Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

Di Girolamo G.D., Smarra F., Gattulli V., Potenza F., Graziosi F., D'Innocenzo A., "Data-driven optimal predictive control of seismic induced vibrations in frame structures", Structural Control and Health Monitoring, e2514, 2020 https://doi.org/10.1002/stc.2514.

Data-driven optimal predictive control of seismic induced vibrations in frame structures

G. D. Di Girolamo[1] | F. Smarra[1] | V. Gattulli*[2] | F. Potenza[3] | F. Graziosi[1] | A. D'Innocenzo[1]

[1]Department of Information Engineering, Computer Science and Mathematics, Università degli Studi dell'Aquila, L'Aquila, Italy
[2]Department of Structural and Geotechnical Engineering, Sapienza - University of Rome, Rome, Italy
[3]Department of Civil, Construction-Architectural and Environmental Engineering, Università degli Studi dell'Aquila, L'Aquila, Italy

Correspondence
*Corresponding author Email: vincenzo.gattulli@uniroma1.it

Present Address
Via Eudossiana 18, 00184, Rome, Italy

Summary

Complex civil engineering structural systems are prone to seismic induced vibrations during which their inherent dynamical features are displayed often causing discrepancies with the prediction of classical idealized models. In the paper effective control of such systems is pursued by a novel data-driven modeling and a control methodology based on a technique from Machine Learning: Regression Trees. A Training process to create a state-space model based on partitioning the dataset is illustrated. State- and output-feedback are derived using the recursively identified model in order to reach suitable performance event for an unknown structure. A benchmark frame structure has been used to demonstrate the effectiveness of the entire procedure.

KEYWORDS:
Data-driven, predictive control, earthquake engineering, random tree, time-variant systems

1 | INTRODUCTION

Over the last 30 years, great attention has been devoted to the integration of the Information and Communication Technology (ICT) into the civil engineering field exploring, for example, the potentiality of actively controlling the vibrations in structures [1,2]. In particular, the mitigation of the lateral forces-induced vibrations in the buildings, such as strong wind or earthquake events, can preserve these structures from activation of large displacements, velocities and accelerations. Hence, in the last years control systems have been largely introduced and designed to reduce structural response even paying attention to specific aspects such as for example the peak response reduction [3]. Control systems can be used not only to protect the structures from damage induced by a catastrophic event [4], but also to maintain its strength and serviceability during the whole design life. In general, a closed-loop vibration control system can be classified in 3 different categories: active, passive and semi-active [5,6,7,8].
Active control is the technology which realizes the idea to use the information describing the actual structural response to evaluate control force to be applied in order to reach assigned performances. Indeed, to protect actively a structure, analytical, numerical and experimental studies have been conducted to demonstrate the effectiveness of this strategy in different applications [9,10].
It is easy to understand how, in order to obtain the maximum performance of the control action, the model resolution and a careful selection of its parameters play a crucial role. In this hand, on-line identification of the parameters of a linear model of a structure under seismic excitation [11] permits to update the structural model on the basis of the measured response, which becomes more challenging and effective in multi-physic problems in which fluid-structure interaction should be controlled [12].
Generally, the design of model-based controllers relies on a mathematical model derived using the nominal values of the mechanical parameters describing the structure. Techniques used to find an accurate control solution are based on Lyapunov stability theory, bang-bang control, adaptive control [13] and clipped optimal control [14], which use a deterministic model for the design. Other algorithms are optimal control-based [15] in which the Linear Quadratic Regulator (LQR) is one of the most used design method [16,17]. It allows to define the optimal control force gains able to minimize a performance index appropriately weighted

on the controlled state and control force. Moreover, in civil engineering, a different target can be pursued like reduction of displacement, velocity or absolute accelerations and inter-story drifts[15]. LQR design is based on a cost function that has to be minimized through a reasonable trade-off between the control cost and the achievable reduction of structural vibrations. However, this approach requires measurements of all the state variables, which in real applications is often impossible. Under certain hypotheses Linear Quadratic Gaussian control (LQG) can be exploited[18], which allows to estimate the internal (unmeasured) variables through an observer, e.g. Kalman Filter. Nevertheless, also the LQG has some drawbacks due to time delay between input and output, high sensitivity to spillover phenomenon, and no guarantees on stability in terms of robustness. Recently, a solution to solve these issues has been proposed exploiting a linear combination of measured signals for feedback (optimal static output feedback - OSOF or partial state feedback)[19]; the procedure aims to minimize the difference between the performance obtained by LQR and OSOF. The proposed method permits to choose optimal gains which minimize the influences on the performance due to the sensor location and system initial condition. An alternative to the aforementioned control algorithms is Model Predictive Control (MPC)[20], which is a well-known control strategy used to design optimal control actions to obtain a desired system performance. Differently from LQR and LQG, MPC minimizes a performance index over a finite horizon and is applied following a receding horizon scheme. The advantage of such approach is that at each time step it solves a finite horizon optimal control problem, and applies only the first input of the computed optimal sequence. At the next time step it updates the current state with the real measurement, and repeats the optimization. This allows, step by step, to correct modeling errors that otherwise can accumulate, as in the case of the LQR and LQG. MPC is a mature technique that has been already introduced in the civil engineering field and applied to reduce seismically-induced vibrations and displacements: some of the main research articles apply MPC to reduce seismically-induced vibrations, and the authors verify the equivalent effectiveness of the MPC in simulation environments compared to the optimal control methods[21,22,23]. A semi-active predictive control strategy has been also proposed for damping control of base-isolated structures employing semi-active fluid dampers when subjected to earthquakes[24]: a key factor to provide such optimal control strategies is that MPC uses dynamical models that can be updated at each step to predict the system's behavior over a finite horizon. In this context, a structural model of a complex system may need to be updated on the basis of measured data, even during control actions. MPC has been also recently applied in[25] to structural monitoring considering both near-field (pulse) and far-field ground motions.

All the previous approaches are based on a dynamical model of the structure, which is in some cases difficult to derive. Many researchers have tackled this problem using different data-driven model-free approaches (e.g. fuzzy sliding mode controller[26], fuzzy logic control[27] and fuzzy logic and neural networks[28]). In[29] the authors propose a method based on an online control of active structural systems taking into account both seismic uncertainties and time delay combining Reinforcement Learning (RL) algorithms (which include a fuzzy gain-scheduling controller) with a state predictor in order to face the input time delay. The performance of this methodological approach have been evaluated experimentally using a massive database of potential earthquakes. Also in[30] authors introduce a method for vibration suppression of an unknown system based on an artificial-neural network that utilizes a principle of RL. Although there have been remarkable results, the known main disadvantage of model-free techniques is that a large number of interactions between agent and environment is needed before learning a good control policy.

In general, the drawback of the previous approaches is the uncertainty due to inaccurate physical models, the variability of future excitations (as the earthquake[31]), or input time delay of the actuators' dynamics and processing time. Complex structures on which high performance and reliability are needed when seismic events occur have to be modelled with a high level of precision in order to implement advanced control algorithms. The development of such analytical models and the identification of all the parameters can be a hard and time consuming task, often prohibitive from a practical viewpoint. Moreover it is not always possible to even find a good trade-off between the simplicity of a model and its ability to represent faithfully the behavior of a building under certain inputs.

However, the advancement in electronics and sensory components allows the measurement of accelerometer data of the response of a structure from external disturbances, by placing sensors in critical nodes of a building. These data may be used for the identification of models by using techniques from control theory, even considering delays[32].

The direct use of measured data to derive a suitable model able to predict the dynamical behavior of the observed system has reached a certain level of reliability within the scientific community thanks to several works related to the so-called Stochastic Subspace Identification method (SSI). These methods are able to identify on the basis of measured variables a linear state space model, with a selected order, which assures the minimization of the error between measured and predictive quantities in their time evolution under stochastic disturbances[33].

Another possibility is to use machine learning algorithms to create models using historical data available to the system and

apply advanced predictive control techniques. The advantage of using data-driven techniques is that they do not require any a-priori physical model of the structure: they only use information collected from the sensors measurements and extract dynamical models of the system just from the data. In machine learning there are several techniques that can be used for this scope, as for example Neural Networks, Regression Trees (RT), Random Forests, Gaussian Processes, and others. Indeed many interesting results present in the literature exploit the aforementioned approaches in order to perform predictions on the response of a structure subject to an earthquake load, early warnings and damage estimation due to reduced stiffness[34,35,36,37,38,39].

However, the existing techniques do not provide models that are directly suitable for structural control, and in particular for applying MPC. For this reason, a recent very active research line in the literature tackles the problem of combining Machine Learning and Model Predictive Control, demonstrating its effectiveness in different application domains (e.g. in the energy and automation communities[40,41,42,43,44,45,46] just to cite a few). On this line of research, and in order to address the drawbacks discussed above, in this paper we leverage the theory derived in[45,46], where static and dynamical models are identified using regression trees-based algorithms, which enable N-steps lookahead closed-loop model predictive control for energy systems. We will show how models identified combining Regression Trees theory with ARX system identification can be used to capture the dynamics of a structure subject to earthquakes, and to reduce the oscillations and the displacements by setting up an active predictive control strategy based on MPC. The reason behind the choice of Regression Trees as a modeling technique lies in the fact that, as shown in[46], it is possible to model system's dynamics with a Switching Affine dynamical model, rather than the nonlinear models provided by Neural Networks[47] and Gaussian Processes[48]: as a direct consequence, it is possible to set up a data-driven MPC problem that can be efficiently solved in run-time via Quadratic Programming. More in detail, we create N regression trees (see Appendix A for a detailed definition of the classical Regression Trees algorithm as in[49]), each one used to predict the state evolution of the system on a different future step $1, 2, \dots, N$ of the horizon. To each leaf of each tree we associate a linear dynamical model using the samples in the leaf via the Least Squares methodology. Finally, through an extension of the state-space that takes into account the inputs to apply over the horizon, we are able to derive a Switching Affine dynamical model in state-space form that can be directly used for implementing MPC.

To show the advantages of our RT-based identification technique we compare it with Stochastic Subspace Identification (SSI) over a dataset obtained from a set of experiments on a large scale prototype two floor building with symmetric mass distribution, realized in the Structural Laboratory of the DiSGG (University of Basilicata, Italy) and mounted on a rigid base. We first identify two models (RT- and SSI-based) and compare the prediction accuracy both in the time and frequency domain on the experimental dataset. Then we compare, in a simulative environment, the performance of the MPC algorithms based on the two models, with respect to acceleration peaks, displacements, velocities and kinetic energy. We will show that our RT-based technique outperforms classical SSI both in terms of prediction accuracy and control performance.

Paper organization. In Section 2 we recall the theoretical background of our model identification and control methodologies using Regression Trees. In Section 3 we describe the experimental setup we used to derive the dataset for the identification process. In Section 4 we illustrate the identification processes we used to derive models based on the dataset of the experimental setup using both SSI- and RT-based techniques. In Section 5 we compare standard SSI and our novel RT-based technique both in terms of prediction accuracy and control performance.

Notation. We denote by $|\mathcal{X}|$ the cardinality of the set $\mathcal{X}$ and by $\biguplus$ the disjoint union. Given a vector $x \in \mathbb{R}^n$ we denote by x_i the i^{th} component of x.

2 | DATA-DRIVEN FRAMEWORK

The aim of this section is to describe how to identify a switching affine dynamical model of a system starting from historical data, by appropriately combining techniques from machine learning and control theory. We will define the standard CART (Regression Tree) algorithm (Section 2.1), illustrate how it can be modified to identify a switching affine dynamical model (Section 2.2), and finally show how to use such model to design a Model Predictive Control (MPC) problem (Section 2.3). In the rest of the paper we will demonstrate that such methodology can be used to model a building subject to earthquakes, and apply MPC to control the building in order to minimize the amplitude of the oscillations induced by the earthquake on the structure.

2.1 | Dataset definition and the CART (Regression Tree) Algorithm

In this section we define how the dataset is organized and illustrate the regression tree algorithm. Let a dataset $(\mathcal{X}, \mathcal{Y})$ be given, composed by the samples collected from the sensors placed on the structure, where $\mathcal{X} = \{s_k^x\}_{k=1,\dots,|\mathcal{X}|}$, is the set of predictor variables (or features), and $\mathcal{Y} = \{s_k^y\}_{k=1,\dots,|\mathcal{Y}|}$, with $|\mathcal{Y}| = |\mathcal{X}|$, is the set of response variables. Each sample s_k^x, s_k^y in the dataset is a vector referring to all the measurements taken from the sensors at each time instant k, and their regressive terms. In particular,

$$s_k^x = [w(k), \dots, w(k-\delta_w), x(k), \dots, x(k-\delta_x), u(k), \dots, u(k-\delta_u)] \in \mathbb{R}^{\delta_w r + \delta_x n + \delta_u m} \tag{1}$$

is the sample of the dataset corresponding to the measurements taken at time k from the sensors, and associated to the disturbances $w \in \mathbb{R}^r$, as for example the earthquake accelerations the structure is subject to, the outputs/states $x \in \mathbb{R}^n$, as for example the displacement, the velocity, and the acceleration induced by the earthquake and the dampers on the structure, and the control variables $u \in \mathbb{R}^m$, as for example the forces applied to the structure to reduce its oscillation. Similarly,

$$s_k^y = x(k+1) \in \mathbb{R}^n \tag{2}$$

is the sample of data associated, at time k, to the value of the outputs/states at the next time step, and represents the variable we wish to predict. Without any loss of generality and for the sake of simplicity, in the rest of the section we consider only a single output variable in the set $\mathcal{Y}$, i.e.

$$s_k^y = x_i(k+1) \in \mathbb{R},\ i \in \{1, \dots, n\}. \tag{3}$$

The same methodology can be trivially applied to the multiple output case by considering different trees for different outputs or multi-output regression trees, as shown in[42].

