Numerical methods for linear time-invariant systems analysis and design in the behavioral setting

Ivan Markovsky[1,2], Mohammad Alsalti[3], and Matthias Müller[3]

1. Catalan Institution for Research and Advanced Studies
Pg. Lluis Companys 23, 08010 Barcelona, Spain

2. Centre Internacional de Mètodes Numèrics en Enginyeria
Edifici C1, Campus Nord UPC, 08034 Barcelona, Spain
E-mail: imarkovsky@cimne.upc.edu

3. Leibniz University Hannover
Institute of Automatic Control
30167 Hannover, Germany
E-mail: {alsalti, mueller}@irt.uni-hannover.de

Abstract

Analysis and design of dynamical systems is transitioning from a model-based to a data-driven paradigm. Theoretical foundation for the data-driven paradigm is provided by the behavioral approach, which views systems as sets of trajectories, however, currently it lacks robust and efficient computational methods. A key subproblem is computing a basis for the finite-horizon behavior. This can be done using a parametric representation or trajectories of the system. The obtained basis for the finite-horizon behavior is a non-parametric representation of the system. Contrary to classical non-parametric representations, *e.g.*, impulse response and frequency response, it is finite and exact, captures nonzero initial conditions, and is applicable to uncontrollable systems. It also leads to new computational methods for systems analysis, signal processing, and control. Examples presented in the paper are finding the system's complexity, finding an input/output partitioning of the variables, and checking controllability.

Keywords: Behavioral approach, data-driven methods, numerical algorithms, structured matrix computations.

1 Introduction

In systems and control, the notion of a dynamical system is often conflated with the equations used to represent the system, or with the parameters of these equations. Since the notion of a dynamical system is rarely defined formally, it seems pedantic to insist on the distinction. Expressions such as "a state-space system" and "the system (A,B,C,D)" are misnomers but are commonly accepted and do not cause confusion. Nevertheless, the system–representation distinction is important and has far reaching consequences. In particular, it allows us to decouple problem statements from solution methods and is thus essential for the development and comparison of solution methods.

We use the "behavioral" definition of a dynamical system due to its simplicity and generality. The idea of viewing a system as a set of trajectories—the *behavior*—was put forward by Jan C. Willems in [23]. It started what came to be known as the *behavioral approach* to systems theory [18, 24]. Initially the behavioral approach was a niche area of theoretical research, motivated by deficiencies of the classical input/output setting. This changed recently with the development of *direct data-driven methods*, which have impact on real-life applications [14].

The distinction among system, representation, and parameters is essential in direct data-driven analysis and design. Starting from data, it is natural to formulate problems in terms of the given data. The behavioral approach provides the necessary language for doing this because systems are subsets of the space where the data lives. This allows us to introduce in a problem an unknown data-generating system without a reference to a representation and parameters.

Direct data-driven methods are sometimes refered to as "model-free". This term can, however, be misleading because it depends on the meaning of "model." For some, "model" means "system," while for others, it means

"parametric representation". In data-driven problems, existence of an underlying data-generating system is assumed, however, direct data-driven methods for solving the problems do not identify a parametric representation of the data-generating system. Thus, these methods are "model-free" in the sense of "free of parametric representations."

A behavioral / representation-free problem formulation decouples what we aim at—the problem—from how we can possibly achieve it—solution methods, *i.e.*, it abstracts from the details of how the problem is solved and focuses on what is given and what is desired. A representation of the system is needed for analysis and computations when solving problems. Once a representation is chosen, the analysis is done using the equations of the representation and computations are done using the parameters of the representation. Thus, a problem is solved by: 1) choosing a representation and 2) using analytical and computational methods dependent on the chosen representation. The key point is that the choice of the representation is a part of the solution method, not the problem formulation.

The insight that the same problem can be solved by different representations and techniques is not new. A well-known example is the dichotomy of time-domain vs frequency-domain methods—a problem can be solved in either the time-domain or the frequency-domain. The methods and their properties may be different, however, the computed results are equivalent in the sense that they solve the same high-level problem. The classical approach to analysis and design is missing the "high-level problem" because it invariably starts with a given model representation; for linear time-invariant systems, typically a state-space or a transfer function.

