proposed preliminary concept in Sec. 2.2 (Fig. 3). The inferred model of $\mathcal{S}_2$ is denoted by $\widetilde{\mathcal{S}}_2$, and DMD-related operations are highlighted in the upper section of the figure. Although data collection is explicitly shown only for the initial steps, it is performed throughout the whole co-simulation.

At time $t_1$, $\widehat{\mathbf{z}}_1$ is collected 1: $\widehat{\mathbf{x}}_1$ (representing $\mathbf{u}_1$) is received from the co-simulation manager, and $\widehat{\mathbf{u}}_1$ (initial input for $\mathcal{S}_2$ treated as $\mathbf{y}_1$) is assumed known. After local integration but before the communication point, $\mathbf{y}_1$ (corresponding to $\widehat{\mathbf{u}}_1$) is computed 2 – typically using a ZOH extrapolation of $\mathbf{u}_2$. Upon receiving $\mathbf{u}_1$ $(t_2)$, $\widehat{\mathbf{y}}_1 = \widehat{\mathbf{x}}_2$ becomes available, enabling also the construction of $\widehat{\mathbf{z}}_2$.

As per *Step 1* of the algorithm, pure data collection continues until $w$ snapshot pairs are gathered and the matrix $\widehat{\mathbf{Z}}_k$, defined in (5), becomes full row rank. Once this condition is met, the inferred model $\mathbf{S}_k$ is found following *Step 2* of the algorithm 3. From this point onward, only the modeling operations are highlighted in Fig. 5. Further, the computation of $\mathbf{y}_1$ may utilize the predicted $\mathbf{u}_1$, i.e., $\widetilde{\mathbf{x}}_{k+2}$ 4, found with the inferred model.

Upon receipt of the actual $\mathbf{u}_1$, a new data pair $(\widehat{\mathbf{z}}_{k+1},\ \widehat{\mathbf{y}}_{k+1})$ becomes available. The model $\mathbf{S}_k$ is then updated to $\mathbf{S}_{k+1}$ 5 as specified in *Steps 3(a) – 3(e)* of the algorithm. This enables the next prediction of $\mathbf{u}_1$ according to *Step 3(f)*.

## 3. Examples and results

This section demonstrates and discusses the results of implementing the proposed method for aiding co-simulation solution accuracy. We present three exemplary co-simulation problems, starting with a well-established co-simulation benchmark – oscillatory system (Sec. 3.1), which has already been introduced in Sec. 2.1. Then, we co-simulate a system containing a coupling between multi-degree-of-freedom parts in Sec. 3.2. Finally, a nonlinear system with a hydraulic part that requires the use of a multi-rate coupling scheme is presented in Sec. 3.3.

The examples have been implemented in Matlab, and all use a semi-implicit forward Euler integration formula. The co-simulation implementation follows the explicit Jacobi scheme, and its results are compared to a monolithic formulation counterpart. Each case utilizes the windowed version of the DMD algorithm to enhance the model's validity at a given step.

### 3.1. Two-mass oscillator

Our most basic test case for evaluating the proposed method is a double spring-mass oscillator introduced in Sec. 2.1. Two versions were used – a conventional one with linear characteristics, and another one with nonlinear springs. As mentioned, the direct feedthrough issue impacts the computation of the subsystems $\mathcal{S}_1{}'$s output, i.e., $\mathbf{y_1}$. This issue arises because the subsystem can only approximate the outputting coupling force at the end of a macro-step, $f^{\mathrm{c}}(t_{k+1})$, because the value of the spring displacement $x_2(t_{k+1})$, on which the coupling force directly depends, remains unknown at this stage.

To improve the approximation, we infer the dynamics model of the rest of the system from the perspective of subsystem $\mathcal{S}_1$. This model is built based on the interface variables, whose time history, coinciding with the co-simulation nomenclature, renders input-output characteristics. In other words, we develop a model that describes the dynamics of the system's part, where the input signal $\widehat{\mathbf{u}}$ is the output from $\mathcal{S}_1$ – $\mathbf{y_1}$, and the output $\widehat{\mathbf{x}}$ is the input sent from the manager to subsystem $\mathcal{S}_1$ – $\mathbf{u_1}$, see Fig. 3.

Examining the physical meaning of these coupling variables, where the input to the modeled system is the coupling force $f_1^{\mathrm{c}}$ and the output is the position $\xi_1^{\mathrm{c}}$ of the mass in subsystem $\mathcal{S}_2$, it is seen that, essentially, we build a model of subsystem $\mathcal{S}_2$ within $\mathcal{S}_1$, as presented in Fig. 3 from Sec. 2.3. It will not always be true, as the co-simulation manager might modify the coupling variables, i.e., $f_1^{\mathrm{c}} \neq f_2^{\mathrm{c}}$ and $\xi_1^{\mathrm{c}} \neq \xi_2^{\mathrm{c}}$. However, in this particular setting, we assume the manager passes the variables unmodified.

Following the above interpretation and derivations in Sec. 2.3, we build and update a linear model $\mathbf{S}_k^1$ such that

$$\begin{bmatrix} \xi_{k+1}^{\mathrm{c}} \\ \dot{\xi}_{k+1}^{\mathrm{c}} \end{bmatrix} = \mathbf{S}_k^1 \begin{bmatrix} \xi_k^{\mathrm{c}} \\ \dot{\xi}_k^{\mathrm{c}} \\ f_k^{\mathrm{c}} \end{bmatrix}, \tag{20}$$

where we change the subscripts to indicate the $k-$th step at which the measurements, model, and predictions are taken or valid. As presented in Sec. 2.4, the model matrix $\mathbf{S}_k^1$ is updated whenever a new snapshot pair is available. The current method assumes an arbitrary window length choice. For this example, we have chosen $w = 100$, which comes from a qualitative inspection of normalized mean squared error (NMSE) on positions of the subsystems for different window lengths when one of the springs is nonlinear. For an entirely linear case, the window length $w$ was found to have little to no impact on the quality of the results as far as its value verifies $w \geq 4$.

Before further interpretation, let us establish a definition of NMSE as follows

$$\mathrm{NMSE} = \frac{1}{s}\sum_{i=1}^{s}\sum_{j=1}^{l}\frac{\epsilon_{i,j}^2}{\sigma_j^2}, \tag{21}$$

where $s$ is the number of elements represented by position-level coordinates (for the discussed system, $s = 2$), $l$ is the number of prediction/measurement pairs used in the analysis, $\epsilon_{i,j}$ is the error between the monolithic and the co-simulated solutions for the $i$-th element at $j$-th step, and $\sigma_j$ is the standard deviation of the $j$-th position solved with the monolithic approach – the position to which the co-simulation should ideally converge.

In Fig. 6, the NMSE is presented as a function of the window length $w$. The error value decreases significantly up to a window length of $w = 100$. Beyond this point, changes in the error value become negligible, reflecting a general exponential decay characteristic of the error (illustrated by a red dashed line, which is an exponential function fitted to the observed results). However, the function is not strictly monotonic, and even relatively small window lengths (e.g., around 5) can yield reasonable results.

However, in practical applications, such an investigation could be cumbersome. Therefore, it is necessary to implement automatic criteria for deciding the window length. Although we do not include this extension in the proposed method, its implementation is straightforward due to the other procedure we use.

Initially, the proposed method gradually extends the window size $w$, starting from its minimal size $w = 2$, in a soft start procedure. This approach is based on the need to collect sufficient snapshots before using the full assumed window size (e.g., $w = 100$). If only the standard ZOH extrapolation is used until enough data is collected, the solution can diverge from the monolithic solution and may not converge once the DMD procedure begins refined predictions (see Fig. 7). Hence, the soft start method builds models even with only 2 data snapshots, gradually extending the window as new measurements arrive until the target window size is reached. Until these 2 data snapshots are collected, the ZOH extrapolation is used.

### 3.1.1. Linear oscillator

If all springs in the example are linear, the model inferred inside the $\mathcal{S}_1$ subsystem becomes an exact representation of $\mathcal{S}_2$. The dynamics of subsystem $\mathcal{S}_2$ can be described as follows

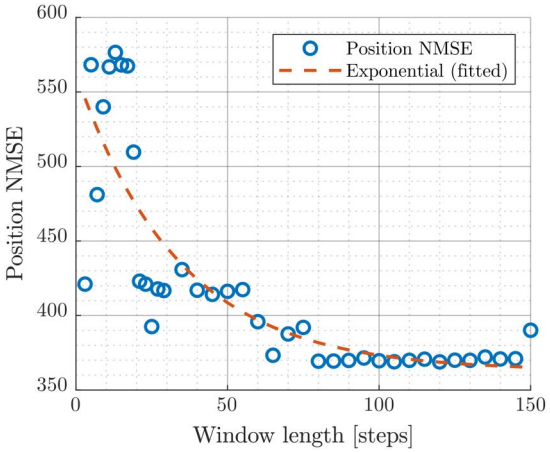


**Figure 6.** Normalized mean squared error (NMSE) of position as a function of the window length $w$ with an additional exponential curve fitted (nonlinear case).