The Regression Trees algorithm creates a tree structure $\mathcal{T}$ by partitioning the dataset $\mathcal{X}$ into smaller regions, the *leaves* of the tree, following specific rules (see Appendix A and[49]). Each leaf i contains a subset of samples from $\mathcal{X}$: more precisely, let $\ell_i \subset \mathcal{X}$, with $i = 1, \dots, p$, be the set of samples contained in the i^{th} leaf of $\mathcal{T}$, then the leaves of $\mathcal{T}$ form a partition of $\mathcal{X}$:

$$\mathcal{X} = \biguplus_{i=1}^{p} \ell_i, \quad \left(\ell_\alpha \cap \ell_\beta = \emptyset,\ \forall \alpha \neq \beta\right). \tag{4}$$

After partitioning the dataset, the algorithm associates to each leaf ℓ_i a prediction $\hat{y}_i$ given by the average of the response values associated to each sample in ℓ_i. This algorithm is known as CART (see A for a general overview, and[49] for more details).

2.2 | Data-driven modeling using regression trees

The prediction provided by the CART algorithm consists of a constant value (A3): as a consequence, the obtained model lacks of a structure, and can not be used in a MPC problem setup. To address this issue, the following approach has been addressed in[46]: the goal is to define a data-driven state-space modeling framework to predict $x(k+j)$ over an horizon defined by $j = 1, \dots, N$, given the measurements of the variables at the current time step.

The first step is to split the set $\mathcal{X} = \{\mathcal{X}_c, \mathcal{X}_{nc}\}$ into the set $\mathcal{X}_c = \{s_1^c, \dots, s_{|\mathcal{X}|}^c\}$, of data associated to the m control variables at time k, i.e. $s_k^c = u(k) \in \mathbb{R}^m$, and the set $\mathcal{X}_{nc} = \{s_1^{nc}, \dots, s_{|\mathcal{X}|}^{nc}\}$, of data associated to the $r+n$ disturbance and output/state (non-control) variables at time k and to the m control variables at time $k-1$, and their regressive terms, i.e.

$$s_k^{nc} = [w(k), \dots, w(k-\delta_w), x(k), \dots, x(k-\delta_x), u(k-1), \dots, u(k-\delta_u)] \in \mathbb{R}^{\delta_w r + \delta_x n + (\delta_u - 1)m}. \tag{5}$$

The training process to create a state-space model starting from our split dataset is divided into 3 steps:

Step 1: we train N regression trees $\mathcal{T}_j,\ j = 1, \dots, N$ using the CART algorithm, one for each future predictive step up to the future horizon N;

Step 2: we fit an affine model for each leaf of each tree using the samples contained in that leaf;

Step 3: we create for each sequence of leaves $i_1, \dots, i_N$, respectively of $\mathcal{T}_1, \dots, \mathcal{T}_N$, a switching affine dynamical model.

Step 1. Let us consider a predictive horizon equal to N. We use the partitioning procedure defined above to grow multiple trees $\mathcal{T}_1, \dots, \mathcal{T}_N$, only using the dataset $(\mathcal{X}_{nc}, \mathcal{Y})$. Each tree is grown optimising the prediction of $x(k+j)$ at the j^{th} step of the future horizon.

Step 2. We use the data contained in each leaf of the dataset $\mathcal{X}_c$ to associate, to each leaf ℓ_{i_j} of each tree $\mathcal{T}_j$, a model of the form:

$$x(k+j) = A'_{i_j} x(k) + \sum_{\alpha=1}^{j} B'_{i_j,\alpha} u(k+\alpha-1) + f'_{i_j}. \tag{6}$$

The reason for including matrices $B'_{i_j,\alpha}$ that multiply past inputs in the identification, is to enforce the prediction accuracy that regression trees can provide. Matrices A'_{i_j}, $B'_{i_j,\alpha}$ and f'_{i_j} can be identified using several methods: in this paper we consider the least square methodology defined in the following. Let us consider the experiments associated to the samples $s_{k_1}, \dots, s_{k_\epsilon}$ in the leaf ℓ_{i_j} at time instants $k_1, \dots, k_\epsilon$, and their past δ_y values. For each leaf ℓ_{i_j}, let us define

$$\Lambda_{i_j} = \begin{bmatrix} 1 & \cdots & 1 \\ x(k_1) & \cdots & x(k_\epsilon) \\ \vdots & & \vdots \\ x(k_1-\delta_y) & \cdots & x(k_\epsilon-\delta_y) \\ u_1(k_1) & \cdots & u_1(k_\epsilon) \\ \vdots & & \vdots \\ u_m(k_1) & \cdots & u_m(k_\epsilon) \\ \vdots & & \vdots \\ u_1(k_1+j-1) & \cdots & u_1(k_\epsilon+j-1) \\ \vdots & & \vdots \\ u_m(k_1+j-1) & \cdots & u_m(k_\epsilon+j-1) \end{bmatrix}^\top \quad \xi_{i_j} = \begin{bmatrix} f \\ a_1 \\ \vdots \\ a_{\delta_y} \\ b_{1,1} \\ \vdots \\ b_{m,1} \\ \vdots \\ b_{1,j} \\ \vdots \\ b_{m,j} \end{bmatrix}, \tag{7}$$

$$\lambda_{i_j} = \begin{bmatrix} x(k_1+j) & \cdots & x(k_\epsilon+j) \end{bmatrix}^\top, \tag{8}$$

where Λ_{i_j} is the matrix containing all the samples within the leaf[†], ξ_{i_j} contains the parameters to be estimated and then used to build matrices A'_{i_j}, B'_{i_j}, and f'_{i_j} in (6), and λ_{i_j} is a vector containing the values of samples within the leaf at the prediction horizon $k+j$. We use this setup to formalize the following optimization problem:

Problem 1.

$$\begin{aligned} &\underset{\xi_{i_j}}{\text{minimize}} \quad \| \Lambda_{i_j} \xi_{i_j} - \lambda_{i_j} \|_2^2 \\ &\text{subject to } \Gamma_{eq}\, \xi_{i_j} = \gamma_{eq} && (9) \\ &\qquad\qquad \Gamma_{ineq}\, \xi_{i_j} \leq \gamma_{ineq} && (10) \end{aligned}$$

where (9) and (10) are used to constraint elements in ξ_{i_j} due to physical constraints of the system (if any a priori knowledge is available). The elements of ξ_{i_j}, obtained solving Problem 1, are used to build matrices A'_{i_j}, $B'_{i_j,\iota}$ and f'_{i_j} in (6) as follows

$$A'_{i_j} = \begin{bmatrix} a_1 & a_2 & \cdots & a_{\delta_y} \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ & & \vdots & \\ 0 & 0 & \cdots & 0 \end{bmatrix} \quad B'_{i_j,\iota} = \begin{bmatrix} b_{1,\iota} & \cdots & b_{m,\iota} \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ & \vdots & \\ 0 & \cdots & 0 \end{bmatrix} \quad f'_{i_j} = \begin{bmatrix} f \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{11}$$

This first 2 steps of the procedure are depicted in Figure 1.

Step 3. To derive a switching affine dynamical model starting from the dynamical model (6), that can be used to setup an MPC problem[‡], we first define an extension of the state-space $x_e = \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix}$, where x_e is composed by the system state $\bar{x} \in \mathbb{R}^n$ and input dummy variables $\bar{u} = [u_{-N+1}, \dots, u_{-2}, u_{-1}]^\top \in \mathbb{R}^{m(N-1)}$, which represent the past values of the inputs applied to the system during the previous $N-1$ steps. We thus define the following switching model to predict the state for the next N future steps

$$x_e(k+j) = A^e_{\sigma_j(x(k),w(k))} x_e(k+j-1) + B^e_{\sigma_j(x(k),w(k))} u(k+j-1) + f^e_{\sigma_j(x(k),w(k))},\ j = 1, \dots, N, \tag{12}$$

[†]Note that these samples are not necessarily adjacent

[‡]Namely, such that at time k we are able to predict $x(k+1), x(k+2), \dots, x(k+N)$ as a function of the measurements $x(k), w(k)$ and of the control inputs.

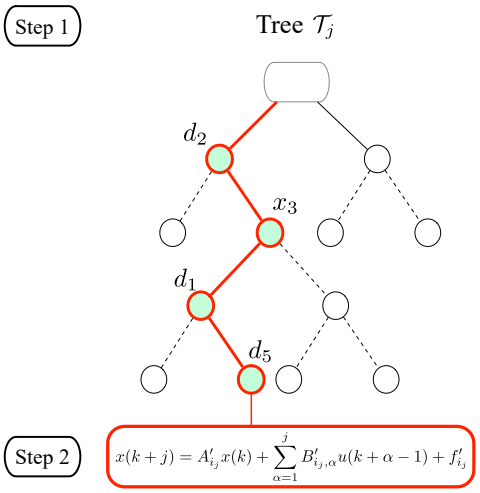

FIGURE 1 Separation of variables. *Step 1:* Tree $\mathcal{T}_j$ is trained only on the disturbances $\mathcal{X}_{nc}$ as the features. *Step 2:* In the leaf ℓ_{i_j} of the tree $\mathcal{T}_j$, a linear regression model parametrized by matrices A'_{i_j}, B'_{i_j} and f'_{i_j} is defined as a function of the current state and the control variables.

where

$$A^e_{\sigma_j(\cdot)} = \begin{bmatrix} A_{\sigma_j(\cdot)} & \bar{B}_{\sigma_j(\cdot),-N+1} & \bar{B}_{\sigma_j(\cdot),-N+2} & \cdots & \bar{B}_{\sigma_j(\cdot),-1} \\ \mathbf{0} & \mathbf{0} & I & \cdots & \mathbf{0} \\ & & & \ddots & \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & I \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \tag{13}$$

$$B^e_{\sigma_j(\cdot)} = \begin{bmatrix} \bar{B}^\top_{\sigma_j(\cdot),0} & \mathbf{0} & \cdots & \mathbf{0} & I \end{bmatrix}^\top \tag{14}$$

$$f^e_{\sigma_j(\cdot)} = \begin{bmatrix} \bar{f}^\top_{\sigma_j(\cdot)} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}^\top \tag{15}$$

and where the switching signal $\sigma_j((x(k), w(k)), j = 1, \dots, N$ associates to any sample measured at time k, a sequence of leaves $i_1, \dots, i_N$ respectively of trees $\mathcal{T}_1, \dots, \mathcal{T}_N$. Note that, in run-time control, Equation (12) can be used in the standard formulation of N-steps linear MPC, since the switching sequence of $\sigma_j(x(k), w(k))$, being a function of $(x(k), w(k))$, is available at time k for any $j = 1, \dots, N$.

The following proposition shows that, given a model as in (6) obtained from Step 2, it is possible to construct an equivalent model as in (12).

Proposition 1. Let A'_{i_j}, $B'_{i_j,\alpha}$ and f'_{i_j}, $\alpha = 0, \dots, j$, $j = 1, \dots, N$, be given as output of Step 2. For any initial condition $\bar{x}(k) = x(k) = x_k$, any sequence of leaves $i_1, \dots, i_N$, and any control sequence $u(k), \dots, u(k+N-1)$, if A'_{i_j} is invertible for any $j = 1, \dots, N$ then there exist $A^e_{i_j}, B^e_{i_j}$ and $f^e_{i_j}$, $j = 1, \dots, N$, such that $\bar{x}(k+j) = x(k+j)$, $\forall j = 1, \dots, N$[§].

Proof. We only provide a sketch of the proof: for the complete proof one can refer to[46]. The main idea is, given a switching sequence $i_1, \dots, i_N$, to inductively obtain Equation (12) from Equation (6). In the base step $j = 1$, it is straight to obtain Equation (12) starting from (6). In the next step $j = 2$, one can compute $x_e(k) = (A^e_{i_1})^{-1} x_e(k+1) - B^e_{i_1} u(k) - f^e_{i_1}$ from the previous step and replace it in (6) for $j = 2$. By iteration, one can define $A^e_{i_j}, B^e_{i_j}$ and $f^e_{i_j}$ for any $j = 1, \dots, N$ by replacing the matrices in

[§]We recall that $\bar{x}(k)$ is the first component of the extended state variable $x_e = \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix}$

Equations (13),(14),(15) by the following:

$$A_{i_j} = \begin{cases} A'_{i_1} & \text{if } j = 1 \\ A'_{i_j}(A'_{i_{j-1}})^{-1} & \text{if } j > 1 \end{cases} \tag{16}$$

$$\bar{B}_{i_j,-\mu} = \begin{cases} B'_{i_j,0} & \text{if } \mu = 0 \\ B'_{i_j,j-\mu} - A'_{i_j}(A'_{i_{j-1}})^{-1} B'_{i_{j-1},j-\mu} & \text{if } 0 < \mu < j \\ \mathbf{0} & \text{if } \mu \geq j \end{cases} \tag{17}$$

$$\bar{f}_{i_j} = \begin{cases} f'_{i_j} & \text{if } j = 1 \\ f'_{i_j} - A'_{i_j}(A'_{i_{j-1}})^{-1} f'_{i_{j-1}} & \text{if } j > 1 \end{cases} \tag{18}$$

□

The whole procedure is summarized in Algorithm 1.

Algorithm 1 Switching affine model

1: **EXECUTION TIME: OFF-LINE**
2: **procedure** TRAINING AFFINE MODELS IN LEAVES
3: Set $\mathcal{X}_c \leftarrow$ control features
4: Set $\mathcal{X}_{nc} \leftarrow$ non-control features
5: Build N predictive trees $\mathcal{T}_j$ using $(\mathcal{X}_{nc}, \mathcal{Y})$
6: Compute A'_{i_j}, $B'_{i_j,\iota}$ and f'_{i_j} in (6) solving Problem 1 for each leaf ℓ_{i_j} of $\mathcal{T}_j$
7: Compute $A^e_{i_j}$, $B^e_{i_j}$ and $f^e_{i_j}$ in (12) via Proposition 1 for each sequence of leaves $i_1, \dots .i_N$
8: **end procedure**

2.3 | Data-driven Switched Affine Model Predictive Control using Regression Trees

Once the switching affine model (12) is created, it can be used to formalize the following MPC problem.

Problem 2.

