2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Controller implementability: a data-driven approach

Alberto Padoan, Jeremy Coulson, and Florian Dörfler

Abstract— **We study the controller implementability problem, which seeks to determine if a controller can make the closed-loop behavior of a given plant match that of a desired reference behavior. We establish necessary and sufficient conditions for controller implementability which only rely on raw data. Subsequently, we consider the problem of constructing controllers directly from data. By leveraging the concept of canonical controller, we provide a formula to directly construct controllers that implement plant-compatible reference behaviors using measurements of both reference and plant behaviors.**

I. Introduction

The problem of control design can be split into three parts: (i) to describe the set of admissible controllers; (ii) to describe the properties that the controlled system should have; and (iii) to find an admissible controller such that the resulting controlled behavior has desired properties [1]. Generally, in order to solve a control design problem, one needs access to a model of the system to be controlled and a model of the reference behavior. However, in many situations of practical interest obtaining such models is expensive, time-consuming, or simply impossible, and the control designer only has access to measured data [2]. This has motivated the development of a wide range of new direct data-driven control methods that bypass system identification and aim to compute controllers directly from data, see, *e.g.*,the recent survey [2].

The paper studies the controller implementability problem [3]. The problem is to find, if possible, a controller which makes the closed-loop behavior of a plant equal to that of a desired reference behavior. While the problem does admit a model-based solution, our objective is to provide an alternative solution that is compatible with modern data-driven approaches. Using the language of behavioral systems theory [4], we regard finite-horizon behaviors of finite-dimensional, linear, time-invariant (LTI) systems as subspaces represented by raw data matrices [2]. We establish necessary and sufficient conditions for controller implementability which can be tested directly from raw data. Furthermore, we also consider the problem of constructing controllers directly from data. We provide a formula to directly construct controllers that implement plant-compatible reference behaviors using only measurements of the reference and plant behaviors.

Contributions: The contributions of the paper are twofold. We establish new necessary and sufficient conditions for solving the controller implementability problem, thus characterizing all implementable controlled behaviors in both model-based and data-driven scenarios. We provide a formula for a canonical controller that implements any given reference behavior, whenever this is possible; the controller depends solely on the reference and plant behaviors and can be directly obtained from data.

Related work: The controller implementability problem has been originally studied in [3], where necessary and sufficient conditions for implementability are given for continuous-time behaviors. The concept of a canonical controller has been implicitly defined in the seminal paper [3] and, subsequently, formalized for general systems, e.g., in [5] and [6]. Our results extend existing results presented in [3 , 5 , 6] to discrete-time, finite-horizon behaviors, eliminating the need for parametric models and enabling the direct use of raw data. Over the past two decades, the data-driven approaches have received increasing attention, primarily due to the surge in availability of data, see, *e.g.* the recent survey [2]. A simple, yet paradigmatic instance of the controller implementability problem is the *exact model matching* problem [7 , 8], whereby one seeks a state feedback law for a given finite-dimensional LTI system to make the closed-loop transfer function equal to a given transfer function. The problem has been widely studied in a model-based context [7 – 9] and it is well-known that the problem can be reduced to solving a set of linear algebraic equations [8 , 9]. The recent paper [10] presents analogous findings in a data-driven context. Our results generalize the findings of [10] in a representation-free setting, without requiring plant and reference to have the same order, the controller to be static, or state measurements to be available.

Paper organization: Section II provides preliminary results from behavioral systems theory. Section III formalizes the data-driven controller implementability problem. Section IV contains the main results of the paper, including necessary and sufficient implementability conditions which rely only on raw data and a formula for the direct data-driven construction of controllers that implement any plant-compatible reference behavior. Section V provides a summary of our main results and an outlook to future research directions. The proofs of our main results are deferred to the appendix.

Notation: The set of positive integers is denoted by $\mathbb{N}$. The set of real numbers is denoted by $\mathbb{R}$. For $T \in \mathbb{N}$, the set of integers $\{1, 2, \ldots, T\}$ is denoted by $\mathbf{T}$. The image, kernel, and Moore-Penrose inverse of the matrix $M \in \mathbb{R}^{p\times m}$ are denoted by $\operatorname{im} M$, $\ker M$, and $M^\dagger$, respectively. A map f from X to Y is denoted by $f : X \to Y$; $(Y)^X$ denotes the collection of all such maps. The inverse image of the set Y under f is denoted by $f^{-1}(Y)$.

A. Padoan and F. Dörfler are with the Department of Information Technology and Electrical Engineering at ETH Zürich, Zürich, Switzerland {apadoan, dorfler}@control.ee.ethz.ch. J. Coulson, is with the Department of Electrical and Computer Engineering at the University of Wisconsin-Madison, USA. jeremy.coulson@wisc.edu.

II. Preliminary results

This section recalls key notions and results from behavioral systems theory [4], with a focus on discrete-time LTI systems.

A. Time series and Hankel matrices

We use the terms *time series* and *trajectory* interchangeably. The set of time series $w = (w(1), \ldots, w(T))$ of length $T \in \mathbb{N}$, with $w(t) \in \mathbb{R}^q$ for $t \in \mathbf{T}$, is defined as $(\mathbb{R}^q)^{\mathbf{T}}$. The set of infinite-length time series $w = (w(1), w(2), \ldots)$, with $w(t) \in \mathbb{R}^q$ for $t \in \mathbb{N}$, is defined as $(\mathbb{R}^q)^{\mathbb{N}}$.