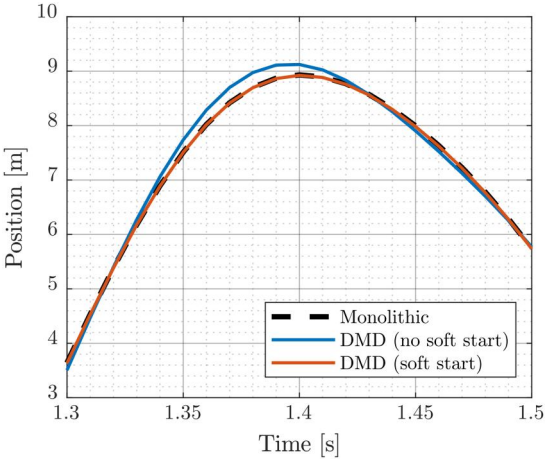


**Figure 7.** Comparison of the windowed DMD with and without soft start procedure implemented based on position of mass from the $\mathcal{S}_1$ subsystem (linear case).

**Table 1.** Summary of parameters related to deriving explicit formula for $\widetilde{\mathbf{S}}_2$ and their values used for the simulation of linear oscillator.

| Parameter | Value |
|---|---|
| $h$ | 0.01 s |
| $m_2$ | 1.00 kg |
| $k_2$ | 1000.00 $\frac{\text{N}}{\text{m}}$ |

$$m_2\ddot{\xi}_1^c = -f_1^c - k_2\xi_1^c, \tag{22}$$

where assumptions $\xi_1^c = \xi_2^c = \xi^c$ and $f_1^c = -f_2^c$ were used. Further, changing the subscripts into step-number based $k$ indices and inserting (22) into the integration scheme,

$$\begin{aligned} \xi_{k+1}^c &= \xi_k^c + h\dot{\xi}_{k+1}^c \\ \dot{\xi}_{k+1}^c &= \dot{\xi}_k^c + h\ddot{\xi}_k^c \end{aligned} \tag{23}$$

we can find an explicit formula for the model $\mathbf{S}_2$

$$\widetilde{\mathbf{S}}_2 = \begin{bmatrix} 1 - \dfrac{h^2k_2}{m_2} & h & -\dfrac{h^2}{m_2} \\ -\dfrac{hk_2}{m_2} & 1 & -\dfrac{h}{m_2} \end{bmatrix} = \begin{bmatrix} 0.9000 & 0.0100 & -0.0001 \\ -10.0000 & 1.0000 & -0.0100 \end{bmatrix}, \tag{24}$$

where values shown in Table 1 were used to compute its numerical form. As highlighted in the introductory sentence, the DMD procedure finds exactly the same values for $\mathbf{S}_2$ as in $\widetilde{\mathbf{S}}_2$. The exact representation is obtained after collecting 3 snapshot pairs. Since the model is linear and the measurements are noise-free, this aligns with the fact that the $\widehat{\mathbf{Z}}_3$ is full rank by then.

This compliance is also verifiable by comparing the energy preservation in the system (refer to Sec. 2.1 for details on the system's conservativeness). Figure 8 provides a detailed view of the time history of the entire system's energy between 1 s and 2 s. A broader perspective, covering the entire 10 s simulation, is presented in Fig. 9, showing the relative error of different co-simulation methods compared to the monolithic solution. When the proposed approach is used to predict the incoming value of the other part's position, the plot overlaps with that of the monolithic solution, whereas the co-simulation using ZOH diverges significantly. Such excellent

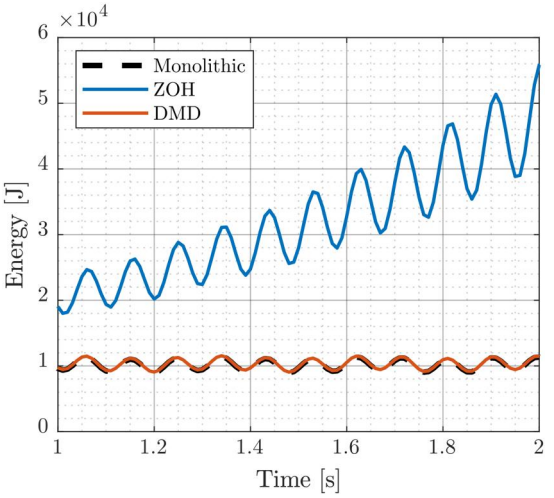


**Figure 8.** Energy time history in solutions from the monolithic solver, ZOH-based extrapolation co-simulation, and DMD-aided prediction co-simulation.

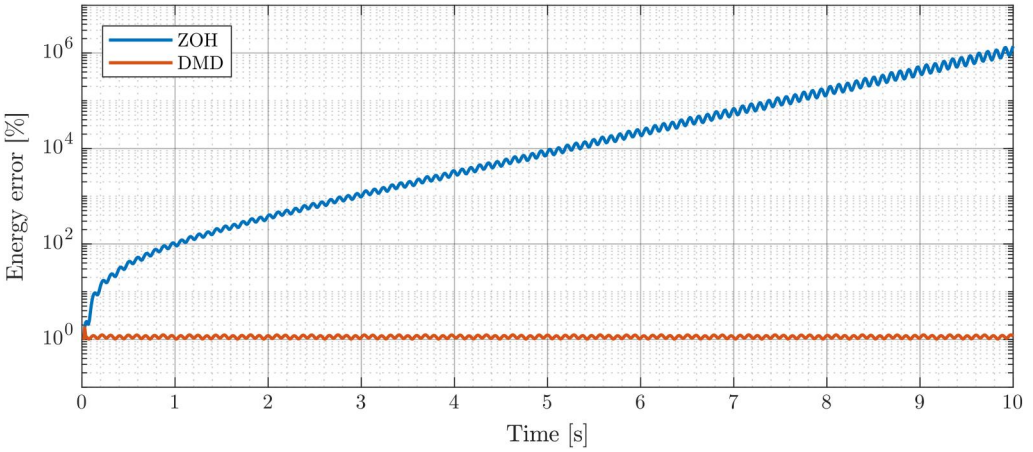


**Figure 9.** Relative energy error for ZOH-based extrapolation and DMD-aided prediction co-simulation, computed with respect to the monolithic solution (linear case).

compliance with the monolithic results implies that the proposed method can help avoid issues related to direct feedthrough.

Although the system is conservative and its energy should remain constant, some fluctuations around a specific value are visible in Fig. 8. However, these fluctuations are the same for both the DMD-aided co-simulation and the monolithic solutions. In fact, the oscillation of the energy level about its theoretically constant value is a consequence of the use of the symplectic Euler integration formula, which is common to both approaches, and does not stem from the co-simulation coupling. This can be easily verified by decreasing the integration step-size, which results in a similar reduction of the amplitude of the fluctuation of the energy level for both simulation strategies.

### *3.1.2. Nonlinear oscillator*

The DMD expectedly identified the rest of the system in the linear case. Examining how the approach performs when the identified part is nonlinear might help gain a complete

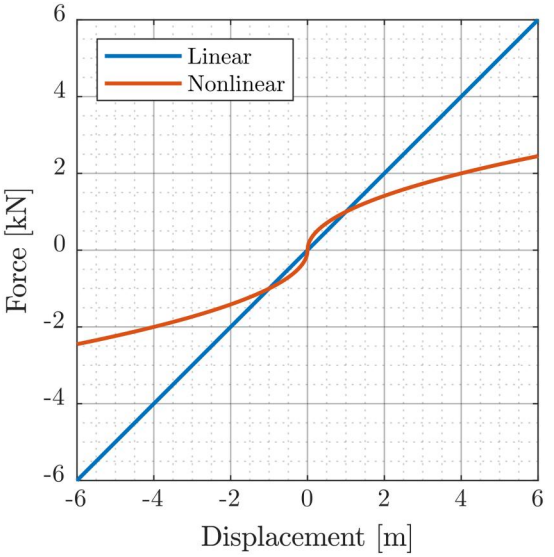


**Figure 10.** Different spring characteristics used in linear and nonlinear cases. The spring connecting mass $m_2$ with the wall is either linear or follows a square root.

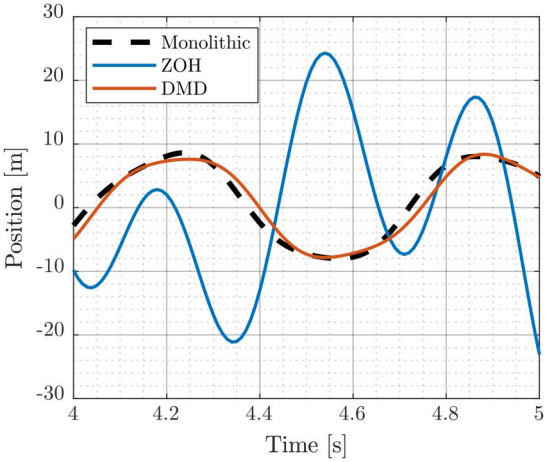


**Figure 11.** Comparison of ZOH and DMD-based co-simulation solutions in a nonlinear environment. The position history is plotted for subsystem $\mathcal{S}_1$.

understanding. Since we use a sliding-window approach, the method is anticipated to be still effective because the linearization point will continually move with the system's state. The main concern here is determining the optimal window length $w$, as the results vary based on this choice, as shown and discussed in Fig. 6.