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & \sum_{j=1}^{N} x_{k+j}^\top Q x_{k+j} + u_{k+j-1}^\top R u_{k+j-1} \\ \text{subject to} \quad & x_{k+j} = A^e_{\sigma_j(x(k),w(k))} x_{k+j-1} + B^e_{\sigma_j(x(k),w(k))} u_{k+j-1} + f^e_{\sigma_j(x(k),w(k))} \\ & u_{k+j-1} \in \mathcal{U} \\ & x_k = x(k),\ j = 1, \dots, N. \end{aligned}$$

Given the current measurements $x(k)$ and $w(k)$ at each instant k, in run-time, one can narrow down to a leaf ℓ_{i_j} of each tree $\mathcal{T}_j$ to find the switching signal $\sigma_j(x(k), w(k)) = i_j$, for $j = 1, \dots, N$, to obtain the switching affine model (12) for each step of the horizon, and then solve a classical MPC problem using Quadratic Programming. This process is described in Algorithm 2.

Remark 1. In many practical cases, the knowledge/forecast at each time k of the future disturbance signal $(w(k+1), \dots, w(k+N-1))$ can be used to learn the model, as we illustrate in the building automation setup we addressed in [45] where the disturbance consists of weather conditions. This can improve the prediction accuracy and the control performance *if*, at each step k of the real-time control, we can predict the future disturbance signals, i.e. the earthquake ground accelerations, for the future time steps $k+1, \dots, k+N-1$. In this paper we decided not to assume the possibility to predict the earthquake accelerations in the real-time control for two main reasons: (1) since in most cases such prediction is very difficult/impossible to be derived, we show that even without it we obtain an accurate predictive model; (2) the extension of our technique to that case is trivial.

Algorithm 2 Data-driven Model Predictive Control

EXECUTION TIME: RUN-TIME (ON-LINE)
procedure DETERMINE OPTIMAL CONTROL INPUT
  while $k \geq 0$ **do**
    Get measurement sample $(x(k), w(k))$ from sensors
    for all trees $\mathcal{T}_j,\ j = 1, \dots, N$ **do**
      Find leaf i_j narrowing down the j^{th} tree
    end for
    Given the leaves sequence $i_1, \dots, i_N$ define $A^e_{i_j}$, $B^e_{i_j}$ and $f^e_{i_j}$ obtained by Algorithm 1
    Solve Problem 2 using Quadratic Programming
    Apply the first input of the optimal sequence, i.e. $u(k) = u^*_k$
    Assign $k = k + 1$
  end while
end procedure

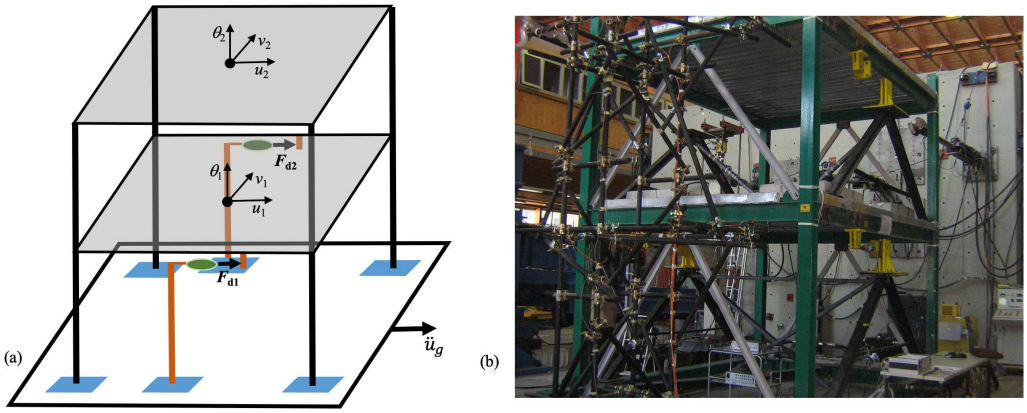

FIGURE 2 (a) Sketch of the analytical model. (b) DPC-ReLUIS benchmark structure (called JETPACS) at DISGG of University of Basilicata (Italy).

3 | CASE STUDY DESCRIPTION

The case study is related to a benchmark structure used in the experimental activity of the Italian DPC-ReLUIS Research Project 2005-2008 [50,51], illustrated in Figure 2. Such building prototype was realized to simulate the seismic response of a two-story, large scale (2:3), steel structure. It was built at the Structural Laboratory of the DiSGG (University of Basilicata, Potenza, Italy) with the aim of assessing the experimental performances of different passive and semi-active strategies in the mitigation of seismic-induced vibrations. The structure was connected to an actuator at the base able to provide a seismic acceleration and, moreover, was equipped of braces to dampen the seismically-induced oscillations through the dissipative action of magnetorheological dampers working on the first inter-story drift. The steel structure had a total height of 4.00 m with an equal inter-story height of 2.00. The two planar dimensions were different: 4.00 m in x-direction (along excitation direction) and 3.00 m in y-direction (cross excitation direction). The cross sections were IPE180 and HEB140 for beams and columns, respectively, while the material were a steel Fe360. In order to reproduce the real dynamic behavior were carried out output-only experimental tests in November 2007 aimed at updating the mechanical parameters of a structural model. All information regarding the identification procedures applied and the related results can be found in [52]. Subsequently, experimental tests related to the semi-active control of the prototype were implemented in October 2008. The control variables, i.e. the voltage send to the wonder-box (conditioning device able to transform the voltage in magnetic field) were selected according to the Clipped Optimal Control strategy. Moreover, in

the execution of tests, was chosen to assign to the control variable only two values, 0 and 5 Volt. The transition between the two state was assessed on the base of the comparison between the actual force in the devise and the active optimal control force. The latter was calculated according to the H2/LQG optimization criterion while the comparison between the two forces was carried out by a Heaviside function. The control algorithm, H2/LQG, receives in input eight accelerometric measurements, coming from structural points opportunely selected, and provides in output the vector of the control forces. In particular, a dynamic observer, activated by such accelerometric signals (four measurements for each floor corresponding to the accelerations in X- and Y-direction in diagonally opposite points) estimated the values of the state variables described by the three plane baricentric displacements and velocities. Combining these values with the optimal coefficients (LQR control algorithm) it was possible to calculate the optimal control force. More than 40 tests were carried using as seismic input three spectrum-compatible (Italian code OPCM 3431, seismic zone 1, soil B) ground accelerograms with different intensity levels. Moreover, thanks to the dSPACE real-time processor, the resulting closed-loop cycle was found to be completed without significant time-delay. In the following sections we will describe more in detail the dataset generated using the experimental setup described above in terms of measured variables (forces, accelerations, etc), sampling times, number of samples and number of experiments.

4 | MODEL IDENTIFICATION

First of all, we pre-processed the dataset with all the collected data checking if all measurements are coherent and if there are voids that may distort the models. The training datasets are composed by 22 closed-loop experiments described in Section 4 considering different parameters for the control algorithm. Considering a sampling time of 10 ms, the training set is made of $|\mathcal{X}_{\text{train}}, \mathcal{Y}_{\text{train}}| = 900$ thousand samples. We kept out an experiment that we used for the validation, therefore the testing dataset

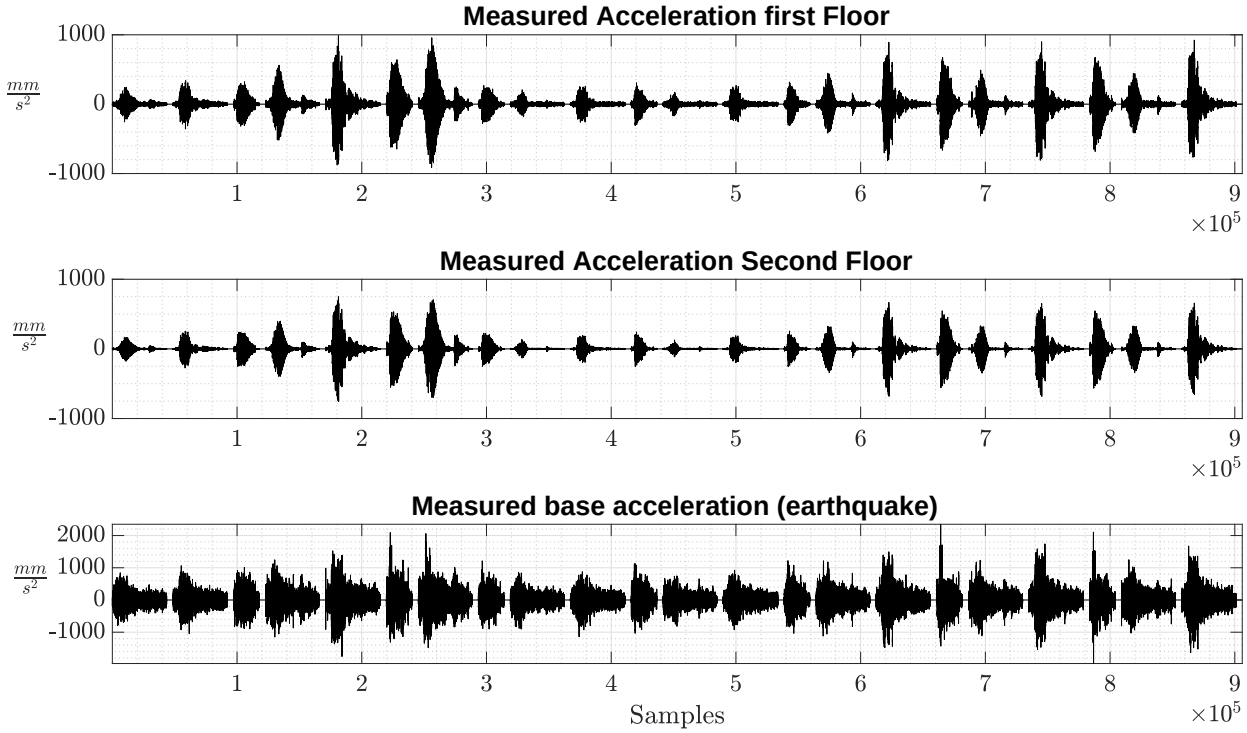

FIGURE 3 Part of data collected for the model identification. Acceleration along x axis (Training Data).

$(\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}})$ is composed by 25 thousand samples and an earthquake excitation different from the training dataset. Of course, the

same datasets (Train and Test) have been used both for the identification with SSI (N4SID) algorithm, the Data Driven algorithm (Algorithm 2) and the validation. Figure 3 shows the measurement of the accelerometers used to train the models.

As Data come from measurements of a structure in symmetric configuration, the experiment only took into account the acceleration components along the earthquake direction: the whole dataset is indeed composed by the following measurements:

- $w(k) \in \mathbb{R}$, the earthquake acceleration in $[\frac{mm}{s^2}]$.
- $x(k) \in \mathbb{R}^2$, the accelerations of the 2 floors along the earthquake direction in $[\frac{mm}{s^2}]$.
- $u(k) \in \mathbb{R}^2$, the control inputs applied to the structure in $[N]$.

Earthquake acceleration $w(k)$ is a signal without a statistical characterisation for our system, so we considered it as a disturbance that acts linearly on the state equations of our model, but cannot be predicted.

As the main aim of this work is to compare the new identification method illustrated in section 2.2, with the classical SSI applied in the Civil Enginering field, we first provide a short overview of SSI and illustrate how we derived and the SSI-based model, then we will illustrate how we applied the RT-based model.

SSI-based model: Stochastic Subspace Identification methods had a great development in both theory and practice since they can identify different models directly from the input-output data. The most influential ones are: Canonical Variate Analysis[53] (CVA), Multivariable Output Error State Space (MOESP)[54] and Numerical Subspace State Space System Identification (N4SID)[55]. Past works showed that the three algorithms use the same subspace to determine the order of the model and the extended observability matrix, but that the weights are different, basically all methods return models which are optimal but according to different optimization metrics[56]. Subspace methods have been developed also for certain classes of nonlinear system such as Hammerstein, Wiener and bilinear which are not considered in our Structure model. Other classical approaches for the identification of linear models, are ARX-ARMAX methods which exploit the least-square method in order to identify the parameters of a polynomial model that links the input-output samples[57]. N4SID algorithms have been proven to be always convergent and numerically stable since they only make use of QR and Singular Value Decompositions. The advantages of such methodologies are several, indeed they offer the possibility to correctly identify MIMO systems with very good numerical robusteness. MATLAB System Identification Toolbox provides functions, Simulink's blocks, and an app for constructing mathematical models of dynamical systems from measured data; it allows the design of models and express them in time-domain and frequency-domain, continuous-time and discrete-time, linear and nonlinear, and so on. In this paper we will exploit, due to its semplicity and robustness, the N4SID algorithm implemented in the System Identification toolbox in order to obtain two Linear Time Invariant (LTI) models. In particular, we derived:

- a *model* (Focus on prediction) that can be expressed in innovation form or classic state space form[58]: it ensures a lower prediction error and will be used in our Model Predictive Control formulation;
- a *simulation model* (Focus on simulation) that produces an accurate response to an input using input data: this model will be used to simulate the response of the experimental setup to the SSI- and RT-based MPC closed-loops, with the aim to compare the control performances of the models in a fair way.

The N4SID algorithm allows to chose freely the order of the Linear Model representing our structure; alternatively we can give an interval of values and the toolbox will suggest a Model Order value analyzing the Hankel singular values for models of different orders. As shown in figure 4 an increase of the model order would not bring a significative improvement of the accuracy of the model. According to Matlab algorithm, a model order of 4 is the best tradeoff between accuracy and complexity. We also want to point out that increasing too much the Model Order, with our data may introduce problems of stability for the identified model. The models identified with the Matlab Identification toolbox consist of 4 components for the internal state, 2 scalar inputs, 2 scalar outputs (Accelerations) and 1 scalar disturbance that acts on the state. Figure 4 also shows the interface of the Identification toolbox and some of the settings used to derive our models. It is also possible to constraint the structure of the dynamic matrix to be in a particular form (Canonical, Observability, Modal), but as this will not any influence on our problem, we left this choice to Matlab, forcing only to avoid unstable modes in the final model.