In addition to contributing a high-level perspective, the behavioral approach brings also new types of representations, new solution techniques, and more recently, new computational methods. In this paper, we introduce and use for development of computational methods a non-parametric representation of the finite-horizon behavior, which is inspired by a result that become known as the *fundamental lemma* [25]. We focus on deterministic linear time-invariant systems, in which case the representation is given by a set of basis vectors. It's key features are:

- *generality*—applies to any finite-dimensional linear time-invariant system,
- *simplicity*—the required mathematics for its derivation and usage is linear algebra only, and
- *applicability*—as shown in the paper, it leads to novel direct data-driven methods for analysis and design.

The generality is due to its behavioral nature. In particular, it does not require a partitioning of the variables into inputs and outputs, and represents multivariable, uncontrollable, and autonomous systems without ad hoc adaptations. The simplicity and applicability are due to the fact that it is a basis of a linear subspace. These advantages, however, come at a price: with the increase of the time horizon, the required memory and computations increase. This issue can be addressed by methods for structured matrices, such as the ones based on the displacement rank theory [7, 8].

The following sections elaborate the ideas outlined above. Section 2 explains the distinction between a system and a representation and Section 3 explains the importance of representation-free problem formulations, thus separating problems from solution methods. Section 4 puts the theory of Sections 2 and 3 into practice. The examples presented are the problems of finding the system's complexity, finding an input/output partitioning of the variables, and checking controllability. The methods developed for these problems are new, use the nonparametric representation of the finite-horizon behavior, and are suitable for direct data-driven analysis. They are implemented in the Behavioral Toolbox [11] and are illustrated by numerical examples, available from `https://imarkovs.github.io/bt`

2 Dynamical systems and their representations

At the core of the behavioral approach to systems theory is the distinction between a system and a representation. First, we introduce sets of signals and systems and the basic operations of shifting and restricting them to an interval. Then, we define the properties of linearity, time-invariance, and controllability. The familiar definitions of these properties are on the representation level. Defining the properties on the systems' trajectories level makes them more general and intuitive. For example, the classical definition of linearity assumes input/output partitioning of the variables and zero initial conditions. The behavioral definition does not make such assumptions and has an appealing geometrical interpretation—a linear system is a linear subspace. Another case in point is controllability. The classical definition applies to state-space representations and involves non-intuitive conditions on the model parameters. The behavioral definition, in contrast, applies to general infinite-dimensional systems and states directly the meaning of controllability as the property that ensures well-posedness (existence of a solution) of the state transfer control problem.

Apart from conceptual and pedagogical advantages, the behavioral approach leads to new representations and computational methods. The class of systems considered is discrete-time, bounded complexity, linear, time-invariant. This class of systems is characterized by a set of integers and admits compact parametric representations by recursive formulas. We review the familiar input/state/output and kernel representations as examples of parametric representations, as well as a nonparametric representation of the finite-horizon behavior. The latter can be obtained from the image of the Hankel matrix constructed by a trajectory of the system. The kernel representation involves a polynomial operator and opens the path for solving systems and control problems by polynomial algebra. Using the so called multiplication matrix, we represent the action of a polynomial operator by a matrix–vector multiplication. This allows us to use classical linear algebra for solving problems in behavioral systems theory. Both the Hankel matrix and the multiplication matrix have special structure that can be exploited in development of efficeint computational methods.

2.1 Dynamical systems as sets of trajectories

We consider discrete-time signals and systems. The time index takes on its values in the set of integers $\mathbb{Z}$ (two-side infinite signal), the set of natural numbers $\mathbb{N}$ (one side-infinite signal), or a finite interval $(1,\ldots,T)$. The corresponding sets of discrete-time, real, q-variate signals are denoted by $(\mathbb{R}^q)^{\mathbb{Z}}$, $(\mathbb{R}^q)^{\mathbb{N}}$, and $(\mathbb{R}^q)^T$, respectively.