To introduce nonlinearity, we change the characteristics of the spring connecting mass $m_2$ with the wall from linear to square root, as depicted in Fig. 10, which results in the following force formula

$$f_2 = k_2 \mathrm{sgn}\xi_2^c \sqrt{\left|\xi_2^c\right|}. \tag{25}$$

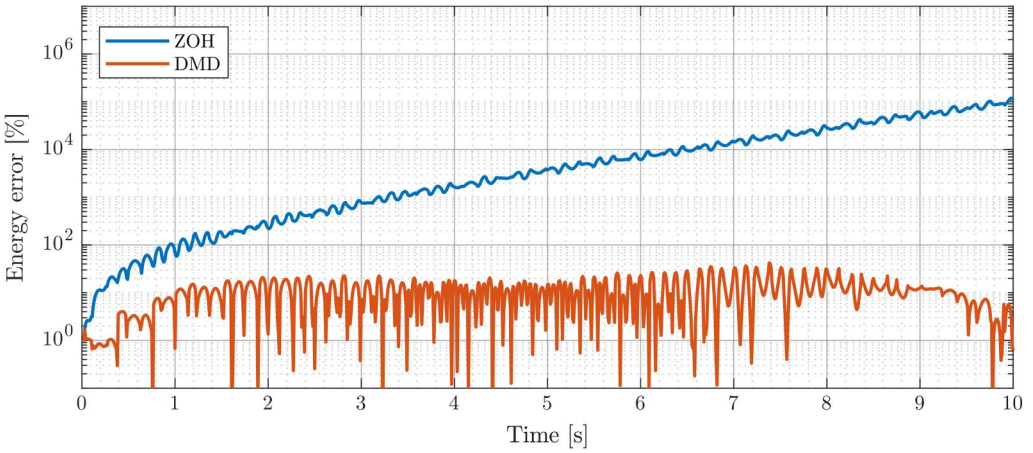


**Figure 12.** Relative energy error for ZOH-based extrapolation and DMD-aided prediction co-simulation, computed with respect to the monolithic solution (nonlinear case).

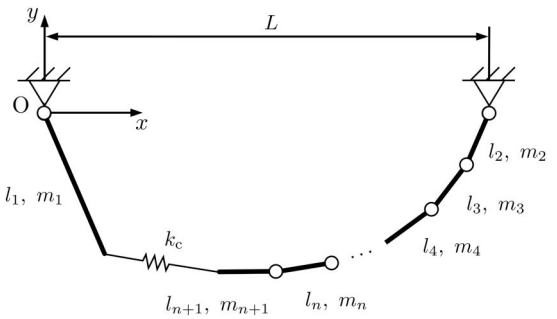


**Figure 13.** Scheme of multilink pendula example – single pendulum on the left connected by a spring with the tip of the multilink pendulum on the right.

Other parameters of the system and the co-simulation remain the same as in the linear case, i.e., we use a single-rate scheme with substeps and steps all equal to $h = 0.01$ s and the window length $w = 100$.

Figure 11 presents a zoom-in view of the time history of the simulated position of the first subsystem, $\mathcal{S}_1$, after significant differences between the solutions emerge. The results show that the DMD-aided co-simulation aligns with the monolithic solution more closely than the standard extrapolation approach. However, unlike in the linear case, there are still some discrepancies between the co-simulation and the monolithic solution. This observation could motivate further research, such as exploring an adaptive choice of window size.

Figure 12 presents a comparison analogous to that in Fig. 9, focusing on the relative energy error. The ZOH-based extrapolation approach exhibits characteristics similar to those of the linear case, with the error growing proportionally over time. Compared to the linear spring system, the DMD-based approach shows a larger error in the nonlinear case, as already seen in Fig. 11. However, the relative energy error remains less than 10% for most of the simulation time.

### 3.2. Multilink pendula

The two-mass oscillator introduced in Sec. 3.1 was divided into two subsystems, each with a single degree of freedom. To improve co-simulation accuracy, we employed a DMD framework to model subsystem $\mathcal{S}_2$ within $\mathcal{S}_1$. From the dimensionality perspective, the previous case relied on

**Table 2.** Values of the parameters of the multilink pendula system. The lengths are choosen such that the pendula's tips coincide in a horizontal position.

| Parameter | Value |
|---|---|
| $n$ | 10 |
| $L$ | 7.0 m |
| $l_1$ | 2.0 m |
| $l_2, \ldots, l_{n+1}$ | 0.5 m |
| $m_1$ | 1.00 kg |
| $m_2, \ldots, m_{n+1}$ | 0.10 kg |
| $k_c$ | 50.0 $\frac{\text{N}}{\text{m}}$ |

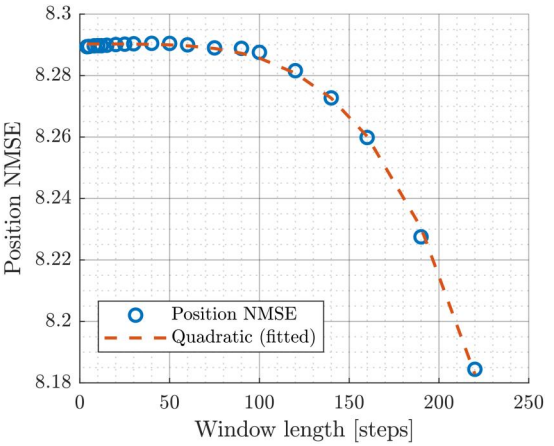


**Figure 14.** Normalized mean squared error (NMSE) of position as a function of the window length $w$ – minimum not found (quadratic curve fitted).

measurements from a one-dimensional subspace to model the dynamics of a single-degree-of-freedom body. A natural extension of this approach is to investigate how the proposed method performs when inferring a multi-degree-of-freedom model from data obtained through a coupling interface operating in a lower-dimensional subspace.

Therefore, in this example, we consider a system of two pendulums, as illustrated in Fig. 13. The left-side pendulum, with mass and length $m_1$, $l_1$, respectively, is a single pendulum, which along with a coupling linear spring constitutes subsystem $\mathcal{S}_1$. Subsystem $\mathcal{S}_2$ consists of a multi-link pendulum with $n$ links, each with mass $m_2, \ldots, m_{n+1}$ and length $l_2, \ldots, l_{n+1}$ (values attached in Table 2). Analogously to the two-mass oscillator, subsystem $\mathcal{S}_1$ evaluates the coupling force $\mathbf{f}_c$ at the interface, which is received as input by $\mathcal{S}_2$ ($x$, $y$ components). In return, $\mathcal{S}_2$ outputs the position $\xi_{n+1}$ and velocity $\dot{\xi}_{n+1}$ of the $n$-th link's tip ($x$ and $y$ coordinates) to the co-simulation manager.

This coupling introduces a direct feedthrough problem, the consequences of which we aim to minimize with the proposed approach. Within subsystem $\mathcal{S}_1$, we build a model of subsystem $\mathcal{S}_2$, maintaining the assumptions outlined in Sec. 3.1, where the co-simulation manager does not modify the coupling variables. In summary, based on the coupling force and the tip position and velocity, we aim to predict the tip's state at the next time step. The key challenge distinguishing this case is that subsystem $\mathcal{S}_1$ does not have full knowledge of the internal dynamics of $\mathcal{S}_2$. Instead, it only observes the pendulum tip's position through the coupling interface rather than the complete configuration of the multi-link pendulum.

We set the initial conditions for the simulation such that the tips of the pendula coincide in a horizontal position, with zero velocity and no initial spring elongation. Therefore, at the

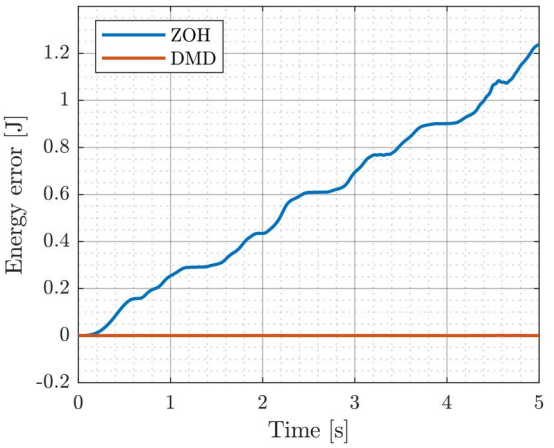


**Figure 15.** Energy error for ZOH-based extrapolation and DMD-aided prediction co-simulation, computed with respect to the monolithic solution.

beginning, only gravitational forces affect the system. The system is integrated using a constant time step of $h = 1$ ms. Figure 14 presents the NMSE of the position as a function of the sliding window length $w$ in the DMD-aided approach. Across the tested range of window lengths, there is no clear convergence to a minimum. However, based on the fitted quadratic curve, a general trend suggests that the error decreases as the window size increases. We have chosen a window size of $w = 100$ (Fig. 15).

Figure 17 compares the absolute error in total energy between the standard ZOH-based extrapolation and the DMD-aided prediction and modeling over a 5-s simulation. The ground truth for the error is the energy computed from the monolithic solution of this simulation, which remains constant – aside from numerical integration errors – since the investigated system is conservative. The results show that while the energy error in the ZOH-based solution increases linearly, the DMD-aided approach maintains a constant energy level.

Moderately surprisingly, despite subsystem $\mathcal{S}_2$ not being fully observed from the perspective of $\mathcal{S}_1$, the DMD framework still produces a useful model that attenuates discrepancies caused by the direct feedthrough. The results of this rigid-flexible example are further referred to in the overall discussion in Sec. 4.