RT-based model: We applied the techniques described in Section 2. For the regressive terms of Equation (1) we chose $\delta_d = \delta_x = \delta_u = 20$, while we considered a predictive horizon of $N = 20$ steps (i.e. 2 seconds). Using data from $(\mathcal{X}_{\text{train}}, \mathcal{Y}_{\text{train}})$ we trained 20 trees for each component of the state $x \in \mathbb{R}^2$, each one predicting $x(k+j)$ over the predictive horizon for $j = 1, \dots, 20$. In the splitting procedure we set the minimum number of samples for each leaf equal to 10 then the CART algorithm will then

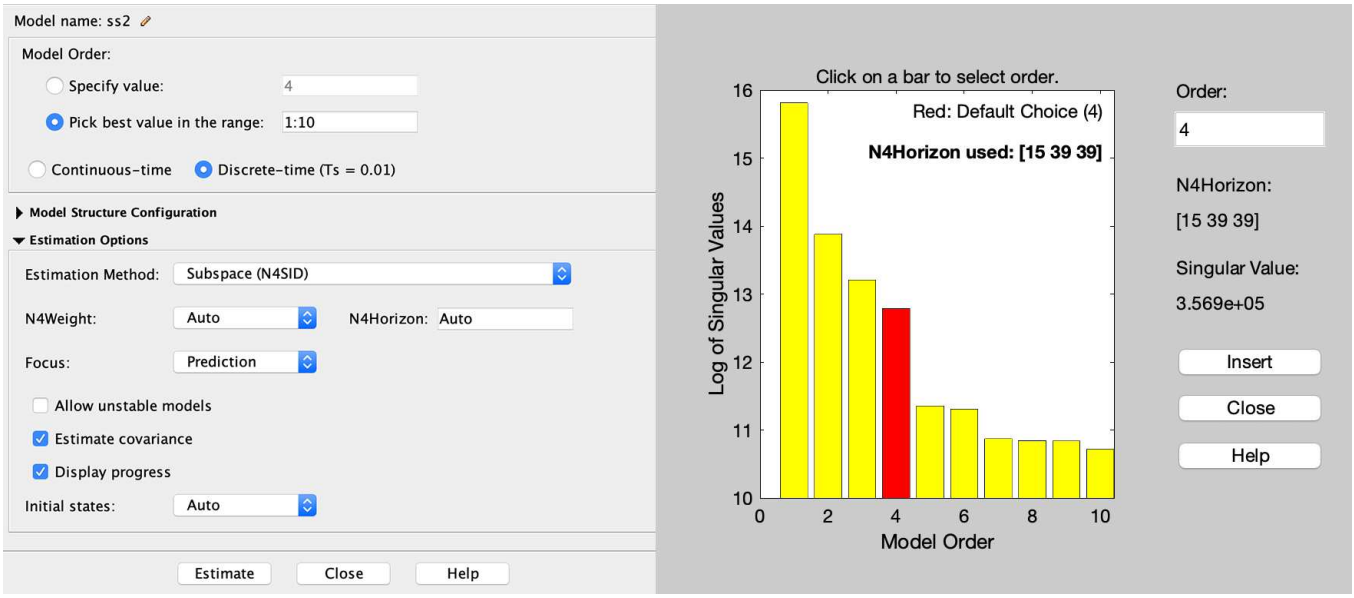

FIGURE 4 Matlab System Identification Toolbox APP. Some settings used to derive LTI models

optimally split the feature space considering this constraint. We thus obtain 40 trees which provide us parameters to build model (12). Such models have been derived constraining the parameters of the B matrices in each leaf in the Least Square Problem 1. In particular, Equation (10) has been setup to provide such parameters with an absolute value lower than 1. This technical detail is due to physical reasons, since we assume to apply to the system no more than the maximum force u designed by the controller. Therefore in the model, the different dynamics will be affected by a percentage of the total force applied to the structure.

5 | VALIDATION

In this section we provide simulation results in order to compare the identification and control performance between our RT-based approach and the classical N4SID identification approach. We first provide identification results, showing the accuracy of the models both in time and frequency domain, and then we compare the control performance.

5.1 | Identification accuracy

Validation of the predictive data-driven models for the 2 accelerations was performed using the testing dataset $(\mathcal{X}_{\text{test}}' \mathcal{Y}_{\text{test}})$. More precisely, given measurements at current time step k, i.e. $w(k)$, $x(k)$, $u(k-1)$, and their regressive terms, we predicted the behavior of the system over the horizon, i.e. $x(k+1), \dots, x(k+20)$, using the procedure described in Algorithm 2 from step 1 to step 8. The predicted values $x(k+j), j = 1, \dots, 20$ are compared with the corresponding true values. The results for the prediction of the Data Driven model at the next time step, i.e. $x(k+1)$, for the 2 components are shown in Figure 5. To each acceleration (True and Predicted), the plot of the absolute value of the error is associated as well. A zoom of the most critical parts, i.e. parts with the largest errors, is provided in Figure 6, to appreciate the difference among the predicted (red line) and the real (black line) states. The prediction accuracy of the state, at $k+5$ and $k+10$, and its zoom of the most critical parts, given the actual measurements, is shown respectively in Figures 7, 8,9 and 10 respectively. We can see how in this case the prediction error is more significant, although still a decent prediction.

In order to have a better comparison of the SSI- and RT-based identification approaches, in Figure 11 the Normalized Root Mean Square Error (NRMSE) among the predicted and the real trajectories is shown for each state over the horizon N and over the whole simulation period. The yellow and violet line show the prediction error for the model identified with SSI algorithm, while the blue and red lines show the error over the horizon for the Switching Affine Regression Tree model. The prediction error (NRMSE) at $k+i$, for $i = 1$ is very small and comparable between the two identification approaches: however, the advantage of the RT-based model arises already for $i \geq 2$. The performance of the identification increases as the predictive horizon raises,

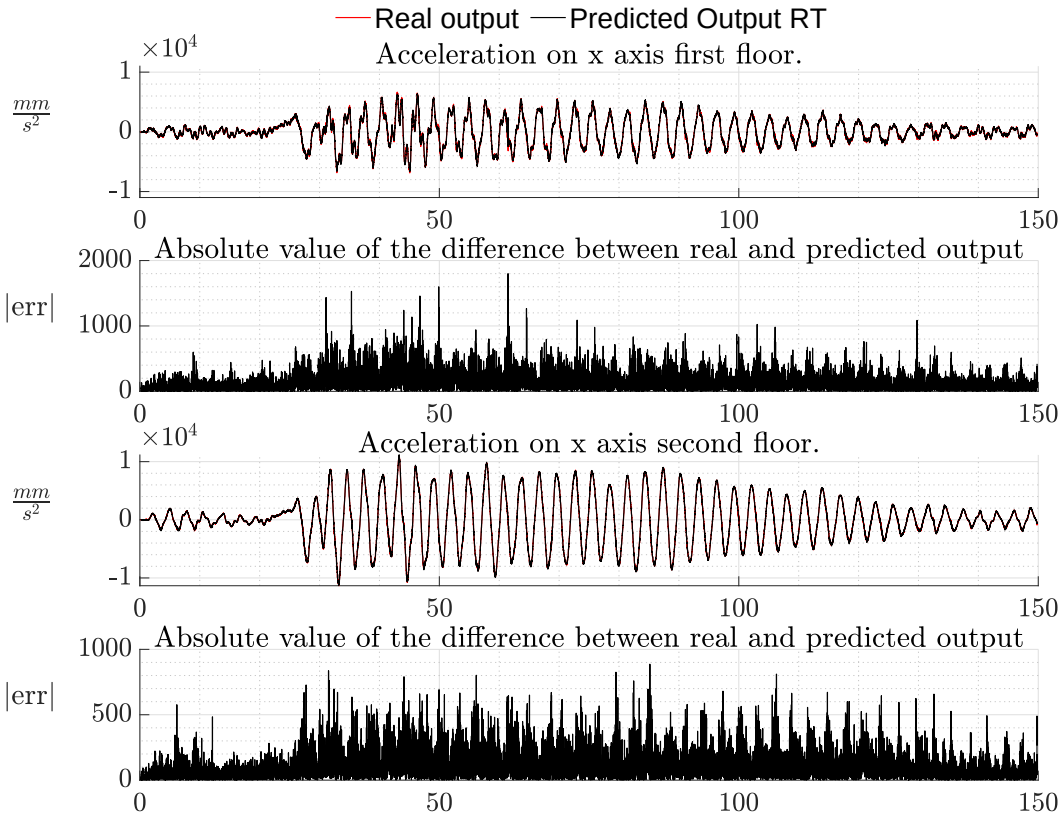

FIGURE 5 State predictions, i.e. accelerations of the 2 floors, at $k+1$, and associated absolute errors.

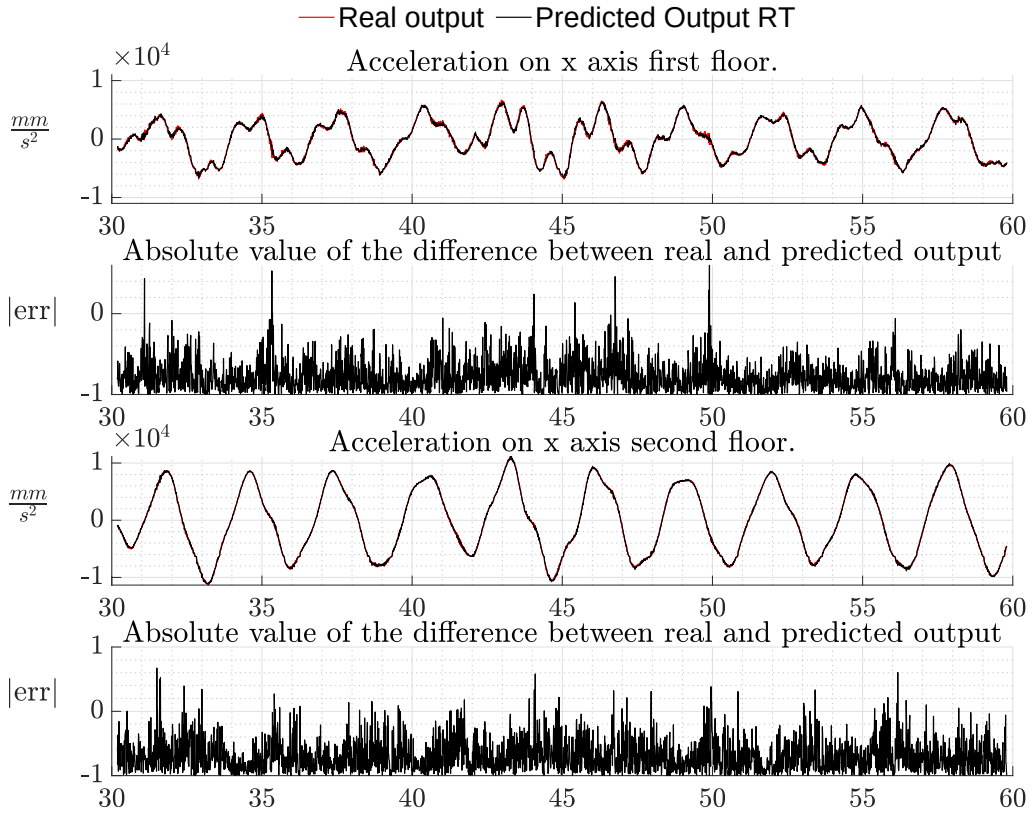

FIGURE 6 Zooms of state predictions, i.e. State predictions, i.e. accelerations of the 2 floors, at $k+1$, and associated absolute errors.

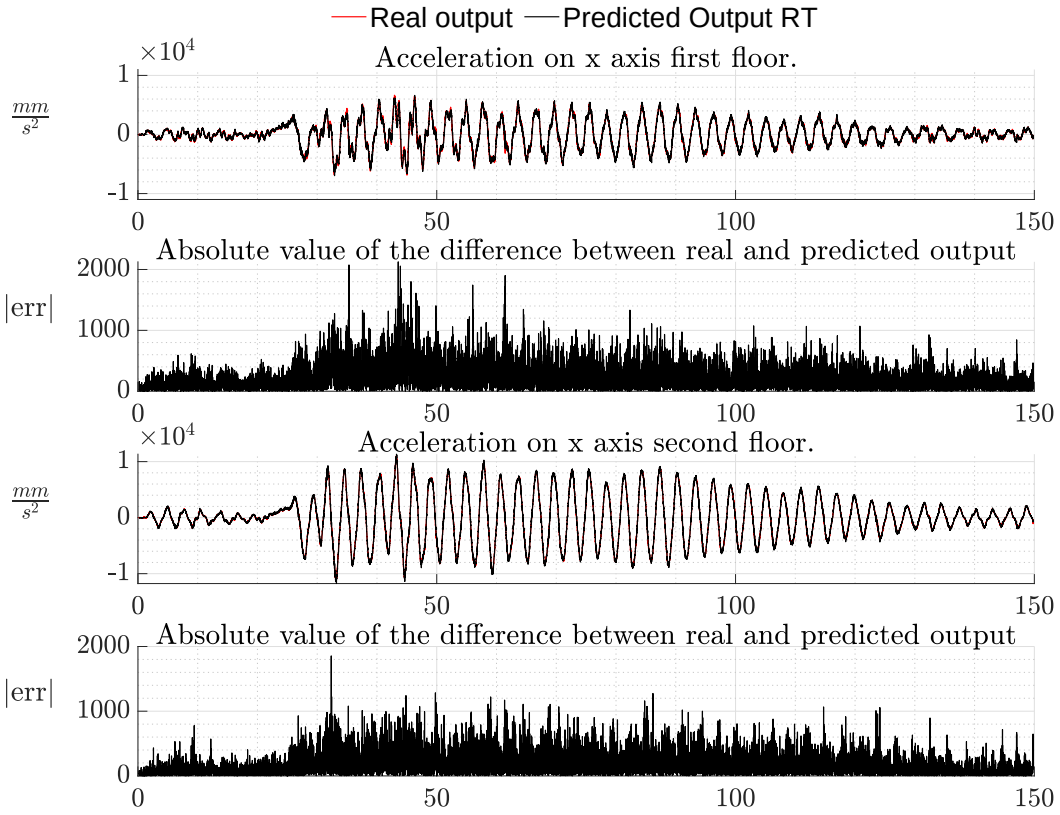

FIGURE 7 State predictions, i.e. State predictions, i.e. accelerations of the 2 floors, at $k + 5$, and associated absolute errors.