1) The cut operator. Restricting time series over subintervals gives rise to the cut operator. Formally, given $w \in (\mathbb{R}^q)^{\mathbf{T}}$ and $L \in \mathbf{T}$, the *cut operator* is defined as

$$w|_L = (w(1), \ldots, w(L)) \in (\mathbb{R}^q)^{\mathbf{L}}.$$

For infinite-length time series, the definition holds verbatim with $w \in (\mathbb{R}^q)^{\mathbb{N}}$ and $L \in \mathbb{N}$. Applied to a set of time series $\mathcal{W} \subseteq (\mathbb{R}^q)^{\mathbf{T}}$ or $\mathcal{W} \subseteq (\mathbb{R}^q)^{\mathbb{N}}$, the cut operator acts on all time series, defining the *restricted* set $\mathcal{W}|_L = \{w|_L \,:\, w \in \mathcal{W}\}$. By a convenient abuse of notation, we identify the trajectory $w|_L$ with the corresponding vector $(w(1), \ldots, w(L)) \in \mathbb{R}^{qL}$.

2) The shift operator. Shifting elements of time series gives rise to the shift operator. Formally, given $w \in \mathbb{R}^{qT}$ and $\tau \in \mathbf{T}$, the *shift operator* is defined as

$$\sigma^{\tau-1} w = (w(\tau), \ldots, w(T)) \in \mathbb{R}^{q(T-\tau+1)}.$$

For infinite-length time series, the shift operator is defined as $w \mapsto \sigma^{\tau-1} w$, with $\sigma^{\tau-1} w(t) = w(t+\tau-1)$, for any $\tau \in \mathbb{N}$. Applied to a set of time series $\mathcal{W} \subseteq \mathbb{R}^{qT}$ or $\mathcal{W} \subseteq (\mathbb{R}^q)^{\mathbb{N}}$, the shift operator acts on all time series in the set giving rise to the *shifted* set $\sigma^\tau \mathcal{W} = \{\sigma^\tau w \,:\, w \in \mathcal{W}\}$.

3) Hankel matrices. The *Hankel matrix* of depth $L \in \mathbf{T}$ associated with the time series $w \in \mathbb{R}^{qT}$ is defined as

$$H_L(w) = \begin{bmatrix} w(1) & w(2) & \cdots & w(T-L+1) \\ w(2) & w(3) & \cdots & w(T-L+2) \\ \vdots & \vdots & \ddots & \vdots \\ w(L) & w(L+1) & \cdots & w(T) \end{bmatrix}.$$

B. Discrete-time LTI dynamical systems

A *dynamical system* (or, briefly, *system*) is a triple $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$, where $\mathbb{T}$ is the *time set*, $\mathbb{W}$ is the *signal space*, and $\mathcal{B} \subseteq (\mathbb{W})^{\mathbb{T}}$ is the *behavior* of the system. We exclusively focus on *discrete-time* systems, with $\mathbb{T} = \mathbb{N}$ and $\mathbb{W} = \mathbb{R}^q$.

1) Finite-dimensional LTI systems. A system $\mathcal{B}$ is *linear* if $\mathcal{B}$ is a linear subspace, *time-invariant* if $\mathcal{B}$ is shift-invariant, *i.e.*, $\sigma^{\tau-1}(\mathcal{B}) \subseteq \mathcal{B}$ for all $\tau \in \mathbb{N}$, and *complete* if $\mathcal{B}$ is closed in the topology of pointwise convergence [4, Proposition 4]. The model class of all complete LTI systems is denoted by $\mathfrak{L}^q$. By a convenient abuse of notation, we write $\mathcal{B} \in \mathfrak{L}^q$.

2) Kernel representations. Every finite-dimensional LTI system $\mathcal{B} \in \mathfrak{L}^q$ admits a *kernel representation* of the form

$$\mathcal{B} = \ker R(\sigma),$$

where the operator $R(\sigma)$ is defined by the polynomial matrix $R(z) = R_0 + R_1 z + \ldots + R_\ell z^\ell$, with $R_i \in \mathbb{R}^{p \times q}$ for $i \in \boldsymbol{\ell}$, and the set $\ker R(\sigma)$ is defined as $\{w \,:\, R(\sigma)w = 0\}$. Without loss of generality, we assume that $\ker R(\sigma)$ is a *minimal* kernel representation of $\mathcal{B}$, *i.e.*, p is as small as possible over all kernel representations of $\mathcal{B}$.

3) Integer invariants of an LTI system. The structure of an LTI system $\mathcal{B} \in \mathfrak{L}^q$ is characterized by a set of integer invariants [4, Section 7], defined as

- the *number of inputs* $m(\mathcal{B}) = q - \text{row dim} R$,
- the *number of outputs* $p(\mathcal{B}) = \text{row dim} R$,
- the *lag* $\ell(\mathcal{B}) = \max_{i \in \mathbf{p}} \{\deg \text{row}_i R\}$, and
- the *order* $n(\mathcal{B}) = \sum_{i \in \mathbf{p}} \deg \text{row}_i R$,

where $\ker R(\sigma)$ is a minimal kernel representation of $\mathcal{B}$, while row $\dim R$ and $\deg \text{row}_i R$ are the number of rows and the degree of the i-th row of $R(z)$, respectively. The integer invariants are intrinsic properties of a system, as they do not depend on its representation [11, Proposition X.3].

4) Partitions. Given a permutation matrix $\Pi \in \mathbb{R}^{q \times q}$ and an integer $0 < m < q$, the map

$$(u, y) = \Pi^{-1} w \tag{1}$$

defines a *partition* of $w \in \mathbb{R}^q$ into the variables $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^{q-m}$. We write $w \sim (u, y)$ if (1) holds for some permutation matrix $\Pi \in \mathbb{R}^{q \times q}$ and integer $0 < m < q$. Any partition (1) induces the natural projections $\pi_u : w \mapsto u$ and $\pi_y : w \mapsto y$. We call (u, y) a *partition of* $\mathcal{B} \in \mathfrak{L}^q$ if (1) holds for all $w \in \mathcal{B}$.