### 3.3. Hydraulic manipulator

This example is a two-link manipulator driven by a hydraulic actuator, as depicted in Fig. 16. We divide this system into two subsystems: the mechanical subsystem $\mathcal{S}_1$, which consists of a link of length $l_1$ with a distributed mass $m$, and a massless link of length $l_2$ with point masses at its ends, mass $m_{\mathrm{Q}}$ at point Q and mass $m_{\mathrm{R}}$ at R in the figure. The second subsystem, $\mathcal{S}_2$, pertains to the hydraulics, involving a hydraulic actuator with a variable length $s$ coupled with $\mathcal{S}_1$ by a revolute joint at point P, as well as the other components in the hydraulic circuit, namely a pump, a fluid reservoir, and valves. The system works under gravity force $g$. A detailed description of the example and the hydraulics subsystem can be found in Peiret et al. (2018) and Naya et al. (2011), respectively. A monolithic formulation that combines hydraulics and dynamics can also be found in Naya et al. (2011).

We solve the co-simulation using a force-displacement scheme, where subsystem $\mathcal{S}_1$ outputs the length $s$ and rate $\dot{s}$ of the actuator, equivalent to position and velocity of point P, and subsystem $\mathcal{S}_2$ outputs the magnitude $f^{\mathrm{a}}$ of the force exerted at this point by the actuator.

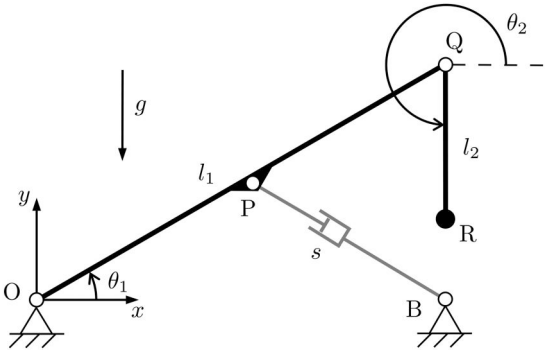


**Figure 16.** Scheme of the hydraulic manipulator consisting of two links with lengths $l_1$ and $l_2$, driven by an hydraulic actuator with variable length $s$.

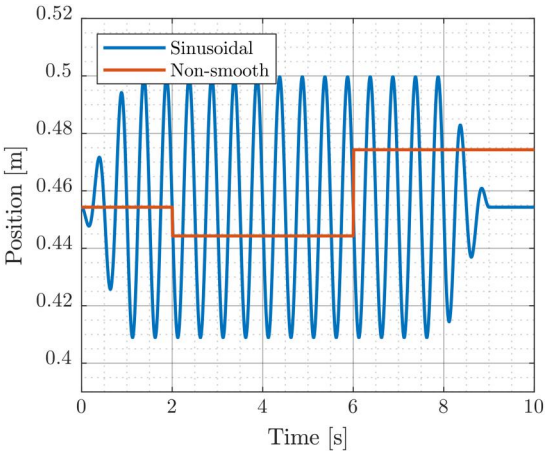


**Figure 17.** Control signals used for simulations – sinusoidal signal and non-smooth one. These represent spool displacement in a control valve of the actuator.

This choice, although convenient, raises the problem of direct feedthrough. Therefore, we propose to infer the dynamics of the rest of the system from the perspective of the hydraulic subsystem $\mathcal{S}_2$. In this context, it can also be interpreted as modeling the dynamics of subsystem $\mathcal{S}_1$ based on observable variables related to the coupling point P. The ultimate goal is to predict the next state $s_{k+1}$, $\dot{s}_{k+1}$ of point P based on its current state $s_k$, $\dot{s}_k$ and the force $f_k^{\mathrm{a}}$ applied by the actuator, with subscripts introduced to denote the step number. Further derivations of the system dynamics will more formally illustrate why this co-simulation case falls into a direct feedthrough issue.

The configuration of the system can be described by a set of independent generalized coordinates $\mathbf{q} = [\theta_1 \ \theta_2]^T$. However, for convenience, we describe the dynamics of this system using a dependent set of coordinates $\mathbf{x}$, which extends the independent set to $\mathbf{x} = [\theta_1 \ \theta_2 \ x_{\mathrm{P}} \ y_{\mathrm{P}}]^T$. This approach explicitly defines the coordinates of point P – $x_{\mathrm{P}}$ and $y_{\mathrm{P}}$.

The system is controlled by adjusting the position $\kappa$ of the spool in a hydraulic control valve. We discuss two control profiles – a sinusoidal one and a non-smooth one, as shown in Fig. 17. Both profiles start with a value of $\kappa_{t=0}$, which is chosen such that the force produced by the actuator keeps the system in static equilibrium for the initial configuration $\mathbf{q}_{t=0} = [\theta_{1,\ t=0} \ \theta_{2,\ t=0}]^T$.

The dynamics equations can be written as

$$\mathbf{M}\ddot{\mathbf{x}} = \mathbf{f} + \mathbf{f}^{\mathrm{a}} + \frac{\partial \mathbf{\Phi}}{\partial \mathbf{x}}^{T} \boldsymbol{\lambda}, \tag{26}$$

$$\mathbf{\Phi} = \begin{bmatrix} l_1 \cos\theta_1 - 2x_{\mathrm{P}} \\ l_2 \sin\theta_1 - 2y_{\mathrm{P}} \end{bmatrix} = \mathbf{0}, \tag{27}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{f}^{\mathrm{a}}$ represents forces exerted by the actuator, $\mathbf{f}$ contains the Coriolis, centrifugal and other forces present in the system, $\mathbf{\Phi}$ holds the constraint equations, and $\boldsymbol{\lambda}$ is the Lagrange multipliers term, also interpreted as the reaction forces. The state coordinates $s$ and $\dot{s}$, already introduced, can be found from

$$\begin{aligned} s &= \sqrt{(x_{\mathrm{P}} - x_{\mathrm{B}})^2 + (y_{\mathrm{P}} - y_{\mathrm{B}})^2}, \\ \dot{s} &= \begin{bmatrix} 0 & 0 & \dfrac{x_{\mathrm{P}} - x_{\mathrm{B}}}{s} & \dfrac{y_{\mathrm{P}} - y_{\mathrm{B}}}{s} \end{bmatrix} \dot{\mathbf{x}}, \end{aligned} \tag{28}$$

where the coordinates $x_{\mathrm{B}}$, $y_{\mathrm{B}}$ of the fixed point B are used.

The magnitude of the hydraulic force $f^{\mathrm{a}}$ follows the expression

$$f^{\mathrm{a}} = (p_2 - p_1)a_{\mathrm{p}} - c\dot{s}, \tag{29}$$

where $p_1$ and $p_2$ are fluid pressures within the cylinder, $a_{\mathrm{p}}$ is the total piston area and $c$ is a viscous friction coefficient representing internal dissipation effects in the actuator. We designate the length of the cylinder by $l^{\mathrm{a}}$ and let the piston divide it into chambers of length $l_1^{\mathrm{a}}$, $l_2^{\mathrm{a}}$, corresponding to the fluid pressures introduced and related to the total actuator length $s$ by

$$l_1^{\mathrm{a}} = 0.5 \cdot l^{\mathrm{a}} + s_{t=0} - s, \tag{30}$$

$$l_2^{\mathrm{a}} = 0.5 \cdot l^{\mathrm{a}} + s - s_{t=0}. \tag{31}$$

The dynamics of the hydraulic part of the system can then be described by the following set of equations

$$\dot{p}_1 = \frac{\beta_1}{a_{\mathrm{p}} l_1^{\mathrm{a}}} \left[ a_{\mathrm{p}}\dot{s}_1 + a_{\mathrm{i}} c_{\mathrm{d}} \sqrt{\frac{2(p_{\mathrm{P}} - p_1)}{\rho}} \delta_{\mathrm{P1}} - a_{\mathrm{o}} c_{\mathrm{d}} \sqrt{\frac{2(p_1 - p_{\mathrm{T}})}{\rho}} \delta_{\mathrm{T1}} \right], \tag{32}$$

$$\dot{p}_2 = \frac{\beta_2}{a_{\mathrm{p}} l_2^{\mathrm{a}}} \left[ -a_{\mathrm{p}}\dot{s}_1 + a_{\mathrm{o}} c_{\mathrm{d}} \sqrt{\frac{2(p_{\mathrm{P}} - p_2)}{\rho}} \delta_{\mathrm{P2}} - a_{\mathrm{i}} c_{\mathrm{d}} \sqrt{\frac{2(p_2 - p_{\mathrm{T}})}{\rho}} \delta_{\mathrm{T2}} \right], \tag{33}$$

where $a_{\mathrm{i}}$, $a_{\mathrm{o}}$ are the variable areas of input and output valves connecting to the pump and the tank, $c_{\mathrm{d}}$ is the discharge coefficient of the valves, $\rho$ is the fluid density, and $p_{\mathrm{P}}$ and $p_{\mathrm{T}}$ are the hydraulic pressure at the pump and the tank, respectively. Coefficients $\delta_{\mathrm{P1}}$, $\delta_{\mathrm{P2}}$, $\delta_{\mathrm{T1}}$, and $\delta_{\mathrm{T2}}$ are Heaviside step functions, that take as arguments the quantities inside the square roots that precede them. The bulk modulus $\beta_1$ and $\beta_2$, corresponding to each cylinder chamber, are a function of the fluid pressure, i.e.,

$$\beta_i = \frac{1 + ap_i + bp_i^2}{a + 2bp_i}, \quad i = 1, 2, \tag{34}$$

where $a$ and $b$ are constants for the fluid. Finally, the valve areas during operation are obtained

**Table 3.** Values of the parameters (par.) of the hydraulic system used in the simulations. Link $l_2$ is considered massless and connects to a point mass at point R.