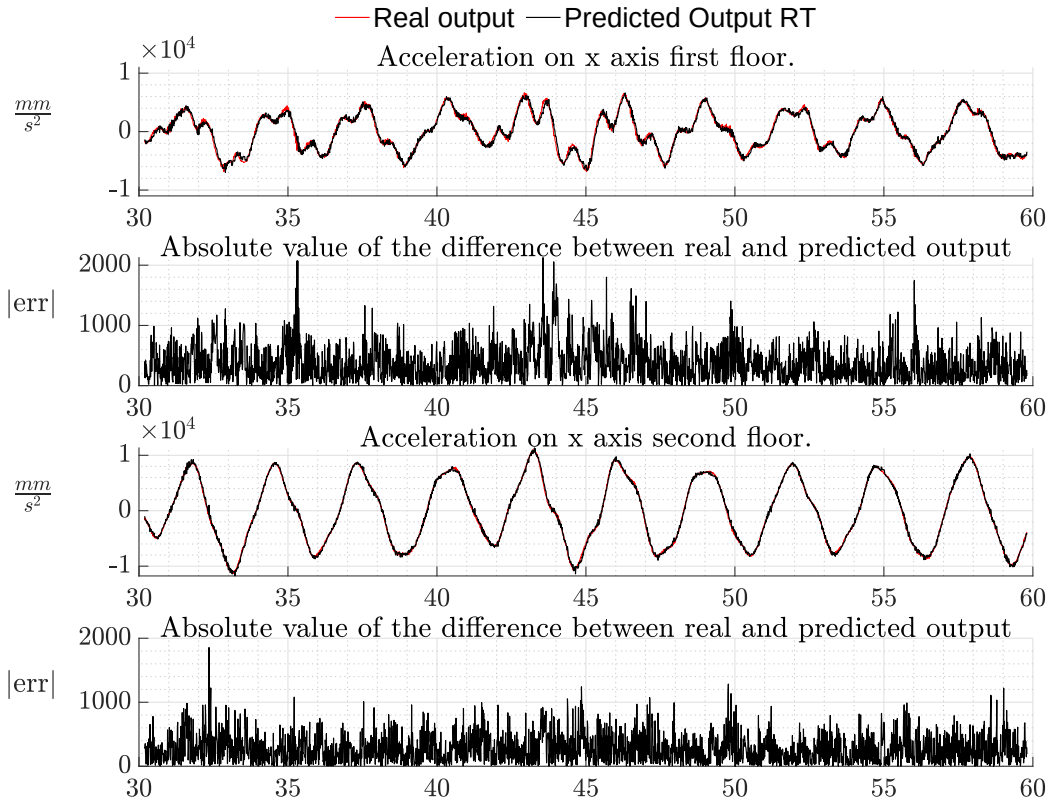

FIGURE 8 Zooms of state predictions, i.e. State predictions, i.e. accelerations of the 2 floors, at $k + 5$, and associated absolute errors.

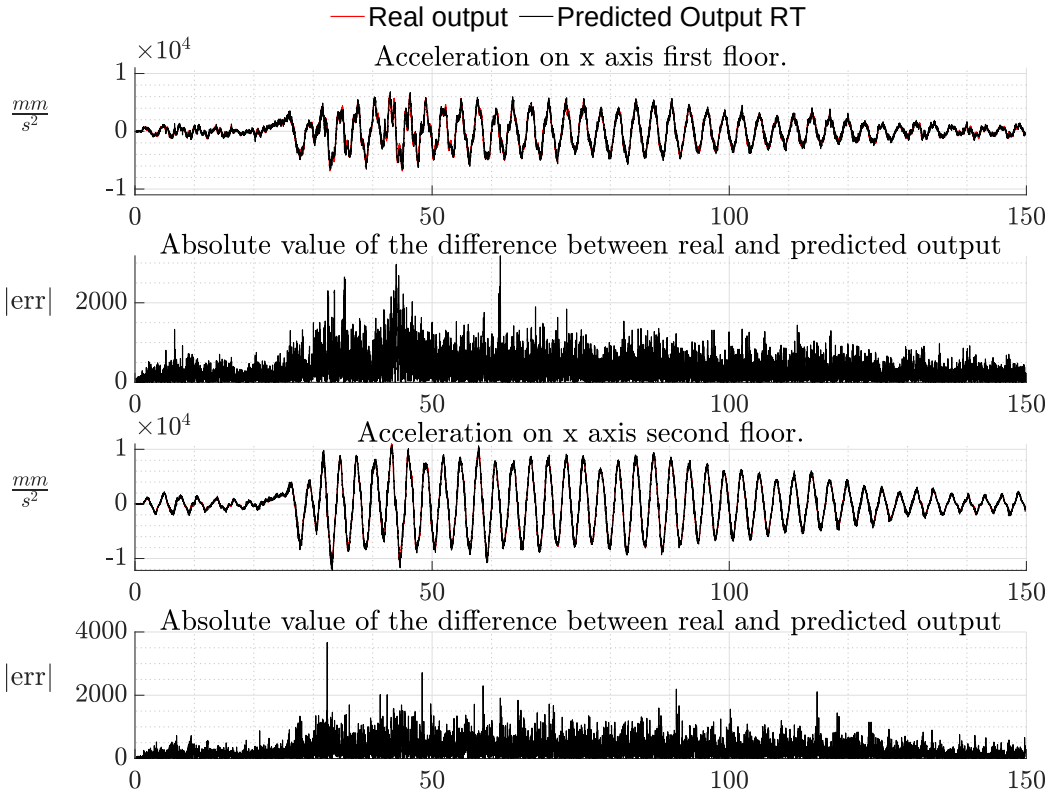

FIGURE 9 State predictions, i.e. State predictions, i.e. accelerations of the 2 floors, at $k + 10$, and associated absolute errors.

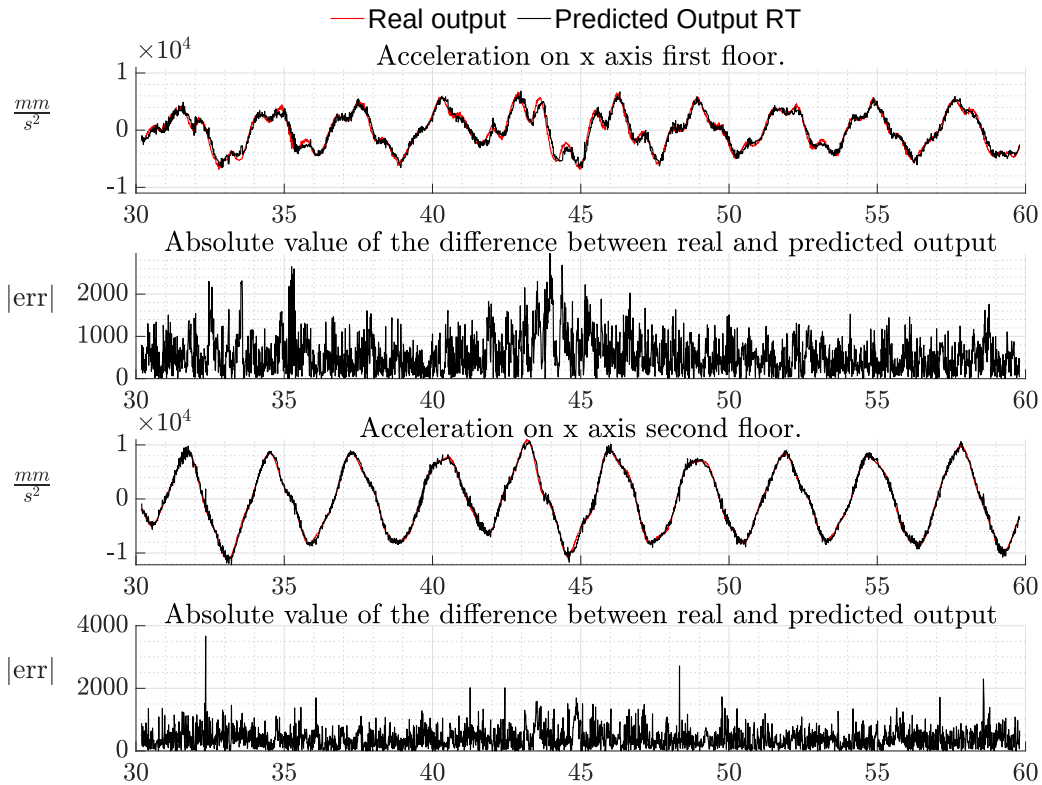

FIGURE 10 Zooms of state predictions, i.e. State predictions, i.e. accelerations of the 2 floors, at $k+10$, and associated absolute errors.

making the RT-based technique much more accurate and, as demonstrated in the next section, suitable for predictive control. Summarizing, in all these figures we can appreciate the quality of the approximation provided by the Data Driven approach. It is worth to remember that this is the prediction accuracy that we get only considering the information available at instant k, i.e. measurements of $x_{\mathsf{up}}(k)$ and $w(k)$, without any knowledge of the disturbance over the horizon. Some isolated spikes occur in the RT-based simulation: this issue is typical of the deterministic algorithm that rules RT, and can be mitigated by using, instead of Regression Trees, Random Forests, where the prediction is given by averaging the response different trees of the forest.

We also compared the two approaches in the frequency domain by computing the modes (Fourier transform) of the real response of the structure to a certain input and the responses of the SSI and RT model. Figure 12 shows the FFT of the 3 responses and confirms the comparison illustrated in the time domain: principal modes are caught from the identified models both in amplitude and in frequency. A better comparison would have been possible by comparing the impulsive responses or the responses to white noise: unfortunately we did not have measurements to make this comparison. We also did not have access to data of the structure with different symmetry configurations, but we expect similar results.

5.2 | Control performance

The results of the previous section show how the RT-based model provides a better prediction accuracy of the output of the system (accelerations on both floors) all over the horizon with respect to the SSI model. In this section we compare the control pewrformance by applying MPC using the two different models. Since the frame structure from which data were generated has been dismantled, we cannot run the control algorithms on the experimental setup. Consequently, in order to compare the control performance of the two different models with MPC, we run simulations considering as plant a *fair* simulation model identified via the Matlab system identification toolbox, as explained in Section 4.

In order to compare the control performance, we will compare the accelarations, the kinetic energy normalized with respect to the mass of the structure and the displacements. For the sake of clarity we will refer by RT-MPC and SSI-MPC to the systems whose control is based respectively on the RT- and SSI-based models. We implemented the control algorithm with cost functions given by $Q = 10^3 I_2$, $R = Q = 10^{-4} I_2$ and $N = 15$ for RT-MPC and $Q = 10^3 I_4$, $R = Q = 10^{-4} I_2$ for SSI-MPC. The dimensions of the weights matrices have been chosen consistently with the number of variables of the state space. Furthermore we assumed the structure was still at the beginning of the simulation (i.e. initial accelerations equals to 0) and no other disturbance, except the earthquake, is acting on the system.

Figure 13 shows the acceleration output of the simulated structure for RT-MPC and SSI-MPC: on average the acceleration of the RT-MPC is lower with respect to the output of the SSI-MPC and Figure 14 shows a zoom on the highest peaks of the outputs. Acceleration peaks of RT-MPC are reduced of about 20% on the first floor and about 30% on the second floor. It is worth to remark that, coherently with the prediction accuracy results, for $N = 1$ the performance of RT-MPC and SSI-MPC are comparable, because the prediction errors of both models at $k + 1$ are almost equal. For $N > 15$, simulation times increase consistently while the performance gain does not increase much, because of the prediction error that increase with N.

Figure 15 shows the cumulative Kinetic energy normalized with respect to the mass:

$$\begin{aligned} E_{RT-MPC}(k) &= \frac{1}{2} \sum_{t=1}^{k} v^2_{RT-MPC}(t) \\ E_{SSI-MPC}(k) &= \frac{1}{2} \sum_{t=1}^{k} v^2_{SSI-MPC}(t) \end{aligned} \tag{19}$$

The velocities are obtained by integrating numerically the accelerations of the output of both controlled systems. The performance of the RT-MPC with respect to the SSI-MPC is more evident from the energetic point of view, because the Kinetic Energy dissipated by the SSI-MPC is Significantly larger with respect to RT-MPC on both floors.

Figure 16 and 17 show how the performance of the RT-MPC compared with the performance of the SSI-MPC is better also in terms of damping of the velocity and the displacement of the structure.