Two basic operations on signals are shifting and restricting. The *unit-shift* $\sigma : (\mathbb{R}^q)^{\mathbb{Z}} \mapsto (\mathbb{R}^q)^{\mathbb{Z}}$ is defined as

$$(\sigma w)(t) := w(t+1), \quad \text{for all } t \in \mathbb{Z}.$$

The *restriction* $w|_L$ of a signal $w \in (\mathbb{R}^q)^{\mathbb{Z}}$ to the interval $(1,\ldots,L)$ is defined as

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

A discrete-time, real, q-variate dynamical system $\mathscr{B}$ is a subset of the set of signals $(\mathbb{R}^q)^{\mathbb{Z}}$, *i.e.*, $\mathscr{B} \subset (\mathbb{R}^q)^{\mathbb{Z}}$. By polymorphism, the restriction and the unit-shift operators can be applied on systems. In this case, the operators act on all signals in the set and produce a new set that consists of the resulting signals:

$$\mathscr{B}|_L := \{\, w|_L \mid w \in \mathscr{B} \,\} \quad \text{and} \quad \sigma\mathscr{B} := \{\, \sigma w \mid w \in \mathscr{B} \,\}.$$

2.2 Systems' properties defined in terms of their trajectories

A system $\mathscr{B} \subset (\mathbb{R}^q)^{\mathbb{Z}}$ is *linear* if $\mathscr{B}$ is a linear subspace of $(\mathbb{R}^q)^{\mathbb{Z}}$ and *time-invariant* if $\sigma\mathscr{B} = \mathscr{B}$. The class of linear time-invariant systems with q variables is denoted by $\mathscr{L}^q$. Unless otherwise stated, in what follows, we consider linear time-invariant systems.

As the properties of linearity and time-invariance, *controllability* is also defined in terms of trajectories, *i.e.*, the system's behavior. Informally, controllability is the property that makes possible to patch the "past" of any trajectory with the "future" of any other trajectory. Formally, a system $\mathscr{B} \subset (\mathbb{R}^q)^{\mathbb{Z}}$ is *controllable* if for any pair of trajectories $w_{\mathrm{p}}, w_{\mathrm{f}} \in \mathscr{B}$, there is a trajectory $w_{\mathrm{c}} \in \mathscr{B}$ and an integer t_{c}, such that $w_{\mathrm{c}}(t) = w_{\mathrm{p}}(t)$ for $t \in (-\infty, 0]$ and $w_{\mathrm{c}}(t) = w_{\mathrm{f}}(t)$ for $t \in [t_{\mathrm{c}}, \infty)$. Thinking of w_{p} as a specification of an initial condition at time $t = 0$ and w_{f} as a specification of a final condition at time $t = t_{\mathrm{c}}$, the "patching of trajectories" problem becomes the classical state transfer problem stated in a representation-free way.

Although a system is defined abstractly as a set of trajectories, its variables can be partitioned into inputs (independent variables) and outputs (dependent variables). The defining property of an input/output partitioning is that the future outputs of the system are uniquely determined by the future inputs and the past inputs and outputs. In addition to making a link with classical systems theory, an input/output partitioning of the variables is needed for defining and solving the simulation problem: given a system, initial conditions, and input, find the corresponding output. In general, an input/output partitioning of the variables is not unique, so that in the behavioral setting one may have different options for simulating a system depending on which variables one chooses as inputs. The number of inputs and therefore the number of outputs in all input/output partitionings, however, is the same so that these numbers, denoted by $\boldsymbol{m}(\mathscr{B})$ and $\boldsymbol{p}(\mathscr{B})$ respectively, are properties of the system.

2.3 Structure of bounded complexity linear time-invariant systems: the integer invariants

In addition to the number of inputs and the number of outputs, linear time-invariant systems have other characteristics, called *integer invariants*. Two of them are particularly important: the lag $\boldsymbol{\ell}(\mathscr{B})$ and the order $\boldsymbol{n}(\mathscr{B})$ of the system. An intuitive interpretation of the lag is that it is the minimal degree of a difference equation representation of the system. Similarly, the order is the minimal state dimension of a state-space representation of the system. We define the *complexity* of a linear time-invariant system as the triple $\boldsymbol{c}(\mathscr{B}) := \big(\boldsymbol{m}(\mathscr{B}), \boldsymbol{\ell}(\mathscr{B}), \boldsymbol{n}(\mathscr{B})\big)$ and the class of bounded complexity linear time-invariant systems with q variables $\mathscr{L}^q_{(m,\ell,n)}$ by

$$\mathscr{L}^q_{(m,\ell,n)} := \{\, \mathscr{B} \in \mathscr{L}^q \mid \boldsymbol{m}(\mathscr{B}) \leq m,\ \boldsymbol{\ell}(\mathscr{B}) \leq \ell,\ \boldsymbol{n}(\mathscr{B}) \leq n \,\}.$$

A bounded complexity linear time-invariant system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ imposes restrictions on its set of trajectories. More specifically, for $T \geq \ell$, $\mathscr{B}|_T \subset (\mathbb{R}^q)^T$, so that $w \in \mathscr{B}|_T$ implies that w is constrained. Assuming a bounded complexity linear time-invariant data-generating system is a prior knowledge that makes missing data estimation, noise filtering, and control problems well-posed, *i.e.*, possible to solve. Another way of viewing the restriction that a bounded complexity linear time-invariant system imposes is that $\mathscr{B}|_T$ and any $w \in \mathscr{B}|_T$ are *structured*.