| Par. | Value | Par. | Value |
|---|---|---|---|
| $l_1$ | 1.0 m | $a_p$ | $65 \cdot 10^{-4}$ m$^2$ |
| $l_2$ | 0.5 m | $l^a$ | 0.442 m |
| $m$ | 200 kg | $c$ | $10^5$ Ns/m |
| $m_Q$ | 250 kg | $c_d$ | 0.67 – |
| $m_R$ | 100 kg | $\rho$ | 850 kg/m$^3$ |
| $(x_B, y_B)$ | $(\sqrt{3}/2, 0)$ m | $a$ | $6.53 \cdot 10^{-10}$ Pa |
| $(\theta_1, \theta_2)$ | $(\pi/6, 3\pi/2)$ m | $b$ | $-1.19 \cdot 10^{-18}$ – |
| $(p_P, p_T)$ | (7.6, 0.1) MPa | $g$ | 9.81 m/s$^2$ |

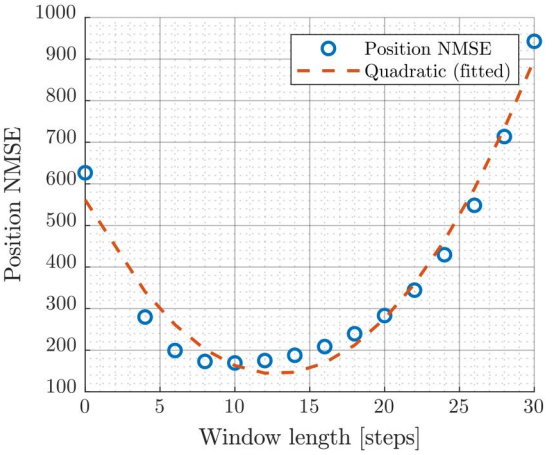


**Figure 18.** Normalized mean squared error (NMSE) of position as a function of the window length $w$ with an additional quadratic curve fitted.

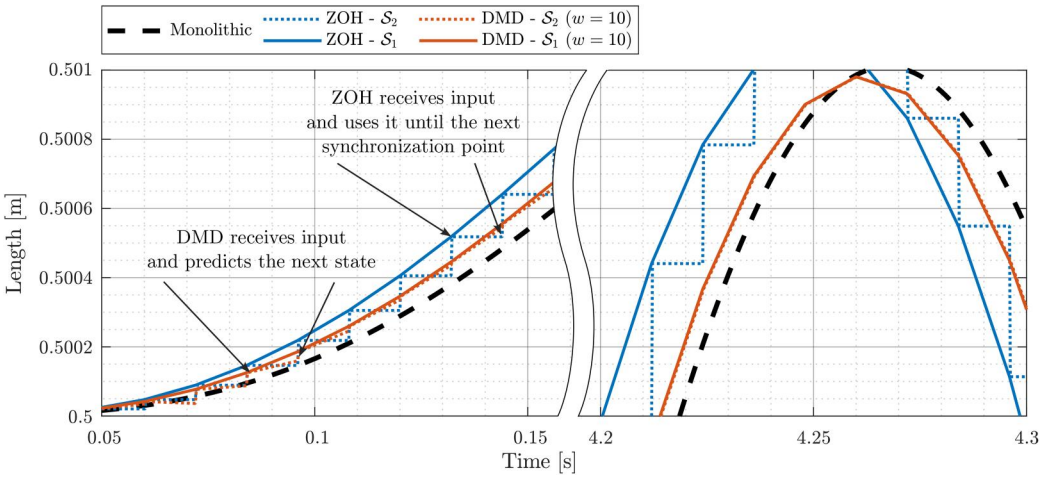


**Figure 19.** The length of the actuator $s$ as observed from different subsystems. As DMD predicts the state at the next communication point upon receiving new input, it is straightforward to interpolate the state between these points.

as a function of the control variable $\kappa$ as

$$a_i = a_f \ \kappa, \tag{35}$$

$$a_o = a_f(1 - \kappa). \tag{36}$$

where $a_f = 5 \cdot 10^{-4}$ m$^2$ is the valve area when fully open. Values of the parameters used in the simulations are attached in Table 3.

The introduction of hydraulic dynamics requires the use of smaller integration steps than those typically used for a mechanical system (Fig. 18). A multi-rate coupling scheme in the co-simulation is the natural choice in such a setting. Specifically, the mechanical subsystem $\mathcal{S}_1$ is solved with a macro step $h_m = 12$ ms by default, while the hydraulics-related subsystem $\mathcal{S}_2$ uses a step $h_h = 0.2$ ms. Without additional procedures in the co-simulation manager, new information about the coupling state $s$, $\dot{s}$ is delivered to the subsystem $\mathcal{S}_2$ every $h_m$. This negatively impacts the accuracy of the solution, as the hydraulic dynamics (29)–(33) depends on the state $s$, $\dot{s}$ and requires the use of outdated state values. This effect is illustrated in Fig. 19, where the dotted blue line represents the state seen from the $\mathcal{S}_2$ perspective using the basic ZOH approach.

This issue can be minimized by employing the proposed DMD-based approach. As new information about the state arrives, the inferred model inside $\mathcal{S}_2$ is updated to predict the next state $s_{k+1}$, $\dot{s}_{k+1}$ at the communication point. If information about the $\mathcal{S}_1$ macro-step is available to $\mathcal{S}_2$, it becomes straightforward to interpolate the state in consecutive $\mathcal{S}_2$ steps. The red lines in Fig. 19 present the results of the DMD-aided co-simulation. After 0.1 s (the time required to gather data for the entire assumed window size $w$), the predictions become accurate, and the interpolation (dotted line) qualitatively overlaps with what could have been computed from the perspective of $\mathcal{S}_1$ (solid line), i.e., having full knowledge about the mechanical system. Initially,

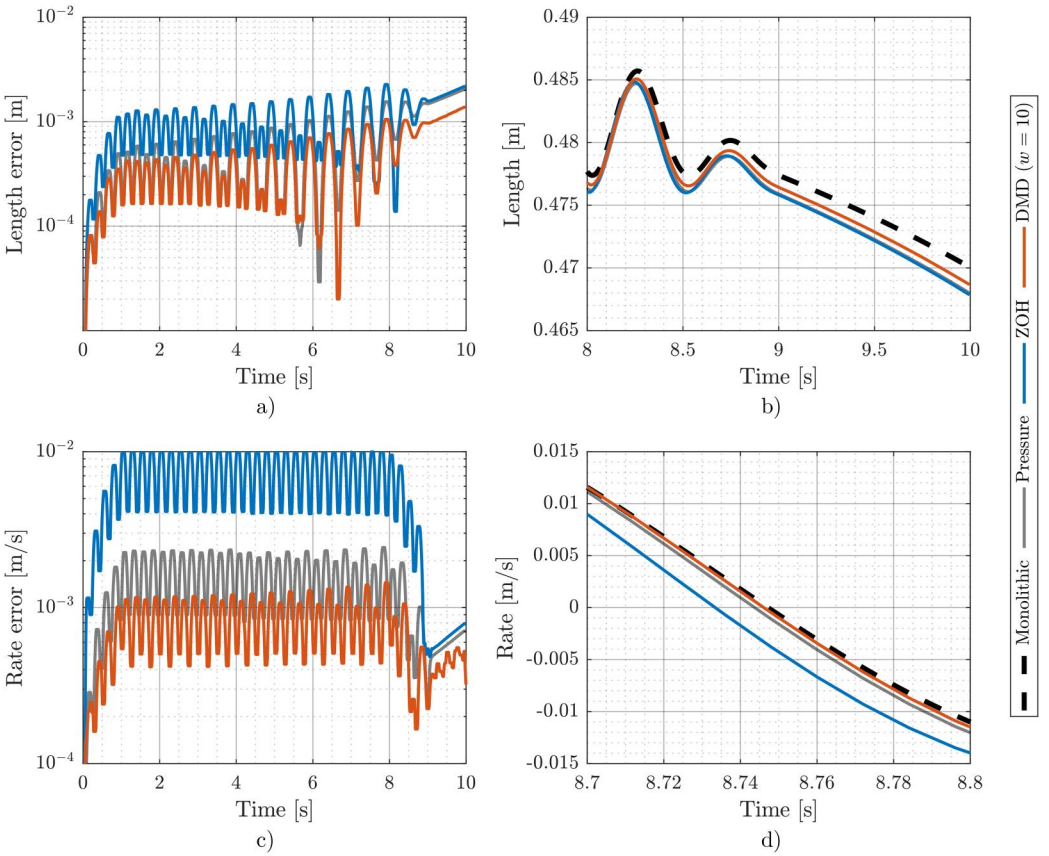


**Figure 20.** Simulation results for different approaches with a sinusoidal control signal and mechanical subsystem step size $h_m = 12$ ms. (a) Actuator length error with respect to (w.r.t.) the monolithic solution; (b) length time history; (c) actuator rate error w.r.t. the monolithic solution; (d) rate time history.

discontinuous changes in the dotted red line are observed, related to the not-yet-accurate predictions.

### 3.3.1. Sinusoidal control signal

Similarly to the two-mass oscillator example, we searched for the optimal window length when applying a sinusoidal control law to the system. The results, shown in Fig. 18, exhibit quadratic characteristics and indicate that a window length of $w = 10$ is a reasonable choice. The method proposed in this work and the window choice decrease the error-index NMSE from 627 to 169.