5) State-space representations. Every finite-dimensional LTI system $\mathcal{B} \in \mathfrak{L}^q$ can be described by the equations

$$\sigma x = Ax + Bu, \quad y = Cx + Du, \tag{2}$$

and admits a (*minimal*) *input/state/output representation*

$$\mathcal{B} = \left\{ (u, y) \in (\mathbb{R}^q)^{\mathbb{N}} \,:\, \exists\, x \in (\mathbb{R}^n)^{\mathbb{N}} \text{ s.t. (2) holds} \right\}, \tag{3}$$

where $\left[\begin{smallmatrix} A & B \\ C & D \end{smallmatrix}\right] \in \mathbb{R}^{(n+p)\times(n+m)}$ and m, n, and p are the number of inputs, the order, and the number of outputs of $\mathcal{B}$, respectively.

C. Data-driven representations of LTI systems

The restricted behavior of a finite-dimensional, discrete-time, LTI system can be represented as the image of a raw data matrix. We summarize a version of this principle known as the *fundamental lemma* [12].

Lemma 1. [13, Corollary 19] Let $\mathcal{B} \in \mathfrak{L}^q$ and $w \in \mathcal{B}|_T$. Assume $\ell(\mathcal{B}) < L \leq T$. Then $\mathcal{B}|_L = \operatorname{im} H_L(w)$ if and only if

$$\operatorname{rank} H_L(w) = m(\mathcal{B})L + n(\mathcal{B}). \tag{4}$$

The rank condition (4) is referred to as the *generalized persistency of excitation* condition [13]. Thus, we call a trajectory $w \in \mathcal{B}|_T$ of a system $\mathcal{B} \in \mathfrak{L}^q$ *generalized persistently exciting (GPE) of order* L if (4) holds. Different variations of this principle can be formulated under a range of assumptions, see, *e.g.*, the recent survey [2] for an overview.

III. Problem formulation

Consider a *plant behavior* $\mathcal{P} \in \mathfrak{L}^{q+k}$, a *reference behavior* $\mathcal{R} \in \mathfrak{L}^q$, and a *controller behavior* $\mathcal{C} \in \mathfrak{L}^k$, as shown in Fig. 1.

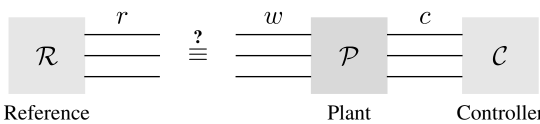

Fig. 1: Control in a behavioral setting.

Following [3], we fix a partition of the variables of the plant behavior $\mathcal{P}$, which induces the natural projections

$$\pi_w : (w, c) \mapsto w, \quad \pi_c : (w, c) \mapsto c, \tag{5}$$

where w are the *to-be-controlled* variables and c are the *control* variables, respectively. The controller behavior $\mathcal{C}$ is interconnected to the plant behavior $\mathcal{P}$ via *variable sharing*. Formally, the *interconnection of* $\mathcal{P}$ *and* $\mathcal{C}$ *via the shared variable* c is defined as

$$\mathcal{P}\|_c\mathcal{C} = \left\{(w, c) \in (\mathbb{R}^{q+k})^{\mathbb{N}} \, : \, c \in \mathcal{C}, \; (w, c) \in \mathcal{P}\right\}. \tag{6}$$

Similarly, we define the *hidden behavior* $\mathcal{N}$ of the plant behavior $\mathcal{P}$ as

$$\mathcal{N} = \left\{w \in (\mathbb{R}^{q+k})^{\mathbb{N}} \, : \, (w, 0) \in \mathcal{P}\right\}. \tag{7}$$

We refer to $\pi_w(\mathcal{P}\|_c\mathcal{C})$ and $\pi_w(\mathcal{P})$ as the *controlled plant behavior* and *uncontrolled plant behavior*, respectively. Fig. 2 offers a pictorial illustration of the aforementioned behaviors.

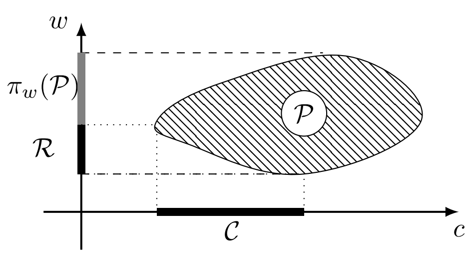

Fig. 2: Pictorial illustration of behaviors $\mathcal{P}$, $\mathcal{R}$, $\mathcal{C}$, and $\pi_w(\mathcal{P})$.

A controller $\mathcal{C} \in \mathfrak{L}^k$ is said to *implement* $\mathcal{R} \in \mathfrak{L}^q$ if $\pi_w(\mathcal{P}\|_c\mathcal{C}) = \mathcal{R}$ [3]. In other words, a controller behavior implements a given reference behavior if the resulting controlled plant behavior obtained from interconnecting the plant with the controller coincides with the reference behavior. Consequently, a behavior $\mathcal{R} \in \mathfrak{L}^q$ is said to be *implementable* if there exists a controller which implements $\mathcal{R}$.

Problem 1 (Data-driven controller implementability problem)**.** Consider a plant behavior $\mathcal{P} \in \mathfrak{L}^{q+k}$ and a reference behavior $\mathcal{R} \in \mathfrak{L}^q$. Given trajectories of length $T \in \mathbb{N}$ of the plant behavior $(w, c) \in \mathcal{P}|_T$ and of the reference behavior $r \in \mathcal{R}|_T$, the *data-driven controller implementability problem* is to find, if possible, a controller $\mathcal{C} \in \mathfrak{L}^k$ which implements $\mathcal{R}$.