One way to expose the hidden structure of $\mathscr{B}|_T$ is to analyze the function

$$d(t) := \dim \mathscr{B}|_t, \quad \text{for } t = 1, 2, \ldots$$

Clearly d is positive, integer-valued, and monotonically increasing. For a bounded complexity linear time-invariant system, d has the following additional structure:

$$d(t+1) - d(t) = \begin{cases} q & \text{if } 1 < t \leq \ell_1 \\ q-1 & \text{if } \ell_1 < t \leq \ell_2 \\ \cdots & \cdots \\ q-p+1 & \text{if } \ell_{p-1} < t \leq \ell_p \\ q-p & \text{if } t > \ell_p, \end{cases} \tag{ℓ_i}$$

where $p = \boldsymbol{p}(\mathscr{B})$ and $\ell_1 \leq \cdots \leq \ell_p$ are the so called *structure indices* of $\mathscr{B}$ [21, Section 7]. The structure indices $(\ell_1, \ldots, \ell_p)$ fully describe the structure of a bounded complexity linear time-invariant system. In particular,

$$\boldsymbol{m}(\mathscr{B}) = q - p, \quad \boldsymbol{\ell}(\mathscr{B}) = \ell_p, \quad \text{and} \quad \boldsymbol{n}(\mathscr{B}) = \ell_1 + \cdots + \ell_p.$$

Another way to expose the structure of $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ is to choose a horizon $T \geq \ell$ and a basis $\{b^1, \ldots, b^r\}$ for $\mathscr{B}|_T$, such that the matrix $B_T := \begin{bmatrix} b^1 & \cdots & b^r \end{bmatrix}$ of the basis vectors is Hankel structured. Finding such a "structured basis" is not possible for all subspaces of $(\mathbb{R}^q)^T$ with dimension r, however, it is possible for all subspaces that represent horizon-T behaviors of a linear time-invariant systems with complexity bounded by (m, ℓ, n).

Finally, the structure of $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ can be (partially) exposed by deriving a parametric representation for $\mathscr{B}$. For example, existence of a state-space representation with a state dimension $\bar{n}$ or a kernel representation with a lag $\bar{\ell}$, implies that $\boldsymbol{n}(\mathscr{B}) \leq \bar{n}$ and $\boldsymbol{\ell}(\mathscr{B}) \leq \bar{\ell}$. This is the topic of Section 2.5, however, first we introduce the notion of a representation and make a distinction between parametric and non-parametric ones, which plays an important role.

2.4 Representations: equations that specify the behavior

In the behavioral setting the three characters "$w \in \mathscr{B}$" mean "the signal w is a trajectory of the system $\mathscr{B}$"—nothing more and nothing less. In the classical setting, saying "$w \in \mathscr{B}$" requires to 1) choose a representation of $\mathscr{B}$, *i.e.*, a specification of $\mathscr{B}$ by equations that define which signals are in $\mathscr{B}$, and 2) write down the equations with the given signal w substituted in them. Apart from taking more space on paper, time to write, and time to read, it says more than required: it gives a way by which the condition can be verified in practice. This is useful for development of computational methods, however, need not be repeated every time we say "$w \in \mathscr{B}$" (which is surprisingly often). The classical setting mixes "ends and means." The behavioral approach separates them. Sections 2.1–2.3 focused on the ends. With this section we start to address the means. The first step is choosing a representation of the system.