The sinusoidal control signal, depicted in Fig. 17, linearly increases its amplitude from 0 to a deviation of 10% from the static equilibrium signal value $\kappa_{t=0}$ after 1 s. The amplitude then remains constant until the 9-th s, after which it linearly decreases to 0 amplitude in 1 s.

In Fig. 20, we present the results of 10-s simulations using different approaches. In addition to the monolithic, ZOH, and DMD solutions, we introduce a method referred to as *pressure*. This approach is based on the observation that the negative impact of direct feedthrough can be reduced by changing the force coupling variable $f^{\mathrm{a}}$ to the chambers' pressures $p_1$, $p_2$. Subsequently, the force value is calculated in $\mathcal{S}_1$ according to (29). As seen in Fig. 20(c), the error associated with the pressure approach is significantly lower than that of the ZOH method. The pressure method allows for more accurate computation of the internal dissipation effects in the actuator, which constitute an important part of the force $f^{\mathrm{a}}$.

However, the DMD approach outperforms the pressure method, exhibiting even smaller errors in both the rate $\dot{s}$ and length $s$ of the actuator. When examining Fig. 20(b), the pressure approach does not differ significantly from the ZOH method. Additionally, while there is still a gap between the monolithic and DMD approaches, the DMD method demonstrates noticeable improvement. Notably, the rate error remains approximately constant during periods without changes in the control sinusoid amplitude and changes only during transient stages.

### 3.3.2. Non-smooth control signal

The discontinuous control signal presented in Fig. 17 will induce non-smoothness in other variables dependent on its value. Using DMD for such signals presents a significant challenge. The theoretical foundation and practical applications of DMD for non-smooth signals are not well-studied, making its effectiveness in such scenarios uncertain. However, signals with such discontinuities are common in real-life applications, making it important to validate the method's performance in these settings as well.

Analogous to previous examples, we tested different window sizes, leading to the expected conclusion that a longer window (e.g., $w = 10$) is not optimal for such a non-smooth signal. Models built with data from longer windows still capture a significant portion of the dynamics linearized around the state before the discontinuous event. We decided to use a window size of $w = 4$. However, for clarity, results for a window $w = 10$ are also included in Fig. 21 and Fig. 22.

The challenge with the multi-rate co-simulation is that system $\mathcal{S}_2$ uses outdated information during the intervals between communication from $\mathcal{S}_1$. Therefore, in addition to the default time step of $h_{\mathrm{m}} = 12$ ms, for which results are shown in Fig. 21, we also tested the impact of a longer time step $h'_{\mathrm{m}} = 20$ ms, with results presented in Fig. 22. In both figures, the error starts at the $3^{\mathrm{rd}}$ s, as the system remains in static equilibrium for the first 2s.

In all presented configurations, the DMD approach demonstrated a less oscillatory response to discontinuous events, exhibiting improvements in both settling time and amplitude. For a time step $h'_{\mathrm{m}} = 20$ ms, the ZOH approach became unstable, whereas the DMD approach yielded results that were close to those of the monolithic method - however, the pressure interface achieved even smaller errors. Generally, a shorter window $w = 4$ resulted in smaller errors than a

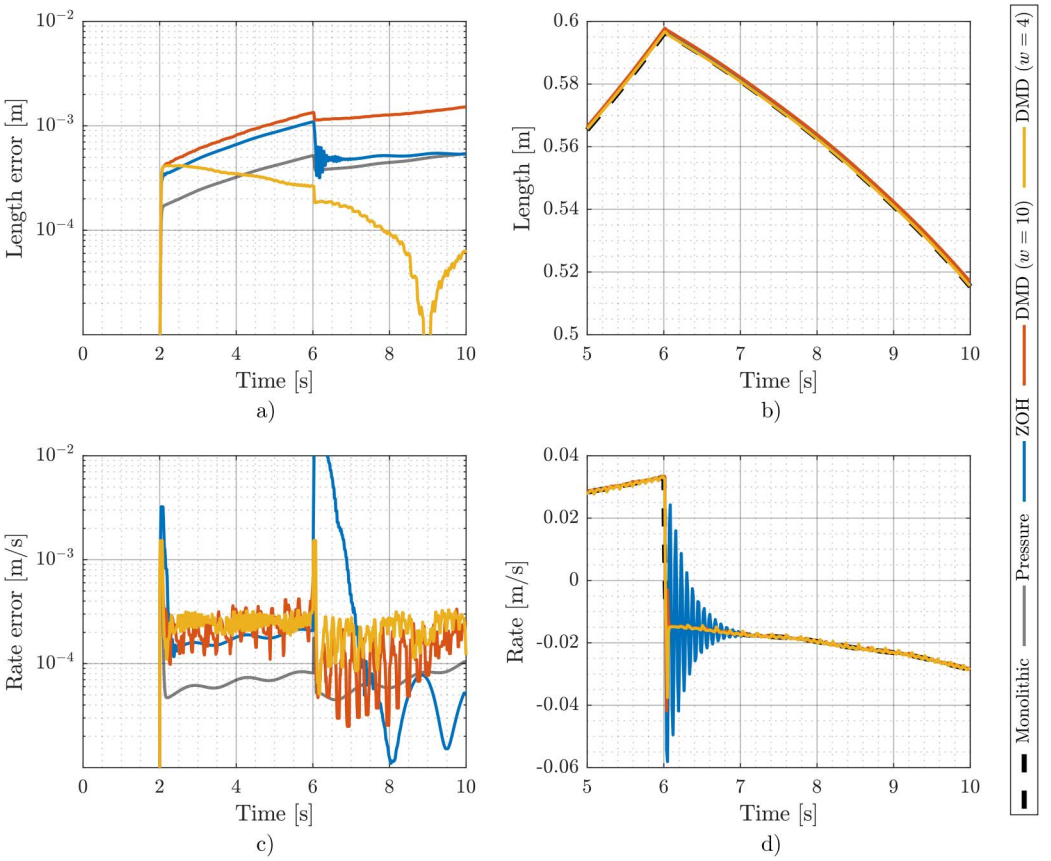


**Figure 21.** Simulation results for different approaches with a non-smooth control signal and mechanical subsystem step size $h_m = 12$ ms. (a) Actuator length error with respect to (w.r.t.) the monolithic solution; (b) length time history; (c) actuator rate error w.r.t. the monolithic solution; (d) rate time history.

longer window $w = 10$. As seen in Fig. 21(a), which shows the error in actuator length, the shorter window size outperformed the pressure method.

## 4. Discussion

The applicability of the presented method depends on the feasibility of determining the external subsystem dynamics from the numerical values of the coupling variables exchanged at the co-simulation interface. Ordinarily, coupling variables do not convey all the information that would enable one to reconstruct the dynamics of a subsystem. This was illustrated in Peiret et al. (2018) by means of the reduced interface model (RIM) concept in the co-simulation of mechanical systems. Let us assume that the dynamics of the external, inferred subsystem $\mathcal{S}_2$ in Fig. 3 is formulated through a set of $n$ independent velocities $\mathbf{v}$ as

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{f} + \mathbf{f}_i \tag{37}$$

where $\mathbf{M}$ is the $n \times n$ mass matrix, $\mathbf{f}$ is the generalized forces term, which includes velocity-dependent inertial forces, and $\mathbf{f}_i$ denotes the interaction forces between subsystems at the coupling interface. These interaction forces act in a subspace that can be parameterized by means of a set of $p$ velocities $\mathbf{w}_i$,

$$\mathbf{w}_i = \mathbf{A}_i\mathbf{v} \tag{38}$$

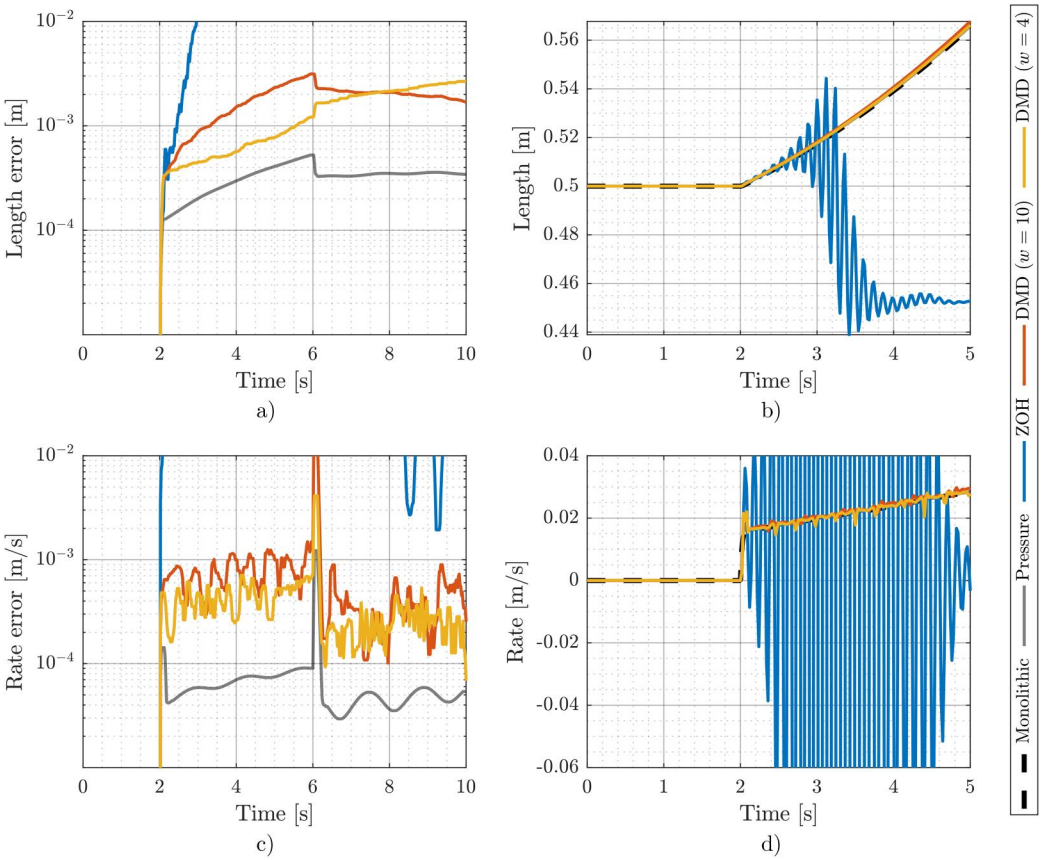


**Figure 22.** Simulation results for different approaches with a sinusoidal control signal and mechanical subsystem step size $h_{\mathrm{m}}' = 20$ ms. (a) Actuator length error with respect to (w.r.t.) the monolithic solution; (b) length time history; (c) actuator rate error w.r.t. the monolithic solution; (d) rate time history.