The data-driven implementability problem is *solvable* if $\mathcal{R}$ is implementable, in which case any controller $\mathcal{C}$ implementing $\mathcal{R}$ is a *solution* of the problem.

IV. Main results

This section contains the main results of the paper and is logically divided in two parts. First, we provide necessary and sufficient conditions for implementablity of a given reference behavior which only rely on measured data. Second, we present a data-driven strategy to obtain controllers for any given implementable reference behavior.

A. Data-driven implementability conditions

The data-driven controller implementability problem is closely related to the controller implementability problem [3], which seeks to determine all implementable reference behaviors $\mathcal{R} \in \mathfrak{L}^q$ for a given plant $\mathcal{P} \in \mathfrak{L}^{q+k}$. The problem has been first studied in a continuous-time setting in [3]. An elegant solution is provided by the following result.

Theorem 1 (Infinite-horizon implementability conditions)**.** [3, Theorem 1] Consider a plant behavior $\mathcal{P} \in \mathfrak{L}^{q+k}$ and a reference behavior $\mathcal{R} \in \mathfrak{L}^q$. Then $\mathcal{R}$ is implementable if and only if

$$\mathcal{N} \subseteq \mathcal{R} \subseteq \pi_w(\mathcal{P}). \tag{8}$$

Theorem 1 provides a powerful necessary and sufficient condition for the existence of controllers implementing a given reference behavior. However, verifying the implementability condition (8) may be challenging in practice because it requires full knowledge of both the hidden behavior and the uncontrolled plant behavior; this is especially true if only measured data of the plant and reference behaviors are available.

We now present a simple, but important extension of the controller implementability theorem, which provides necessary and sufficient condition for a reference behavior to be implementable while only requiring knowledge of the hidden behavior and the uncontrolled plant behavior over a *finite* time horizon.

Theorem 2 (Finite-horizon implementability conditions)**.** Consider a plant behavior $\mathcal{P} \in \mathfrak{L}^{q+k}$ and a reference behavior $\mathcal{R} \in \mathfrak{L}^q$. Suppose $L > \max\{\ell(\mathcal{P}), \ell(\mathcal{R}), \ell(\pi_w(\mathcal{P}))\}$. Then $\mathcal{R}$ is implementable if and only if

$$\mathcal{N}|_L \subseteq \mathcal{R}|_L \subseteq \pi_w(\mathcal{P})|_L. \tag{9}$$

Theorem 2 offers an alternative non-parametric necessary and sufficient condition for the existence of controllers that can implement a given reference behavior. Similar to Theorem 1, Theorem 2 establishes implementability conditions that do not rely on a specific representation. However, unlike Theorem 1, the subspace inclusions (9) only need information about finite-horizon behaviors, whereas subspace inclusions (8) require knowledge of the complete (infinite-dimensional) behaviors.

An important consequence of Theorem 2 is that the subspace inclusions (9) can be translated into implementability criteria which can be verified directly from data. In particular, the following result provides general necessary and sufficient conditions for the implementability of a given reference behavior using data.

Corollary 1 (Data-driven implementability conditions)**.** Consider a plant behavior $\mathcal{P} \in \mathfrak{L}^{q+k}$ and a reference behavior $\mathcal{R} \in \mathfrak{L}^q$. Suppose $L > \max\{\ell(\mathcal{P}), \ell(\mathcal{R}), \ell(\pi_w(\mathcal{P}))\}$. Let $(w, c) \in \mathcal{P}|_T$ and $r \in \mathcal{R}|_T$ be GPE of order L. Define

$$N = H_L(w)\left(I - H_L(c)^\dagger H_L(c)\right) \tag{10}$$

$$R = H_L(r) \tag{11}$$

$$P_w = H_L(w). \tag{12}$$

Then $\mathcal{N}|_L = \operatorname{im} N$, $\mathcal{R}|_L = \operatorname{im} R$, $\pi_w(\mathcal{P})|_L = \operatorname{im} P_w$. Consequently, the reference behavior $\mathcal{R}$ is implementable if and only if the system of linear equations

$$N = R\Phi, \quad R = P_w \Psi, \tag{13}$$

in the unknown matrices Φ and Ψ admits a solution.

Corollary 1 establishes necessary and sufficient conditions for testing the implementability of a given reference behavior directly from data. This, in turn, provides a necessary and sufficient condition for the solvability of the data-driven controller implementability problem.

B. Data-driven canonical controller representation

Theorem 2 and Corollary 1 provide conditions under which a reference behavior is implementable, but do not provide expressions for a controller which implements the reference behavior. We first recall an expression for a controller $\mathcal{C} \in \mathfrak{L}^k$ which implements a given implementable reference behavior $\mathcal{R} \in \mathfrak{L}^q$ and, subsequently, obtain an expression for such controller which relies only on data.

Theorem 3 (Canonical controller)**.** [5, Theorem 2.1] Consider a plant behavior $\mathcal{P} \in \mathfrak{L}^{q+k}$ and a reference behavior $\mathcal{R} \in \mathfrak{L}^q$. Assume $\mathcal{R}$ is implementable. Then $\mathcal{R}$ is implemented by the controller

$$\mathcal{C} = \pi_c(\mathcal{P} \|_w \mathcal{R}). \tag{14}$$

Theorem 3 provides a universal formula which defines the behavior of a controller which implements any implementable reference behavior. Consequently, the controller (14) is referred to as the *canonical controller*. Fig. 3 offers a pictorial illustration of the canonical controller behavior.