In order to fix the ideas and terminology behind system representations, consider the familiar specification of a linear time-invariant single-input single-output system by convolution:

$$y(t) = \sum_{\tau=0}^{\infty} h(\tau) \star u(t-\tau), \quad \text{for } t = 1,2,\ldots, \tag{CONV}$$

where $h \in \mathbb{R}^{\mathbb{Z}_+}$ is the impulse response of the system. In the classical setting (CONV) is "the system". In the behavioral setting, (CONV) is a *representation* of a system

$$\mathscr{B} := \{ w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \in (\mathbb{R}^2)^{\mathbb{N}} \mid \text{(CONV) holds} \},$$

with *parameter* $h \in \mathbb{R}^{\mathbb{Z}_+}$. Due to the linearity of (CONV), $\mathscr{B} \in \mathscr{L}^2$ and, by construction, $\boldsymbol{m}(\mathscr{B}) = 1$. However, $\mathscr{B}$ may not be bounded complexity because $\boldsymbol{n}(\mathscr{B})$ may not be finite. The complexity of $\mathscr{B}$ depends on h.

More generally, representations are given by formulas that involve parameters. When the parameter vector is infinite-dimensional, the representation is called *non-parametric*. The representation (CONV) with $h \in \mathbb{R}^{\mathbb{Z}_+}$ is non-parametric. However, if we restrict h to be finite support, *i.e.*, $h(t) = 0$, for $t > T$, the resulting (finite impulse response) representation has a finite parameter vector $\big(h(0), h(1), \ldots, h(T)\big)$. Another familiar example is representation by a transfer function: $y(z) = H(z)u(z)$. In general, the parameter H is a complex valued function of a complex argument. This makes the transfer function representation non-parametric. However, if we restrict H to be rational, *i.e.*, $H(z) = q(z)/p(z)$, where p and q are polynomials, the parameter vector becomes finite. When the parameter vector of an infinite-horizon representation is finite-dimensional, the representation is called *parametric*.

The convolution and transfer function specify infinite-horizon behaviors. In Sections 2.7 and 2.8, we consider representations of finite-horizon behaviors. For a linear time-invariant system $\mathscr{B} \in \mathscr{L}^q$, the finite-horizon behavior $\mathscr{B}|_T$ is a linear subspace of $(\mathbb{R}^q)^T$, so that it can always be represented by a finite parameter vector. The defining feature of a non-parametric representation in the finite horizon case is that the dimension of the parameter vector grows with the increase of the horizon T. Thus, for $T \to \infty$, it can not be finite.

2.5 Parametric representations of bounded complexity linear time-invariant systems

Bounded complexity linear time-invariant systems admit many representations—state-space, matrix fraction description, transfer function, impulse response, kernel, image, *etc*. In this section, we review the state-space and kernel representations as examples of parametric representations. In Section 2.7, we present a nonparametric representation.

A bounded complexity linear time-invariant system $\mathscr{B}$ admits an *input/state/output representation*

$$\mathscr{B} = \mathscr{B}_{\text{i/s/o}}(A,B,C,D,\Pi) := \{ \Pi^\top \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid \text{there is } x \in (\mathbb{R}^n)^{\mathbb{N}}, \text{ such that } \sigma x = Ax + Bu,\ y = Cx + Du \}, \tag{I/S/O}$$

with parameters a permutation matrix $\Pi \in \mathbb{R}^{q \times q}$ and a matrix $\left[\begin{smallmatrix} A & B \\ C & D \end{smallmatrix}\right] \in \mathbb{R}^{(n+p)\times(n+m)}$. An alternative parametric representation, called the *kernel representation* is [22, Theorem III.1]

$$\mathscr{B} = \ker R(\sigma) := \{ w \mid R(\sigma)w = 0 \}, \tag{KER}$$

with parameter the polynomial matrix

$$R(z) = R_0 + R_1 z + \cdots + R_\ell z^\ell = \begin{bmatrix} R^1(z) \\ \vdots \\ R^g(z) \end{bmatrix} = \begin{bmatrix} R^1_0 + R^1_1 z + \cdots + R^1_{\ell_1} z^{\ell_1} \\ \vdots \\ R^g_0 + R^g_1 z + \cdots + R^g_{\ell_g} z^{\ell_g} \end{bmatrix} \in \mathbb{R}^{g \times q}[z]. \tag{R}$$

Without loss of generality, we can assume that R is such that $\ell_1 \le \ell_2 \le \cdots \le \ell_g$. The vector of row degrees $(\ell_1, \ldots, \ell_g)$ is called the *lag structure* of the kernel representation.