where $\mathbf{A}_{\mathrm{i}}$ is a $p \times n$ Jacobian matrix. It was shown in Peiret et al. (2018) that the dynamics of subsystem $\mathcal{S}_2$ in the interface subspace can be described by an effective mass matrix $\widetilde{\mathbf{M}}$ and an effective force term $\widetilde{\mathbf{f}}$, whose expressions are

$$\widetilde{\mathbf{M}} = \left(\mathbf{A}_{\mathrm{i}}\mathbf{M}^{-1}\mathbf{A}_{\mathrm{i}}^{\mathrm{T}}\right)^{-1}; \quad \widetilde{\mathbf{f}} = \left(\mathbf{A}_{\mathrm{i}}\mathbf{M}^{-1}\mathbf{A}_{\mathrm{i}}^{\mathrm{T}}\right)^{-1}\left(\mathbf{A}_{\mathrm{i}}\mathbf{M}^{-1}\mathbf{f} + \dot{\mathbf{A}}_{\mathrm{i}}\mathbf{v}\right) \tag{39}$$

An extension of the method for systems modeled using dependent velocities can also be found in Peiret et al. (2018). This dynamics representation is conceptually similar to the compound-body dynamics required by some recursive multibody methods like the divide and conquer algorithm (DCA) (Malczyk et al. 2019).

In practice, the interaction between subsystems $\mathcal{S}_1$ and $\mathcal{S}_2$ takes place through the dynamics defined by the terms in eq. (39). Because the size $n$ of subsystem $\mathcal{S}_2$ is often larger than the interface dimension $p$, the dynamics representation given by the effective terms $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{f}}$ does not allow one to reconstruct the full dynamics of $\mathcal{S}_2$. The dynamics inference presented in Sec. 2, therefore, can be used to predict the evolution of $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{f}}$, but cannot explicitly reach the internal, hidden states of $\mathcal{S}_2$. This limitation is not a particular feature of the data-driven method here presented, but is inherent to every explicit co-simulation interface.

When the configuration of subsystem $\mathcal{S}_2$ is subjected to sudden changes, as in the case of impacts, the terms in eq. (38) can undergo quick variations. This led to the need to formulate nonsmooth RIMs to address such changes (Peiret, González, et al. 2020). It must be noted that

establishing such reduced models requires a full knowledge of the dynamics of the external subsystem subjected to impacts. The algorithm presented in Sec. 2 relies on information from previous communication steps to build its description of the external subsystem. Accordingly, a sudden modification of the dynamics might require restarting the algorithm procedures and compromise the accuracy of the predictions.

The algorithm, however, can show a robust behavior with respect to changes in the hidden states in subsystem $\mathcal{S}_2$, as long as their evolution does not cause a variation in $\widetilde{\mathbf{M}}$ and $\widetilde{\mathbf{f}}$ that cannot be captured by the communication rate at the coupling interface. This is confirmed by the results presented in Sec. 3.2, which show that the knowledge of the internal states of the multi-link pendulum is not required to develop a data-driven approximation of the interface terms able to mitigate coupling errors.

## 5. Conclusion

This work presents a data-driven method to infer subsystem dynamics in explicit co-simulation environments, in which the limited, discrete-time exchange of information at the coupling interface often results in degraded accuracy of the numerical integration. In extreme cases, this problem may lead to instability, especially if one or more subsystems are subjected to direct feedthrough. Instead of relying on signal reconstruction approaches to predict the evolution of subsystem inputs and avoid discontinuities at the interface, a solution commonly used in co-simulation applications, here we put forward a method to provide a given subsystem with a physics-based prediction of the evolution of its inputs. The proposed strategy does not require information about the internals of other subsystems in the co-simulation environment and uses only the data conveyed by the coupling variables exchanged through the interface. These data are processed by means of Dynamic Mode Decomposition (DMD) to arrive at a data-based representation of the dynamics of the external subsystem. Such a representation can be then used to predict the evolution of the inputs received in future communication points, enabling a more accurate evaluation of subsystem output values, which in turn improves the accuracy of the overall numerical integration.

The proposed data-driven method was tested in the co-simulation of popular benchmark problems in the literature, including purely mechanical systems such as linear and nonlinear oscillators, and a multiphysics example, namely a hydraulically actuated two-link manipulator, co-simulated in a multi-rate fashion. Results confirmed that the prediction of future inputs, provided by the proposed method, in subsystems subjected to direct feedthrough improved the accuracy of the co-simulation, bringing the results closer to the ones predicted by the theory and delivered by monolithic simulation runs. The method was also able to stabilize co-simulation problems that became unstable if constant input extrapolation was used instead.

The ability of the method to correctly predict future input values depends on the relation between the rate of change of the subsystem dynamics and the update of information at the coupling interface. DMD relies on previous values to infer subsystem dynamics, which means that sudden variations in subsystem motion, such as those caused by impacts, may deteriorate the ability of the algorithm to deliver meaningful predictions. For systems with smooth dynamics, however, the algorithm is able to correct interface discontinuities, even in cases in which the number of coupling variables is smaller than the degrees of freedom of the external subsystem. This was confirmed through the simulation of the dynamics of a multi-link mechanical system.

Future work will focus on automatizing the selection of method parameters, such as the window size, and studying its applicability to more general co-simulation scenarios.

## Acknowledgment

Authors from Universidade da Coruña acknowledge the support of project PID2022-139832NB-I00 funded by MICIU/AEI/10.13039/ 501100011033 and ERDF, EU, and grant ED431C 2023/01 from the Government of Galicia.

## Disclosure statement

No potential conflict of interest was reported by the author(s).

## Funding

This work was supported by Ministry of Science and Innovation of Spain; Politechnika Warszawska. Research was funded by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) programme.

## References

Anitescu, M., and A. Tasora. 2010. "An Iterative Approach for Cone Complementarity Problems for Nonsmooth Dynamics." *Computational Optimization and Applications* 47 (2): 207–235. https://doi.org/10.1007/s10589-008-9223-4

Arnold, M., C. Clauss, and T. Schierz. 2013. "Error Analysis and Error Estimates for co-Simulation in FMI for Model Exchange and co-Simulation V2.0." *Archive of Mechanical Engineering* 60 (1): 75–94. https://doi.org/10.2478/meceng-2013-0005

Bauchau, O. A. 2011. *Flexible Multibody Dynamics*. Springer Dordrecht. https://doi.org/10.1007/978-94-007-0335-3

Ben Khaled-El Feki, A., L. Duval, C. Faure, D. Simon, and M. Ben Gaid. 2017. "CHOPtrey: Contextual Online Polynomial Extrapolation for Enhanced Multi-Core co-Simulation of Complex Systems." *Simulation* 93 (3): 185–200. https://doi.org/10.1177/0037549716684026

Benedikt, M., D. Watzenig, J. Zehetner, and A. Hofer. 2013. "A Nearly Energy-Preserving Coupling Element for Holistic Weak-Coupled System co-Simulations." In: NAFEMS World Congress 2013. Salzburg, Austria

Berger, E., M. Sastuba, D. Vogt, B. Jung, and H. B. Amor. 2015. "Estimation of Perturbations in Robotic Behavior Using Dynamic Mode Decomposition." *Advanced Robotics* 29 (5): 331–343. https://doi.org/10.1080/01691864.2014.981292

Bruder, D., X. Fu, R. B. Gillespie, C. D. Remy, and R. Vasudevan. 2021. "Data-Driven Control of Soft Robots Using Koopman Operator Theory." *IEEE Transactions on Robotics* 37 (3): 948–961. https://doi.org/10.1109/TRO.2020.3038693

Chen, W., S. Ran, C. Wu, and B. Jacobson. 2021. "Explicit Parallel co-Simulation Approach: Analysis and Improved Coupling Method Based on H-Infinity Synthesis." *Multibody System Dynamics* 52 (3): 255–279. https://doi.org/10.1007/s11044-021-09785-x

Dai, X., A. Raoofian, J. Kövecses, and M. Teichmann. 2024. "Model-Based co-Simulation of Flexible Mechanical Systems with Contacts Using Reduced Interface Models." *IEEE Robotics and Automation Letters* 9 (1): 239–246. https://doi.org/10.1109/LRA.2023.3332501