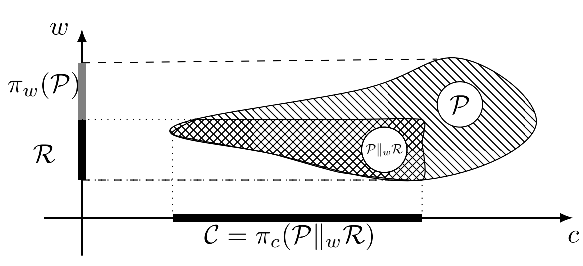

Fig. 3: Pictorial illustration of the canonical controller.

The concept of canonical controller has been implicitly defined in the seminal paper [3] and extended to general systems, e.g., in [5] and [6]. The canonical controller is appealing due to its simple construction and its representation-free formalization of the internal model principle [14].

Remark 1 (Canonical controller and well-posedness)**.** The canonical controller is such that the interconnection of $\mathcal{P}$ and $\mathcal{C}$ is well-posed, *i.e.*, $\mathcal{P}\|_c\mathcal{C} \neq \emptyset$. If $\mathcal{R}$ is implementable, then

$$\mathcal{P}\|_c\mathcal{C} \overset{(14)}{=} \mathcal{P} \cap (\mathcal{P}\|_w\mathcal{R}) \overset{(8)}{=} \mathcal{P}\|_w\mathcal{R} \neq \emptyset,$$

where the last inequality follows from the implementability assumption on $\mathcal{R}$. △

Next, we show that this concept also allows us to define the restricted behavior of a controller $\mathcal{C} \in \mathfrak{L}^k$ which implements a desired reference behavior $\mathcal{R} \in \mathfrak{L}^q$ using only measured data. For $L \in \mathbb{N}$, we define the matrix representations $\Pi_w \in \mathbb{R}^{qL \times (q+k)L}$ and $\Pi_c \in \mathbb{R}^{kL \times (q+k)L}$ of the projections π_w and π_c over the time horizon $[1, L]$ as

$$\Pi_w = \text{block-diag}\left(\begin{bmatrix} I & 0 \end{bmatrix}, \ldots, \begin{bmatrix} I & 0 \end{bmatrix}\right),$$
$$\Pi_c = \text{block-diag}\left(\begin{bmatrix} 0 & I \end{bmatrix}, \ldots, \begin{bmatrix} 0 & I \end{bmatrix}\right).$$

Corollary 2 (Data-driven canonical controller representation)**.** Consider a plant behavior $\mathcal{P} \in \mathfrak{L}^{q+k}$ and a reference behavior $\mathcal{R} \in \mathfrak{L}^q$. Assume $\mathcal{R}$ is implementable and let $\mathcal{C}$ be the canonical controller (14). Let $L > \max\{\ell(\mathcal{P}), \ell(\mathcal{R}), \ell(\pi_w(\mathcal{P}))\}$. Let $(w, c) \in \mathcal{P}|_T$ and $r \in \mathcal{R}|_T$ be GPE of order L. Define

$$P \sim H_L((w, c)), \quad R = H_L(r),$$

and

$$P_p = PP^\dagger, \quad P_r \sim \begin{bmatrix} RR^\dagger & 0 \\ 0 & I_{kL} \end{bmatrix},$$

where $\sim$ denotes similarity under a coordinates permutation. Then

$$\mathcal{C}|_L = \operatorname{im} \Pi_c P_r \left(P_r + P_p\right)^\dagger P_p. \tag{15}$$

Corollary 2 provides a data-based description of the finite-horizon behavior of the canonical controller. This formula serves a dual purpose: it can be used to identify a controller from measured data of the reference and the plant, or for direct control purposes by generating finite-length trajectories of the canonical controller. Note that longer trajectories for specific control requirements may be also generated using the lemma on weaving trajectories [15, Lemma 8.21].

Remark 2 (Persistency of excitation of the data)**.** Corollaries 1 and 2 rely on the assumption that $(w, c) \in \mathcal{P}|_T$ and $r \in \mathcal{R}|_T$ are GPE of order L. In order to check such assumption from data, upper bounds on $n(\mathcal{P})$, $m(\mathcal{P})$, $n(\mathcal{R})$, and $m(\mathcal{R})$ are needed (see the rank condition (4)). Alternatively, the rank condition (4) can be guaranteed to hold for controllable systems if a certain rank condition on the inputs hold [12]. △

Remark 3 (Alternative matrix representations)**.** Corollaries 1 and 2 can be also expressed using alternative data-driven representations of the restricted behaviors $\mathcal{N}|_L$, $\mathcal{R}|_L$, and $\pi_w(\mathcal{P})|_L$, *e.g.*, using Page matrices [16] or mosaic-Hankel matrices [17, 18]. △

Remark 4 (Connections to exact model matching)**.** The controller implementability problem is closely related to the *exact model matching* [7, 8], where the goal is to design a state feedback law for an LTI system to match a reference transfer function. The problem is well-studied in a model-based context [7–9] and it typically reduces to solving a set of linear algebraic equations [8, 9]. The recent paper [10] presents analogous findings in a data-driven context. It can be shown that the data-driven implementability condition (13) generalizes the results obtained in [10]. △

Remark 5 (Reference behaviors that are not implementable)**.** When the reference behavior $\mathcal{R}$ does not satisfy the implementability conditions (8) or, equivalently, (9) for $L > \max\{\ell(\mathcal{P}), \ell(\mathcal{R}), \ell(\pi_w(\mathcal{P}))\}$, one option is to adjust $\mathcal{R}$ [5, Remark 2.6]. This involves excluding w values without corresponding c values in $\mathcal{P}$ and including w values from $\mathcal{R}$ that match with c values in $\mathcal{P}$, creating a new implementable reference $\mathcal{R}'$. Alternatively, one may search for implementable controlled behaviors such that (9) holds, while minimizing the distance from the original reference behavior by exploiting the (Grassmannian) geometry of finite-horizon LTI behaviors [19]. △

V. Conclusion

We have studied the controller implementability problem from the lens of data-driven control, providing necessary and sufficient implementability conditions which can rely solely on raw data. Furthermore, we have addressed the problem of constructing controllers directly from data. By employing the notion of canonical controller, we have presented a formula for generating controllers which implement any plant-compatible reference behaviors in a data-driven fashion. Future research should address noisy scenarios and study an approximate version of the controller implementability problem.