6 | CONCLUSIONS

Exploiting the large amount of data available nowadays in many contexts can enable advanced and more efficient estimation and control techniques. In this paper we showed how this can be promising also in structural monitoring and civil engineering

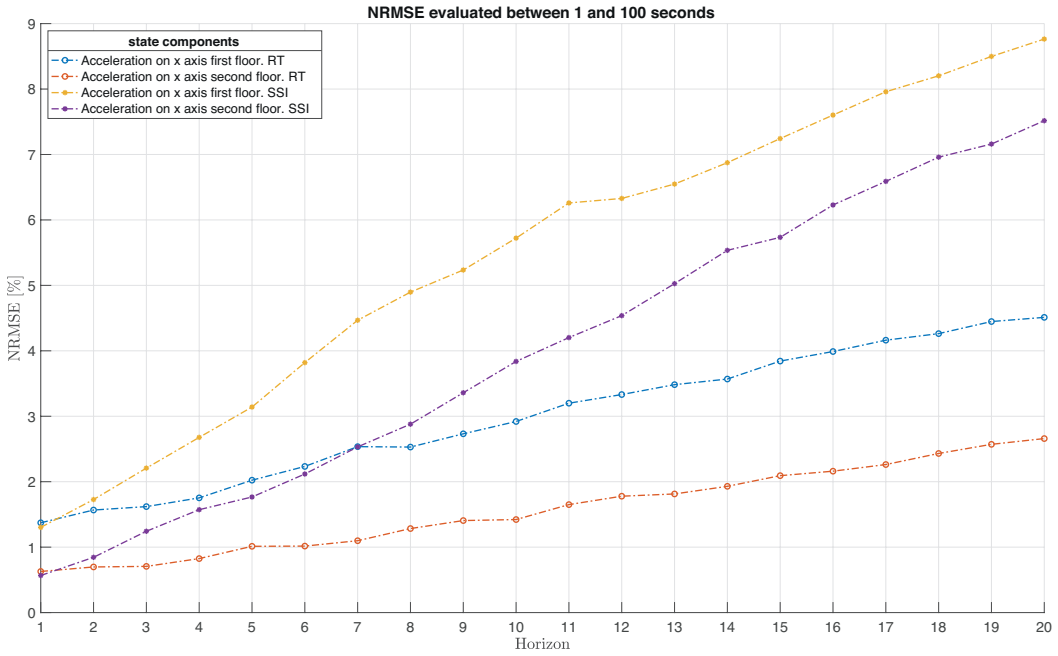

FIGURE 11 Normalized Root Mean Square Error of the accelerations along the predictive horizon. Regression Trees and Stochastic Subspace identification for each floor

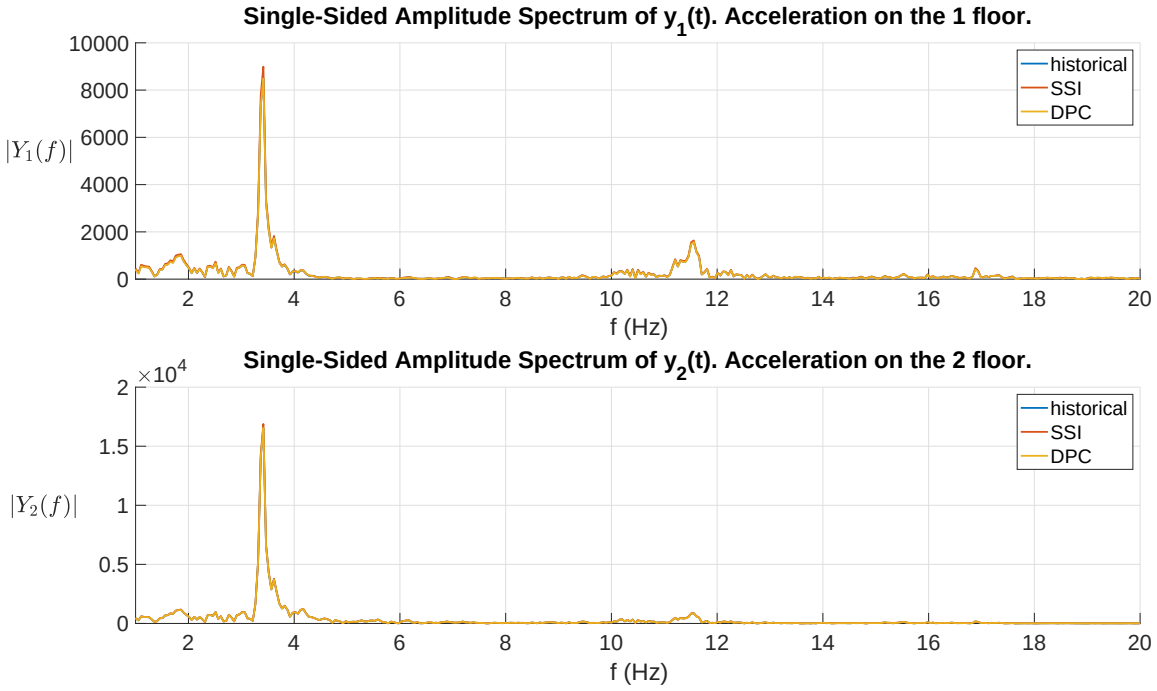

FIGURE 12 FFT evaluated in Matlab of the real response of the structure and the response of the SSI Model and the RT Model

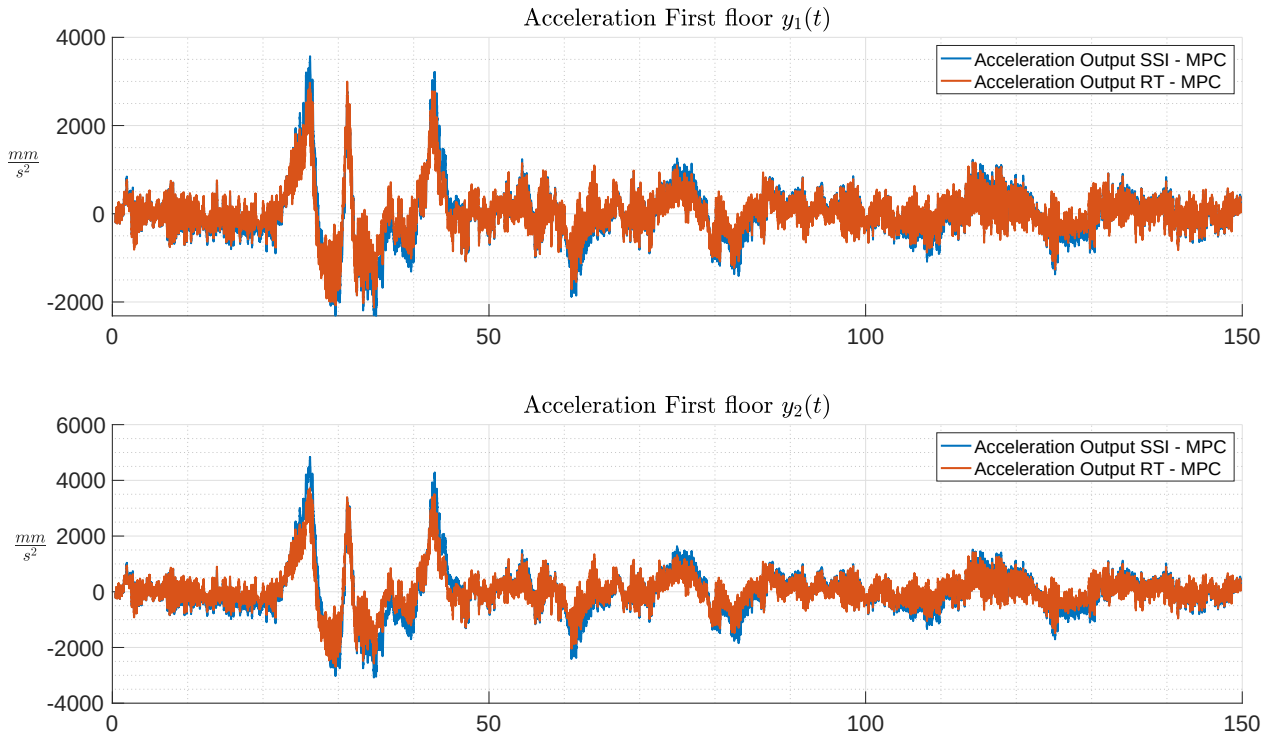

FIGURE 13 Top: acceleration comparison of RT-MPC and SSI-MPC for the first floor. Bottom left: acceleration comparison of RT-MPC and SSI-MPC for the second floor

applications. The novel machine learning algorithm developed in[46] and applied in this work has been able, starting from historical data, to identify a switching affine model that performed better than standard Sub-Space Identification (SSI) techniques in prediction accuracy on real experimental datasets, as well as in closed-loop simulation, showing how the kinetic energy dissipated in the control, and the displacement of the structure can be reduced consistently with respect to a control based on a dynamical model identified with standard SSI. We showed how both models perform equally if we consider a predictive horizon of 1 step, but things change when we increase the predictive horizon to the benefit of the RT model. We remark that our techniques envision that models can be periodically updated (e.g. every day/week/month) as new data incomes (i.e. when a new earthquake event occurs and/or in case of variations of the structure characteristics) to update the model and/or improve the accuracy. In this paper we first compare our technique and SSI in terms of prediction accuracy on the experimental data, then we compare the control performance in a simulative environment. We plan in future work to also compare control performance in real experiments on a frame.

Regression tree is not the only technique that can be exploited to identify a dynamical model: despite its simplicity, it suffer data variance and is prone to overfitting, especially when a tree is particularly deep. A standard way to reduce error due to both bias and variance is to exploit Random Forests, as shown in[46]: we leave the application of Random Forests to future work. Another venue for future work is to also consider a model for the future disturbances: this looks particularly appealing as the near future 5G communication technology (characterised by certified ultra-low latency and large bandwidth) can enable the identification and exploitation of models for the ground acceleration during an earthquake via a large network of geographically distributed sensors.

□

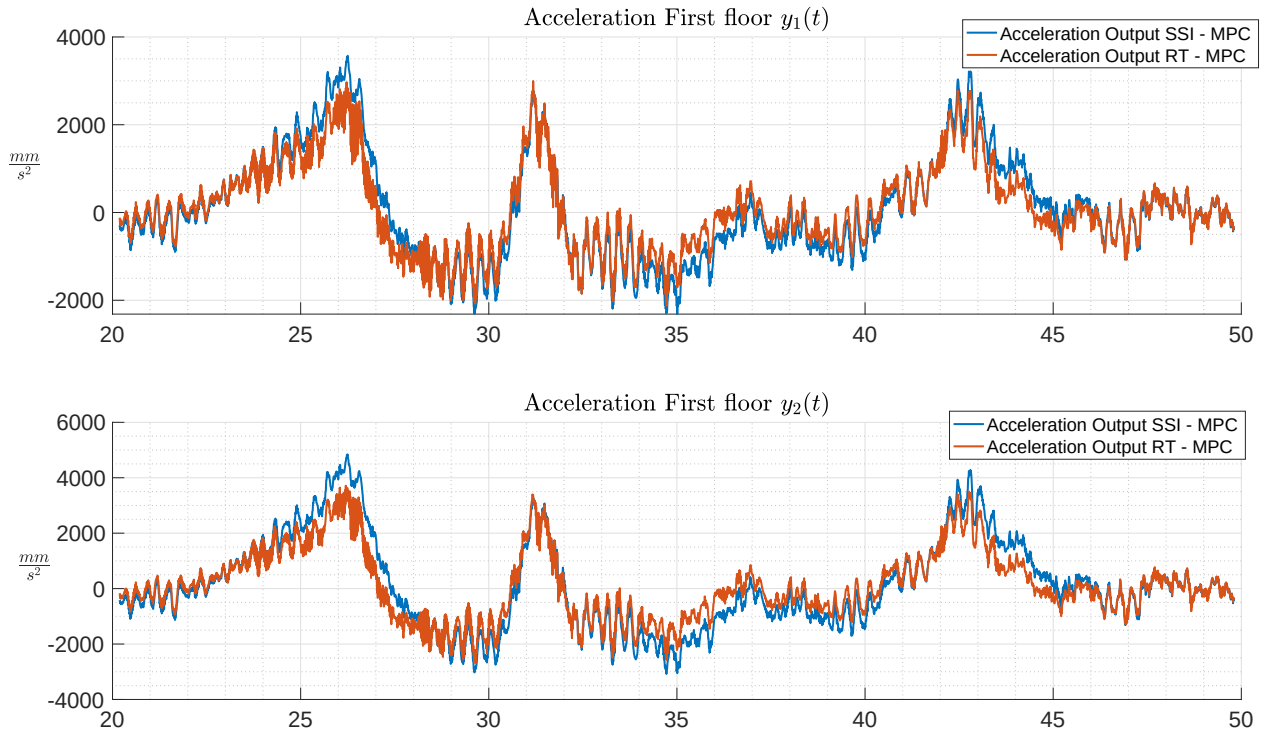

FIGURE 14 Top: focus on the highest peaks of acceleration of RT-MPC and SSI-MPC for the first floor. Bottom left: focus on the highest peak of acceleration of RT-MPC and SSI-MPC for the second floor

APPENDIX

A - REGRESSION TREES

In this appendix we explain how Regression Trees are built using an example adapted from[59]. Tree-based methods partition the feature space into a set of rectangles (more formally, hyper-rectangles) and then fit a simple model in each one. They are conceptually simple yet powerful. Let us consider a regression problem with continuous response $\mathcal{Y} = \{Y\}$ and 2 predictors $\mathcal{X} = \{X_1, X_2\}$, each taking values in the unit interval. The top left plot of Figure A1 shows a partition of the feature space by lines that are parallel to the coordinate axes. In each partition element, we can model Y with a different constant. However, there is a problem: although each partitioning line has a simple description like $X_1 = k$, some of the resulting regions are complicated to describe. To simplify things, we can restrict ourselves to only consider recursive binary partitions, like the ones shown in the top right plot of Figure A1. We first split the space into two regions, and model the response by the mean of Y in each region. We choose the variable and split-point to achieve the best prediction for Y. Then one or both of these regions are split into two more regions, and this process is continued, until some stopping rule is applied. This is the "recursive partitioning" part of the algorithm. For example, in the top right plot of Figure A1, we first split at $X_1 = t_1$. Then the region $X_1 \leq t_1$ is split at $X_2 = t_2$ and the region $X_1 > t_1$ is split at $X_1 = t_3$. Finally, the region $X_1 > t_3$ is split at $X_2 = t_4$. The result of this process is a partition of the data-space into the five regions (or leaves) $R_1, R_2, \cdots, R_5$. The corresponding regression tree model, $\mathcal{T}$, predicts Y with a constant, c_i, in region R_i i.e.,

$$f_{\mathcal{T}}(X_1, X_2) = \sum_{i=1}^{5} c_i I\left\{(X_1, X_2) \in R_i\right\}, \tag{A1}$$

where $I\{X \in R\}$ is a function that is equal to 1, if $X \in R$, and 0, otherwise. This same model can be represented by the binary tree shown in the bottom left of Figure A1. The full dataset sits at the top of the tree. Observations satisfying the condition at each node are assigned to the left branch, and the others to the right branch. The terminal nodes or leaves of the tree correspond to the regions $R_1, R_2, ..., R_5$. The bottom right plot of Figure A1, shows the perspective plot of the regression surface obtained as a result of building a regression tree with the 5 constants $c_i,\ i = 1, 2, 3, 4, 5$.