Eguillon, Y., B. Lacabanne, and D. Tromeur-Dervout. 2022. "F3ORNITS: A Flexible Variable Step Size Non-Iterative co-Simulation Method Handling Subsystems with Hybrid Advanced Capabilities." *Engineering with Computers* 38 (5): 4501–4543. https://doi.org/10.1007/s00366-022-01610-z

Flores, P., J. Ambrósio, and H. M. Lankarani. 2023. "Contact-Impact Events with Friction in Multibody Dynamics: Back to Basics." *Mechanism and Machine Theory* 184 (105): 105305. https://doi.org/10.1016/j.mechmachtheory.2023.105305

Folkestad, C., D. Pastor, and J. W. Burdick. 2020. "Episodic Koopman Learning of Nonlinear Robot Dynamics with Application to Fast Multirotor Landing." In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 9216–9222. https://doi.org/10.1109/ICRA40945.2020.9197510

Gerstmayr, J. 2024. "Exudyn – a C++-Based Python Package for Flexible Multibody Systems." *Multibody System Dynamics* 60 (4): 533–561. https://doi.org/10.1007/s11044-023-09937-1

Glumac, S., N. Varga, F. Raos, and Z. Kovačić. 2022. "Co-Simulation Perspective on Evaluating the Simulation with the Engine Test Bench in the Loop." *Automatika* 63 (2): 275–287. https://doi.org/10.1080/00051144.2022.2031537

Golub, G. H., and C. F. van Loan. 2013. *Matrix Computations*. 4th ed. JHU Press, Baltimore

González, F., M. Á. Naya, A. Luaces, and M. González. 2011. "On the Effect of Multirate co-Simulation Techniques in the Efficiency and Accuracy of Multibody System Dynamics." *Multibody System Dynamics* 25 (4): 461–483. https://doi.org/10.1007/s11044-010-9234-7

Hu, W., Z. Zhou, S. Chandler, D. Apostolopoulos, K. Kamrin, R. Serban, and D. Negrut. 2022. "Traction Control Design for off-Road Mobility Using an SPH-DAE Cosimulation Framework." *Multibody System Dynamics* 55 (1-2): 165–188. https://doi.org/10.1007/s11044-022-09815-2

Kaiser, E., J. N. Kutz, and S. L. Brunton. 2021. "Data-Driven Discovery of Koopman Eigenfunctions for Control." *Machine Learning: Science and Technology* 2 (3): 035023. https://doi.org/10.1088/2632-2153/abf0f5

Kübler, R., and W. Schiehlen. 2000. "Modular Simulation in Multibody System Dynamics." *Multibody System Dynamics* 4 (2-3): 107–127. https://doi.org/10.1023/A:1009810318420

López Varela, Á., D. Dopico Dopico, and A. Luaces Fernández. 2023. "Augmented Lagrangian Index-3 Semi-Recursive Formulations with Projections. Direct Sensitivity Analysis." *Multibody System Dynamics* 61 (2): 195–231. https://doi.org/10.1007/s11044-023-09928-2

Maciąg, Paweł, Paweł Malczyk, and Janusz Frączek. 2020. "Hamiltonian Direct Differentiation and Adjoint Approaches for Multibody System Sensitivity Analysis." *International Journal for Numerical Methods in Engineering* 121 (22): 5082–5100. https://doi.org/10.1002/nme.6512

Malczyk, Paweł, Janusz Frączek, Francisco González, and Javier Cuadrado. 2019. "Index-3 Divide-and-Conquer Algorithm for Efficient Multibody System Dynamics Simulations: Theory and Parallel Implementation." *Nonlinear Dynamics* 95 (1): 727–747. https://doi.org/10.1007/s11071-018-4593-3

Naya, M., J. Cuadrado, D. Dopico, and U. Lugris. 2011. "An Efficient Unified Method for the Combined Simulation of Multibody and Hydraulic Dynamics: Comparison with Simplified and co-Integration Approaches." *Archive of Mechanical Engineering* 58 (2): 223–243. https://doi.org/10.2478/v10180-011-0016-4

Nopour, R., A. Taghvaeipour, M. M. Aghdam, and F. González. 2024. "On the Generalized Bézier-Based Integration Approach for co-Simulation Applications." *Mechanics Based Design of Structures and Machines* 52 (7): 4759–4790. https://doi.org/10.1080/15397734.2023.2239349

Olivier, B., O. Verlinden, and G. Kouroussis. 2022. "Comparison of X–T and X–X co-Simulation Techniques Applied on Railway Dynamics." *Multibody System Dynamics* 55 (1-2): 39–56. https://doi.org/10.1007/s11044-022-09821-4

Peiret, A., F. González, J. Kövecses, and M. Teichmann. 2018. "Multibody System Dynamics Interface Modelling for Stable Multirate co-Simulation of Multiphysics Systems." *Mechanism and Machine Theory* 127: 52–72. https://doi.org/10.1016/j.mechmachtheory.2018.04.016

Peiret, A., F. González, J. Kövecses, and M. Teichmann. 2020. "Co-Simulation of Multibody Systems with Contact Using Reduced Interface Models." *Journal of Computational and Nonlinear Dynamics* 15 (4): 041. 001–14 https://doi.org/10.1115/1.4046052

Peiret, A., F. Gonzalez, J. Kovecses, M. Teichmann, and A. Enzenhoefer. 2020. "Model-Based Coupling for co-Simulation of Robotic Contact Tasks." *IEEE Robotics and Automation Letters* 5 (4): 5756–5763. https://doi.org/10.1109/LRA.2020.3010204

Pikuliński, M., and P. Malczyk. 2021. "Adjoint Method for Optimal Control of Multibody Systems in the Hamiltonian Setting." *Mechanism and Machine Theory* 166: 104473. https://doi.org/10.1016/j.mechmachtheory.2021.104473

Pikuliński, M., and P. Malczyk. 2024. "Optimal Control of Open–Loop Multibody Systems Recovered from Data.". In *Optimal Design and Control of Multibody Systems*, edited by K. Nachbagauer, A. Held, 99–109. Cham: Springer Nature Switzerland.

Pikuliński, M., P. Malczyk, and R. Aarts. 2025. "Data-Driven Inverse Dynamics Modelling Using Neural-Networks and Regression-Based Techniques." *Multibody System Dynamics* 63 (3): 341–366. https://doi.org/10.1007/s11044-024-10024-2

Proctor, J. L., S. L. Brunton, and J. N. Kutz. 2016. "Dynamic Mode Decomposition with Control." *SIAM Journal on Applied Dynamical Systems* 15 (1): 142–161. https://doi.org/10.1137/15M1013857

Rahikainen, J., F. González, and M. Á. Naya. 2020. "An Automated Methodology to Select Functional co-Simulation Configurations." *Multibody System Dynamics* 48 (1): 79–103. https://doi.org/10.1007/s11044-019-09696-y

Raoofian, A., X. Dai, and J. Kövecses. 2024. "Contact Representation in Robotic Mechanical Systems Employing Reduced Models." *IEEE Robotics and Automation Letters* 9 (2): 1556–1563. https://doi.org/10.1109/LRA.2023.3342549

Rodríguez, B., A. J. Rodríguez, B. Sputh, R. Pastorino, M. A. Naya, and F. González. 2022. "Energy-Based Monitoring and Correction to Enhance the Accuracy and Stability of Explicit co-Simulation." *Multibody System Dynamics* 55 (1-2): 103–136. https://doi.org/10.1007/s11044-022-09812-5

Sadjina, S., L. T. Kyllingstad, S. Skjong, and E. Pedersen. 2017. "Energy Conservation and Power Bonds in co-Simulations: Non-Iterative Adaptive Step Size Control and Error Estimation." *Engineering with Computers* 33 (3): 607–620. https://doi.org/10.1007/s00366-016-0492-8

Schmid, P. J. 2022. "Dynamic Mode Decomposition and Its Variants." *Annual Review of Fluid Mechanics* 54 (1): 225–254. https://doi.org/10.1146/annurev-fluid-030121-015835

Schweizer, B., P. Li, and D. Lu. 2015. "Explicit and Implicit Cosimulation Methods: Stability and Convergence Analysis for Different Solver Coupling Approaches." *Journal of Computational and Nonlinear Dynamics* 10 (5). 051007-051019. https://doi.org/10.1115/1.4028503

Tamellin, I., D. Richiedei, B. Rodríguez, and F. González. 2022. "Eigenstructure Assignment and Compensation of Explicit co-Simulation Problems." *Mechanism and Machine Theory* 176: 105004. https://doi.org/10.1016/j.mechmachtheory.2022.105004

Wojtyra, Marek, Marcin Pękal, and Janusz Frączek. 2020. "Utilization of the Moore-Penrose Inverse in the Modeling of Overconstrained Mechanisms with Frictionless and Frictional Joints." *Mechanism and Machine Theory* 153: 103999. https://doi.org/10.1016/j.mechmachtheory.2020.103999

Zhang, H., C. W. Rowley, E. A. Deem, and L. N. Cattafesta. 2019. "Online Dynamic Mode Decomposition for Time-Varying Systems." *SIAM Journal on Applied Dynamical Systems* 18 (3): 1586–1609. https://doi.org/10.1137/18M1192329

Zhang, R., H. Zhang, A. Zanoni, A. Tasora, and P. Masarati. 2022. "Explicit Smooth/Nonsmooth Cosimulation Using Kinematic Constraints." *Multibody System Dynamics* 55 (1-2): 3–37. https://doi.org/10.1007/s11044-022-09829-w