Appendix

A. Proofs

1) Preliminary results. The proofs of our main results rely on several preliminary results about the interplay between coordinate projections, the cut operator, restricted LTI behaviors, and orthogonal projections onto intersections of subspaces.

Lemma 2 (Preimages under coordinate projections)**.** Let $\mathcal{B} \in \mathfrak{L}^q$. Assume (w, c) is a partition of $(\mathbb{R}^{q+k})^{\mathbb{N}}$, with $w \in \mathbb{R}^q$ and $c \in \mathbb{R}^k$. Then

$$\pi_w^{-1}(\mathcal{B}) = \mathcal{B} \times (\mathbb{R}^k)^{\mathbb{N}}.$$

Proof. By definition, we have

$$\pi_w^{-1}(\mathcal{B}) = \left\{(w, c) \in (\mathbb{R}^{q+k})^{\mathbb{N}} \, : \, w \in \mathcal{B}\right\} = \mathcal{B} \times (\mathbb{R}^k)^{\mathbb{N}}.$$

□

Lemma 3 (Coordinate projections and LTI behaviors)**.** Let $\mathcal{B} \in \mathfrak{L}^{q+k}$. Assume (w, c) is a partition of $\mathcal{B}$, with $w \in \mathbb{R}^q$ and $c \in \mathbb{R}^k$. Then

$$\pi_w(\mathcal{B})|_L = \Pi_w(\mathcal{B}|_L)$$

for all $L \in \mathbb{N}$.

Proof. ($\subseteq$). We first show $\pi_w(\mathcal{B})|_L \subseteq \Pi_w(\mathcal{B}|_L)$. Let $(\tilde{w}(1), \ldots, \tilde{w}(L)) \in \pi_w(\mathcal{B})|_L$. Then there is $w \in \pi_w(\mathcal{B})$ such that $w|_L = (\tilde{w}(1), \ldots, \tilde{w}(L))$. Furthermore, there exists c such that $(w, c) \in \mathcal{B}$ and, hence,

$$(w, c)|_L = (\tilde{w}(1), c(1), \ldots, \tilde{w}(L), c(L)) \in \mathcal{B}|_L.$$

This implies $(\tilde{w}(1), \ldots, \tilde{w}(L)) \in \Pi_w(\mathcal{B}|_L)$, and, hence, $\pi_w(\mathcal{B})|_L \subseteq \Pi_w(\mathcal{B}|_L)$.
($\supseteq$). Next, we show $\pi_w(\mathcal{B})|_L \supseteq \Pi_w(\mathcal{B}|_L)$. Let

$$(\tilde{w}(1), \ldots, \tilde{w}(L)) \in \Pi_w(\mathcal{B}|_L).$$

Then there exists $(\tilde{c}(1), \ldots, \tilde{c}(L))$ such that

$$(\tilde{w}(1), \tilde{c}(1), \ldots, \tilde{w}(L), \tilde{c}(L)) \in \mathcal{B}|_L.$$

Thus, there exists $(w, c) \in \mathcal{B}$ such that

$$(w, c)|_L = (\tilde{w}(1), \tilde{c}(1), \ldots, \tilde{w}(L), \tilde{c}(L)).$$

Then $w \in \pi_w(\mathcal{B})$ and, hence, $w|_L \in \pi_w(\mathcal{B})|_L$. This implies

$$(\tilde{w}(1), \ldots, \tilde{w}(L)) \in \pi_w(\mathcal{B})|_L,$$

which proves $\Pi_w(\mathcal{B}|_L) \subseteq \pi_w(\mathcal{B})|_L$ and, hence, the claim. □

Lemma 4 (Intersection of restricted LTI behaviors)**.** [20, Proposition 16] Let $\mathcal{B} \in \mathfrak{L}^q$ and $\bar{\mathcal{B}} \in \mathfrak{L}^q$. Then

$$(\mathcal{B} \cap \bar{\mathcal{B}})|_L \subseteq \mathcal{B}|_L \cap \bar{\mathcal{B}}|_L.$$

for all $L \in \mathbb{N}$. Furthermore, if $L > \max\{\ell(\mathcal{B}), \ell(\bar{\mathcal{B}})\}$, then

$$(\mathcal{B} \cap \bar{\mathcal{B}})|_L = \mathcal{B}|_L \cap \bar{\mathcal{B}}|_L.$$

Lemma 5 (Cartesian product of restricted LTI behaviors)**.** [20, Proposition 19] Let $\mathcal{B} \in \mathfrak{L}^q$ and $\bar{\mathcal{B}} \in \mathfrak{L}^k$. Then for all $L \in \mathbb{N}$, $(\mathcal{B} \times \bar{\mathcal{B}})|_L = \mathcal{B}|_L \times \bar{\mathcal{B}}|_L$.

Lemma 6 (Inclusion between restricted LTI behaviors)**.** Let $\mathcal{B} \in \mathfrak{L}^q$ and $\bar{\mathcal{B}} \in \mathfrak{L}^q$. Then $\bar{\mathcal{B}}|_L \subseteq \mathcal{B}|_L$ implies $\bar{\mathcal{B}} \subseteq \mathcal{B}$ for $L > \max\{\ell(\bar{\mathcal{B}}), \ell(\mathcal{B})\}$.