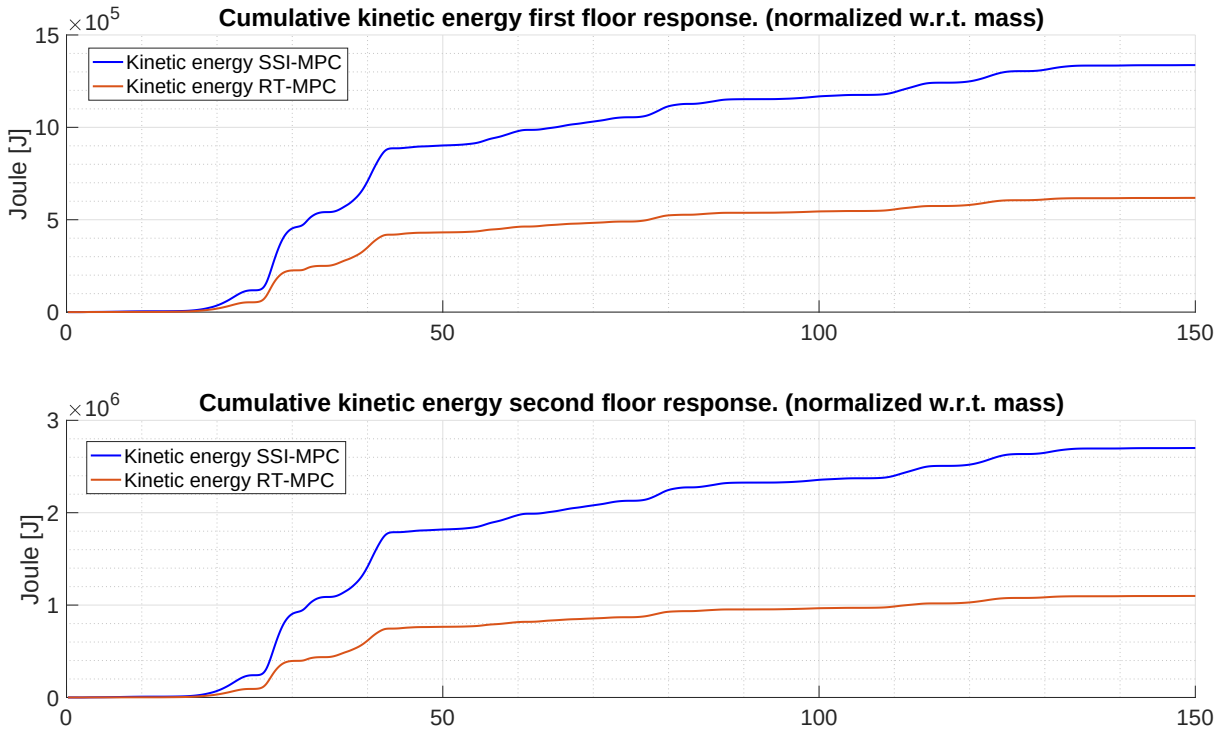

FIGURE 15 Kinetic energy normalized with respect to the mass. Top: first floor RT-MPC and SSI-MPC. Bottom: second floor RT-MPC and SSI-MPC.

Node splitting criteria

Now, a first question is: *how to grow a regression tree?* Suppose our dataset, $(\mathcal{X}, \mathcal{Y})$, consisting of p features, i.e. $\mathcal{X} = \{X_1, X_2, \dots, X_p\}$, and one response variable, i.e. $\mathcal{Y} = \{Y\}$. Suppose we have $|(\mathcal{X}, \mathcal{Y})| = n$ observations (samples): (x_i, y_i), $i = 1, 2, \dots, n$, with $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$. For regression trees we adopt the sum of squares as our splitting criteria, i.e. a variable at a node will be split if it minimizes the following sum of squares between the predicted response and the actual output variable:

$$\sum_i (y_i - f_{\mathcal{T}}(x_i))^2. \tag{A2}$$

The best response c_i (from equation A1 for the partition R_i), is just the average of output samples in the region R_i, i.e.

$$c_i = avg(y_i | x_i \in R_i). \tag{A3}$$

Finding the best binary partition in terms of minimum sum of squares is generally computationally infeasible. A greedy algorithm is used instead. Starting with all of the data, consider a splitting variable j and split point s, and define the following pair of left (R_L) and right (R_R) half-planes

$$\begin{aligned} R_L(j, s) &= \{X | X_j \le s\}, \\ R_R(j, s) &= \{X | X_j > s\} \end{aligned} \tag{A4}$$

The splitting variable j and the split point s is obtained by solving the following minimization:

$$\min_{j,s} \left[\min_{c_L} \sum_{x_i \in R_L(j,s)} (y_i - c_L)^2 + \min_{c_R} \sum_{x_i \in R_R(j,s)} (y_i - c_R)^2 \right] \tag{A5}$$

where, for any choice of j and s, the inner minimization in equation A5 is solved using

$$\begin{aligned} c_L &= \text{avg}(y_i | x_i \in R_L(j, s)), \\ c_R &= \text{avg}(y_i | x_i \in R_R(j, s)). \end{aligned} \tag{A6}$$

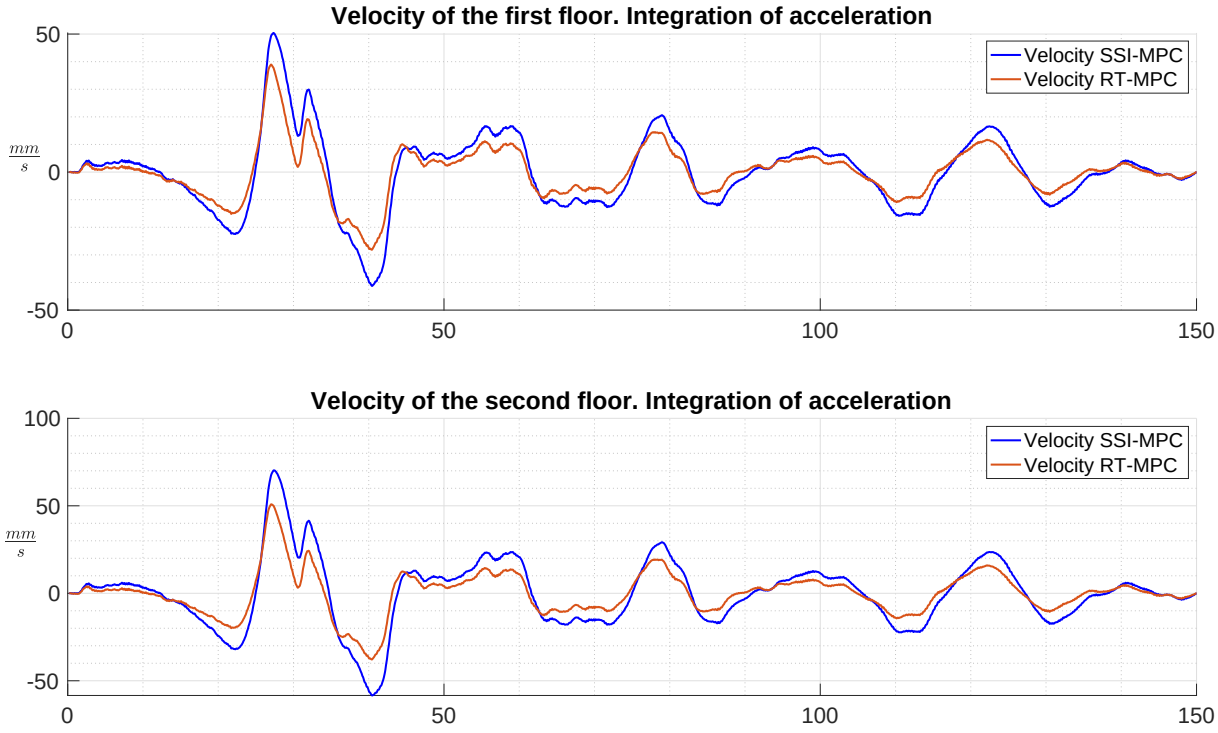

FIGURE 16 Velocities of the two floors of the structure. Top: first floor RT-MPC and SSI-MPC. Bottom: second floor RT-MPC and SSI-MPC.

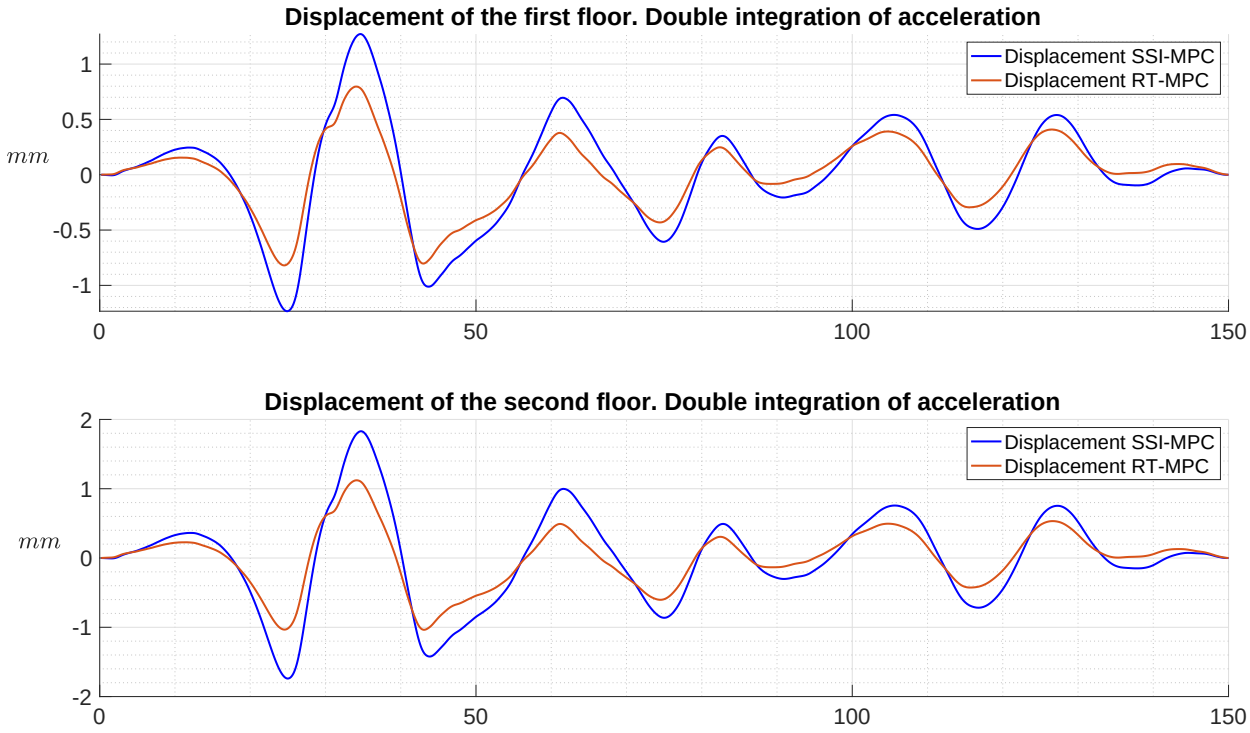

FIGURE 17 Displacements of the two floors of the structure. Top: first floor RT-MPC and SSI-MPC. Bottom: second floor RT-MPC and SSI-MPC.

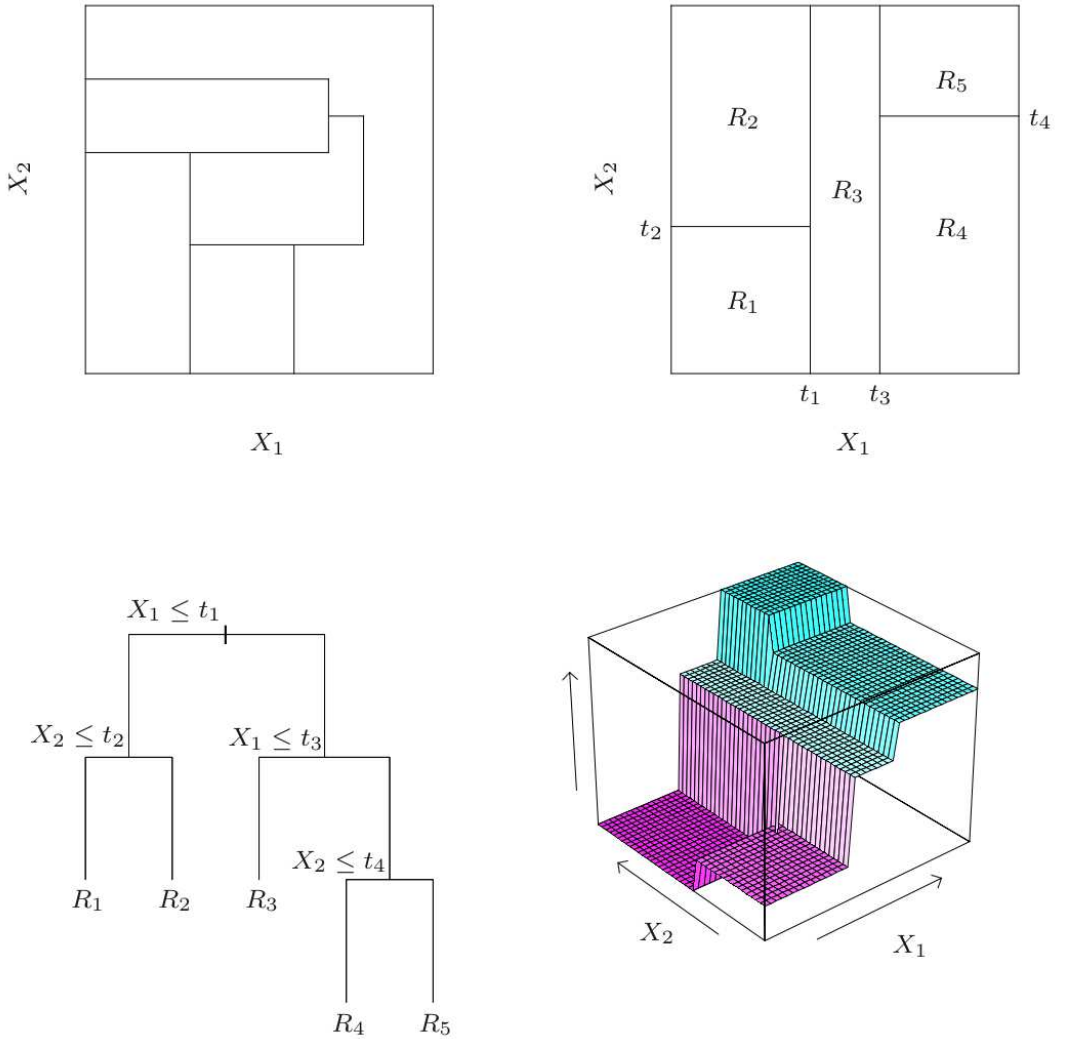

FIGURE A1 Top right: 2D feature space by recursive binary splitting. Top left: partition that cannot be obtained from recursive binary splitting. Bottom left: tree corresponding to the partition. Bottom right: perspective plot of the prediction surface.