Proof. We have that $\bar{\mathcal{B}}|_L \subseteq \mathcal{B}|_L$ if and only if $\bar{\mathcal{B}}|_L \cap \mathcal{B}|_L = \bar{\mathcal{B}}|_L$. By Lemma 4, we have

$$\bar{\mathcal{B}}|_L \cap \mathcal{B}|_L = (\mathcal{B} \cap \bar{\mathcal{B}})|_L.$$

Then $\bar{\mathcal{B}}|_L = (\mathcal{B} \cap \bar{\mathcal{B}})|_L$. By [13, Corollary 14], $\bar{\mathcal{B}} = \mathcal{B} \cap \bar{\mathcal{B}}$ which implies that $\bar{\mathcal{B}} \subseteq \mathcal{B}$, proving the claim. □

Lemma 7 (Projectors on intersection of subspaces)**.** [21, p.2] Let $\mathcal{V}$ and $\mathcal{W}$ be subspaces of $\mathbb{R}^n$ and let $P_{\mathcal{V}}$ and $P_{\mathcal{W}}$ be the orthogonal projectors on $\mathcal{V}$ and $\mathcal{W}$, respectively. Then the orthogonal projector on the intersection of $\mathcal{V}$ and $\mathcal{W}$ is

$$P_{\mathcal{V} \cap \mathcal{W}} = 2P_{\mathcal{V}}(P_{\mathcal{V}} + P_{\mathcal{W}})^{\dagger} P_{\mathcal{W}}.$$

2) Proof of Theorem 2. We prove the claim by showing that (8) is equivalent to (9) for $L > \max\{\ell(\mathcal{P}), \ell(\mathcal{R}), \ell(\pi_w(\mathcal{P}))\}$.
(8)⇒(9): This holds by definition of the cut operator.
(9)⇒(8): We first show that $\mathcal{N}|_L \subseteq \mathcal{R}|_L$ implies $\mathcal{N} \subseteq \mathcal{R}$. First, note that $L > \max\{\ell(\mathcal{N}), \ell(\mathcal{R})\}$. Indeed, by assumption, $L > \ell(\mathcal{R})$. Furthermore, $L > \ell(\mathcal{N})$. Indeed, let

$$\mathcal{P} = \ker \begin{bmatrix} R_w(\sigma) & R_c(\sigma) \end{bmatrix}$$

be a minimal kernel representation for $\mathcal{P}$. Then $\mathcal{N} = \ker R_w(\sigma)$. Thus, $\ell(\mathcal{P}) \geq \ell(\mathcal{N})$ and, hence, $L > \max\{\ell(\mathcal{N}), \ell(\mathcal{R})\}$. By Lemma 6, we conclude that $\mathcal{N} \subseteq \mathcal{R}$. It can be shown that $\mathcal{R}|_L \subseteq \pi_w(\mathcal{P})|_L$ implies $\mathcal{R} \subseteq \pi_w(\mathcal{P})$ using similar arguments. □

3) Proof of Corollary 1. Let $\bar{w} \in \pi_w(\mathcal{P})|_L$. Since $(w, c) \in \mathcal{P}|_T$ and $r \in \mathcal{R}|_T$ are GPE of order L, there exists g and $\bar{c}$ such that

$$\begin{bmatrix} H_L(w) \\ H_L(c) \end{bmatrix} g = \begin{bmatrix} \bar{w} \\ \bar{c} \end{bmatrix}.$$

Thus $\bar{w} \in \operatorname{im} H_L(w)$. Now let $\bar{w} \in \operatorname{im} H_L(w)$. Then there exists g such that $H_L(w)g = \bar{w}$. Thus there exists $\bar{c} \in \operatorname{im} H_L(c)$ such that $(\bar{w}, \bar{c}) \in \mathcal{P}|_L$. Hence, $\bar{w} \in \pi_w(\mathcal{P})|_L$, so that $\pi_w(\mathcal{P})|_L = \operatorname{im} H_L(w)$.

Now let $\bar{w} \in \mathcal{N}|_L$. Then there exists g such that

$$\begin{bmatrix} H_L(w) \\ H_L(c) \end{bmatrix} g = \begin{bmatrix} \bar{w} \\ 0 \end{bmatrix}.$$

Thus, $g \in \ker H_L(c) = \operatorname{im}(I - H_L(c)^\dagger H_L(c))$. This, in turn, implies

$$\bar{w} \in \operatorname{im} H_L(w)(I - H_L(c)^\dagger H_L(c)).$$

Now let $\bar{w} \in \operatorname{im} H_L(w)(I - H_L(c)^\dagger H_L(c))$. Then there exists g such that $\bar{w} = H_L(w)(I - H_L(c)^\dagger H_L(c))g$. Thus,

$$\begin{bmatrix} \bar{w} \\ 0 \end{bmatrix} = \begin{bmatrix} H_L(w)(I - H_L(c)^\dagger H_L(c)) \\ 0 \end{bmatrix} g = \begin{bmatrix} H_L(w) \\ H_L(c) \end{bmatrix} \bar{g},$$

with $\bar{g} = (I - H_L(c)^\dagger H_L(c))g$. Hence, $(\bar{w}, 0) \in \mathcal{P}|_L$, so $\bar{w} \in \mathcal{N}|_L$, showing that

$$\mathcal{N}|_L = \operatorname{im}(I - H_L(c)^\dagger H_L(c)).$$

The second claim now follows directly from Theorem 2 and the fact that the subspace inclusions

$$\operatorname{im} N \subseteq \operatorname{im} R \subseteq \operatorname{im} P_w$$

hold if and only if the system of linear equations (13) admits a solution. □

4) Proof of Corollary 2. By assumption, $\mathcal{R}$ is implementable and, hence, the canonical controller (14) is well-defined. By applying the cut operator to the definition of the canonical controller (14), we obtain

$$\mathcal{C}|_L = \pi_c(\pi_w^{-1}(\mathcal{R}) \cap \mathcal{P})|_L.$$

By Lemma 3, we obtain

$$\mathcal{C}|_L = \Pi_c \left((\pi_w^{-1}(\mathcal{R}) \cap \mathcal{P})|_L \right).$$

Using Lemma 4 and $L > \max\{\ell(\mathcal{P}), \ell(\mathcal{R}), \ell(\pi_w(\mathcal{P}))\}$, we obtain

$$\mathcal{C}|_L = \Pi_c \left(\pi_w^{-1}(\mathcal{R})|_L \cap \mathcal{P}|_L \right).$$

By Lemma 2 and Lemma 5, we can write the above as

$$\mathcal{C}|_L = \Pi_c \left((\mathcal{R}|_L \times \mathbb{R}^{kL}) \cap \mathcal{P}|_L \right).$$

By Lemma 7, the fact that $(w, w_c) \in \mathcal{P}|_T$ and $w_r \in \mathcal{R}|_T$ are GPE of order L, and the definition of the projectors P_p and P_r, we obtain

$$\mathcal{C}|_L = \Pi_c \operatorname{im} P_r \left(P_r + P_p\right)^\dagger P_p.$$

Finally, since Π_c is surjective,

$$\mathcal{C}|_L = \operatorname{im} \Pi_c P_r \left(P_r + P_p\right)^\dagger P_p.$$

This proves the result. □

References

[1] J. C. Willems, "On interconnections, control, and feedback," *IEEE Transactions on Automatic control*, vol. 42, no. 3, pp. 326–339, 1997.

[2] I. Markovsky and F. Dörfler, "Behavioral systems theory in data-driven analysis, signal processing, and control," *Ann. Rev. Control*, vol. 52, pp. 42–64, 2021.

[3] J. C. Willems and H. L. Trentelman, "Synthesis of dissipative systems using quadratic differential forms: Part I," *IEEE Transactions on Automatic Control*, vol. 47, no. 1, pp. 53–69, 2002.

[4] J. C. Willems, "From time series to linear system—Part I. Finite dimensional linear time invariant systems," *Automatica*, vol. 22, no. 5, pp. 561–580, 1986.

[5] A. J. Van der Schaft, "Achievable behavior of general systems," *Syst. Control Lett.*, vol. 49, no. 2, pp. 141–149, 2003.

[6] A. A. Julius, J. C. Willems, M. N. Belur, and H. L. Trentelman, "The canonical controllers and regular interconnection," *Syst. Control Lett.*, vol. 54, no. 8, pp. 787–797, 2005.

[7] W. A. Wolovich, "The use of state feedback for exact model matching," *SIAM J. Control Optim.*, vol. 10, no. 3, pp. 512–523, 1972.

[8] S. Wang and C. Desoer, "The exact model matching of linear multivariable systems," *IEEE Trans. Autom. Control*, vol. 17, no. 3, pp. 347–349, 1972.

[9] S. Wang and E. Davison, "Solution of the exact model matching problem," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 574–574, 1972.

[10] V. Breschi, C. De Persis, S. Formentin, and P. Tesi, "Direct data-driven model-reference control with lyapunov stability guarantees," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 1456–1461.

[11] J. C. Willems, "Paradigms and puzzles in the theory of dynamical systems," *IEEE Trans. Autom. Control*, vol. 36, no. 3, pp. 259–294, 1991.

[12] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. M. De Moor, "A note on persistency of excitation," *Syst. Control Lett.*, vol. 54, no. 4, pp. 325–329, 2005.

[13] I. Markovsky and F. Dörfler, "Identifiability in the behavioral setting," *Vrije Universiteit Brussel, Tech. Rep*, 2020. [Online]. Available: http://homepages.vub.ac.be/~imarkovs/publications/identifiability.pdf.

[14] B. A. Francis and W. M. Wonham, "The internal model principle of control theory," *Automatica*, vol. 12, no. 5, pp. 457–465, 1976.

[15] I. Markovsky, J. C. Willems, S. Van Huffel, and B. De Moor, *Exact and approximate modeling of linear systems: A behavioral approach*. Philadelphia, PA, USA: SIAM, 2006.

[16] J. Coulson, J. Lygeros, and F. Dörfler, "Distributionally robust chance constrained data-enabled predictive control," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3289–3304, 2021.

[17] H. J. van Waarde, C. De Persis, M. K. Camlibel, and P. Tesi, "Willems' fundamental lemma for state-space systems and its extension to multiple datasets," *IEEE Control Systems Letters*, vol. 4, no. 3, pp. 602–607, 2020.

[18] I. Markovsky and F. Dörfler, "Identifiability in the behavioral setting," *IEEE Trans. Autom. Control*, 2022.

[19] A. Padoan, J. Coulson, H. J. van Waarde, J. Lygeros, and F. Dörfler, "Behavioral uncertainty quantification for data-driven control," in *Proc. 61st Conf. Decision Control*, Cancun, Mexico, 2022, pp. 4726–4731.

[20] Y. Yan, J. Bao, and B. Huang, "Distributed data-driven predictive control via dissipative behavior synthesis," *arXiv preprint arXiv:2303.00251*, 2023.

[21] A. Ben-Israel, "Projectors on intersection of subspaces," *Contemporary Mathematics*, vol. 636, pp. 41–50, 2015.