For each splitting variable X_j, the determination of the split point s can be done very quickly and hence by scanning through all of the inputs (X_i's), the determination of the best pair (j, s) is feasible. Having found the best split, we partition the data into the two resulting regions and repeat the splitting process on each of the two regions. Then this process is repeated on all of the resulting regions.

Rather than splitting each node into just two regions at each stage, we might consider multiway splits into more than two groups. While this can sometimes be useful, it is not a good general strategy. The problem is that multiway splits fragment the data too quickly, leaving insufficient data at the next level down. Hence we would want to use such splits only when needed. Also multiway splits can be achieved by a series of binary splits.

Stopping criteria and pruning

At this point, the second question is: *How large should we grow the tree?* Every recursive algorithm needs to know when it's done, i.e. it requires a stopping criteria. For regression trees this means when to stop splitting the nodes. A very large tree might over fit the data, while a small tree might not capture the important structure. Tree size is a tuning parameter governing the modelâĂŹs complexity, and the optimal tree size should be adaptively chosen from the data. One approach is to split tree nodes only if the decrease in sum-of-squares due to the split exceeds some threshold. However, this strategy is myopic, since a seemingly worthless split might lead to a very good split below it. A preferred strategy is to grow a large tree, stopping the splitting process only when some minimum number of data points at a node (`MinLeaf`) is reached. Then this large tree is pruned using cost-complexity pruning methods.

Define a subtree $\mathcal{T}_{sub} \subset \mathcal{T}$ to be any tree that can be obtained by pruning $\mathcal{T}$, i.e. collapsing any number of its non-terminal nodes. Let node i corresponding to the partition R_i. $|\mathcal{T}_{sub}|$ denotes the number of terminal nodes in $\mathcal{T}_{sub}$ Define,

$$\begin{aligned} N_i &= \#\left\{x_i \in R_i\right\}, \\ c_i &= \frac{1}{N_i} \sum_{x_i \in R_i} y_i, \\ Q_i(T) &= \frac{1}{N_i} \sum_{x_i \in R_i} (y_i - c_i)^2 \end{aligned} \tag{A7}$$

where N_i is the number of samples in the partition R_i, c_i is the estimate of Y within R_i and $Q_i(T)$ is the mean square error of the estimate c_i. The cost complexity criteria is then defined as:

$$C_\alpha(\mathcal{T}_{sub}) = \sum_{i=1}^{|\mathcal{T}_{sub}|} N_i Q_i(T) + \alpha |\mathcal{T}_{sub}| \tag{A8}$$

The goal is to find, for each α, the subtree $\mathcal{T}_\alpha \subset \mathcal{T}$ to minimize $C_\alpha(\mathcal{T}_{sub})$. The tuning parameter $\alpha \geq 0$ governs the trade off between tree size and its goodness of fit to the data. For each α one can show that there is a unique smallest subtree $\mathcal{T}_\alpha$ that minimizes $C_\alpha(\mathcal{T}_{sub})$[60]. Estimation of α is achieved by cross-validation.

References

1. Soong T, Spencer Jr B. Supplemental energy dissipation: state-of-the-art and state-of-the-practice. *Engineering structures* 2002; 24(3): 243–259.
2. Spencer Jr B, Nagarajaiah S. State of the art of structural control. *Journal of structural engineering* 2003; 129(7): 845–856.
3. Gattulli V, RC L, TT S. Nonlinear control laws for enhancement of structural control effectiveness. In: ; 1994: 971–980.
4. Ceci AM, Gattulli V, Potenza F. Serviceability and damage scenario in irregular RC structures: post-earthquake observations and modeling predictions. *Journal of Performance of Constructed Facilities* 2011; 27(1): 98–115.
5. Soong TT. Active structural control. *Longman Scientific and Technical* 1990.
6. Soong TT, Dargush GF. Passive energy dissipation systems in structural engineering. 1997.
7. Symans MD, Constantinou MC. Semi-active control systems for seismic protection of structures: a state-of-the-art review. *Engineering structures* 1999; 21(6): 469–487.
8. Preumont A, Seto K. *Active control of structures*. John Wiley & Sons . 2008.
9. Achkire Y, Preumont A. Active tendon control of cable-stayed bridges. *Earthquake engineering & structural dynamics* 1996; 25(6): 585–597.
10. Gattulli V, Alaggio R, Potenza F. Analytical prediction and experimental validation for longitudinal control of cable oscillations. *International Journal of Non-Linear Mechanics* 2008; 43(1): 36–52.
11. Gattulli V, Romeo F. Integrated procedure for identification and control of MDOF structures. *Journal of engineering mechanics* 2000; 126(7): 730–737.
12. Gattulli V, Ghanem R. Adaptive control of flow-induced oscillations including vortex effects. *International Journal of Non-Linear Mechanics* 1999; 34(5): 853–868.
13. Bitaraf M, Hurlebaus S, Barroso LR. Active and Semi-active Adaptive Control for Undamaged and Damaged Building Structures Under Seismic Load. *Computer-Aided Civil and Infrastructure Engineering* 2012; 27(1): 48-64.
14. Jansen LM, Dyke SJ. Semiactive control strategies for MR dampers: comparative study. *Journal of Engineering Mechanics* 2000; 126(8): 795–803.

15. Yanik A, Aldemir U, Bakioglu M. A new active control performance index for vibration control of three-dimensional structures. *Engineering Structures* 2014; 62: 53–64.

16. Miyamoto K SJ. A new performance index of LQR for combination of passive base isolation and active structural control.. *Engineering Structures.* 201; 157: 280-299.

17. Bleicher A FYST. Model-based design and experimental validation of active vibration control for a stress ribbon bridge using pneumatic muscle actuators.. *Engineering Structures.* 2011; 33:: 2237-2247.

18. Feng X, Ou Y, Miah MS. Energy-based comparative analysis of optimal active control schemes for clustered tensegrity structures. *Structural Control and Health Monitoring* 2018; 25(10): e2215.

19. Cruz Neto HJ, Trindade MA. On the noncollocated control of structures with optimal static output feedback: Initial conditions dependence, sensors placement, and sensitivity analysis. *Structural Control and Health Monitoring* 2019; 26(10): e2407.

20. Camacho EF, Alba CB. *Model predictive control*. Springer Science & Business Media . 2013.

21. López-Almansa F, Andrade R, Rodellar J, Reinhorn AM. Modal Predictive Control Of Structures. I: Formulation. *Journal of Engineering Mechanics* 1994; 120(8): 1743-1760.

22. Mei G, Kareem A, Kantor JC. Real time model predictive control of structures under earthquakes. In: ; 2001.

23. Mei G, Kareem A, Kantor JC. Model Predictive Control Of Structures Under Earthquake Using Acceleration Feedback. *Structural Control and Health Monitoring* 2018.

24. F. Oliveira PM, Suleman A. Predictive Control for Earthquake Response Mitigation of Buildings Using Semiactive Fluid Dampers. *Shock and Vibration* 2014.

25. Khodabandehlou H, Pakcan G, Fadali MS, Salem MMA. Active neural predictive control of seismically isolated structures. *Structural Control and Health Monitoring* 2018; 25: e2061.

26. Ning XL, Tan P, Huang DY, Zhou FL. Application of adaptive fuzzy sliding mode control to a seismically excited highway bridge. *Structural Control and Health Monitoring* 2009; 16(6): 639-656.

27. Anh ND, Bui HL, Vu NL, Tran DT. Application of hedge algebra-based fuzzy controller to active control of a structure against earthquake. *Structural Control and Health Monitoring* 2013; 20(4): 483-495.

28. Gu X, Yu Y, Li Y, Li J, Askari M, Samali B. Experimental study of semi-active magnetorheological elastomer base isolation system using optimal neuro fuzzy logic control. *Mechanical Systems and Signal Processing* 2019; 119: 380–398.

29. Khalatbarisoltani A, Soleymani M, Khodadadi M. Online control of an active seismic system via reinforcement learning. *Structural Control and Health Monitoring* 2019; 26(3): e2298.

30. Dworakowski Z, Mendrok K. Reinforcement learning for vibration suppression of an unknown system. In: Uhl T. , ed. *Advances in Mechanism and Machine Science*Springer International Publishing; 2019; Cham: 4045–4054.

31. Barbieri M, Ilanko S, Pellicano F. Active vibration control of seismic excitation. *Nonlinear Dynamics* 2017; 25(10): 41-52.

32. De Iuliis V, Germani A, Manes C. Identification of Forward and Feedback Transfer Functions in Closed-Loop Systems with Feedback Delay. *IFAC-PapersOnLine* 2017; 50(1): 12847–12852.

33. Gattulli V. *Identification Methods for Structural Health Monitoring*. 567 of *Springer Verlag*. Eleni Chatzi, Costas Papadimitriou, Book CISM International Centre for Mechanical Sciences . 2016.

34. Suryanita R, Maizir H, Jingga H. Prediction Of Structural Response Due To Earthquake Load Using Artificial Neural Networks. In: ; 2016.

35. Jeng CH, Mo YL. Quick Seismic Response Estimation of Prestressed Concrete Bridges Using Artificial Neural Networks. *Journal of Computing in Civil Engineering* 2004; 18(4): 360-372.

36. Chang CM, Lin TK, Chang CW. Applications of neural network models for structural health monitoring based on derived modal properties. *Measurement* 2018; 129: 457 - 470. doi: https://doi.org/10.1016/j.measurement.2018.07.051

37. Parsaeimaram M, Poursalehi K. An Artificial Neural Network for prediction of Seismic Behavior in RC Buildings with and Without Infill Walls. 2018.

38. Abd-Elhamed A, Shaban Y, Mahmoud S. Predicting Dynamic Response of Structures under Earthquake Loads Using Logical Analysis of Data. *Buildings* 2018; 8(4).

39. Qing Wang XH, Zhang L. Semiactive Nonsmooth Control for Building Structure with Deep Learning. *Complexity* 2017.

40. Macarulla M, Casals M, Forcada N, Gangolells M. Implementation of predictive control in a commercial building energy management system using neural networks. *Energy and Buildings* 2017; 151: 511–519.

41. Ferreira P, Ruano A, Silva S, Conceicao E. Neural networks based predictive control for thermal comfort and energy savings in public buildings. *Energy and Buildings* 2012; 55: 238–251.

42. Jain A, Smarra F, Behl M, Mangharam R. Data-driven model predictive control with regression trees—An application to building energy management. *ACM Transactions on Cyber-Physical Systems* 2018; 2(1): 4.

43. Jain A, Smarra F, Mangharam R. Data predictive control using regression trees and ensemble learning. In: Decision and Control (CDC), 2017 IEEE 56th Annual Conference on IEEE. ; 2017: 4446–4451.

44. Behl M, Smarra F, Mangharam R. DR-Advisor: A data-driven demand response recommender system. *Applied Energy* 2016; 170: 30–46.

45. Smarra F, Jain A, de Rubeis T, Ambrosini D, D'Innocenzo A, Mangharam R. Data-driven model predictive control using random forests for building energy optimization and climate control. *Applied Energy* 2018.

46. Smarra F, Jain A, Mangharam R, D'Innocenzo A. Data-driven Switched Affine Modeling for Model Predictive Control. *Analysis and Design of Hybrid Systems (ADHS'18), IFAC-PapersOnLine* 2018; 51(16): 199–204.

47. Afram A, Janabi-Sharifi F, Fung AS, Raahemifar K. Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system. *Energy and Buildings* 2017; 141: 96–113.

48. Jain A, Nghiem T, Morari M, Mangharam R. Learning and control using Gaussian processes. In: IEEE. ; 2018: 140–149.

49. Breiman L. *Classification and regression trees.* Routledge . 2017.

50. Ponzo FC, Di Cesare A, Nigro D, et al. Jet-pacs project: Dynamic experimental tests and numerical results obtained for a steel frame equipped with hysteretic damped chevron braces. 2012; 16: 662-685.

51. Gattulli V, Lepidi M, Potenza F. Seismic protection of frame structures via semi-active control: modeling and implementation issues. *Earthquake Engineering and Engineering Vibration* 2010; 4(4): 627–645.

52. Antonacci E, De Stefano A, Gattulli V, Lepidi M, Matta E. Comparative study of vibration-based parametric identification techniques for a three-dimensional frame structure. 2012; 19: 579-608.

53. Larimore WE. Canonical variate analysis in identification, filtering, and adaptive control. 1990: 596-604 vol.2.

54. VERHAEGEN M, DEWILDE P. Subspace model identification Part 1. The output-error state-space model identification class of algorithms. *International Journal of Control* 1992; 56(5): 1187-1210.

55. Xie Y, Shi H, Bi F, Shi J. N4SID Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems. *Automatica* 1994; 30: 75–93.

56. Jamaludin IW, Wahab NA, Khalid NS, Sahlan S, Ibrahim Z, Rahmat MF. N4SID and MOESP subspace identification methods. 2013: 140-145.

57. Ljung L. *System Identification: Theory for the User*. Prentice Hall information and system sciences seriesPrentice Hall PTR . 1999.

58. Wills A, Schön TB, Ninness B. Estimating state-space models in innovations form using the expectation maximisation algorithm. 2010: 5524-5529.

59. Hastie T, Tibshirani R, Friedman J. The elements of statistical learning: prediction, inference and data mining. *Springer-Verlag, New York* 2009.

60. Ripley BD. *Pattern recognition and neural networks*. Cambridge university press . 2007.