The representation (I/S/O) is called *minimal* if the *state dimension* $n := \dim x$ is as small as possible over all input/state/output representations of $\mathscr{B}$. Note that this notion of minimality is different from the classical one—(A,B) controllable and (A,C) observable. When $\mathscr{B}$ is uncontrollable, its minimal (in the behavioral sense) input/state/output representations are uncontrollable (in the classical sense). For example, an autonomous system admits a minimal state-space representation, which is uncontrollable because $B = 0$.

Given a bounded complexity linear time-invariant system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$, its kernel representation (KER) is not unique. One source of non-uniqueness is redundant rows of R, *i.e.*, g not being minimal. A kernel representation with the smallest row dimension over all kernel representations of the system is called *minimal*. A kernel representation is minimal if and only if R has full row-rank over the ring of polynomials [22, Proposition III.3 i)]. The smallest row-dimension g over all kernel representations of $\mathscr{B}$ equals the number of outputs $\boldsymbol{p}(\mathscr{B})$ [22, Proposition X.3].

In a minimal kernel representation the redundant rows of R are eliminated, however, R is still not unique. This is due to multiplication of R by a unimodular matrix U, which amounts to taking linear combinations over the ring of polynomials of the rows of R. Such operations may change the lag structure of the representation. A minimal kernel representation that minimizes the total lag $\ell_1 + \cdots + \ell_p$ over all minimal kernel representations of $\mathscr{B}$ is called *shortest-lag* [21, Section 7]. All shortest-lag kernel representations have the same lag structure $(\ell_1, \ldots, \ell_p)$, which reveals the structure indices of $\mathscr{B}$. In particular, the total lag of R in a shortest-lag kernel representation is equal to the order of the system. A kernel representation is shortest-lag if and only if R is row proper [21, Theorem 6 (i)].

2.6 Expressing polynomial operations via structured matrices

As shown in the previous section, analysis of linear time-invariant systems based on the kernel representation requires tools from polynomial algebra—finding the rank of a polynomial matrix, bringing a polynomial matrix to row proper form by a unimodular transformation, *etc*. This section introduces the polynomial multiplication matrix, which allows us to convert polynomial operations to linear algebraic operations. The polynomial multiplication matrix is constructed from the coefficients of a polynomial and is a structured matrix.

With some abuse of notation, we consider the matrix sequence $(R_0, R_1, \ldots, R_\ell) \in (\mathbb{R}^{g\times q})^{\ell+1}$, the matrix polynomial $R_0 + R_1 z + \cdots + R_\ell z^\ell \in \mathbb{R}^{g\times q}[z]$, and the matrix $\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \in \mathbb{R}^{g\times q(\ell+1)}$ interchangeably, denoting them by R.

Given a row-vector polynomial $r \in \mathbb{R}^{1\times q}[z]$ with degree ℓ and $T \geq \ell$, the multiplication matrix $\mathscr{M}_T(r)$ of r with T block-columns is defined as

$$\mathscr{M}_T(r) := \begin{bmatrix} r_0 & r_1 & \cdots & r_\ell & & \\ & r_0 & r_1 & \cdots & r_\ell & \\ & & \ddots & \ddots & & \ddots \\ & & & r_0 & r_1 & \cdots & r_\ell \end{bmatrix} \in \mathbb{R}^{(T-\ell)\times qT}.$$

More generally, for a polynomial matrix (R), the multiplication matrix $\mathscr{M}_T(R)$ with $T \geq \max(\ell_1, \ldots, \ell_g)$ block-columns is defined as

$$\mathscr{M}_T(R) := \begin{bmatrix} \mathscr{M}_T(R^1) \\ \vdots \\ \mathscr{M}_T(R^g) \end{bmatrix} \in \mathbb{R}^{(gT-\ell_1-\cdots-\ell_g)\times qT}.$$

As the name suggests, the multiplication matrix is used to convert a product $C(z) = A(z)B(z)$ of polynomials $A \in \mathbb{R}^{g\times q_a}[z]$ and $B \in \mathbb{R}^{q_a\times q_b}[z]$ to a matrix–vector product $C = A\mathscr{M}_{\deg A+\deg B+1}(B)$.

Using the multiplication matrix, we obtain the following matrix representation of the action of the polynomial operator $r(\sigma)$, where $r \in \mathbb{R}^{1\times q}[z]$, on a finite signal $w \in (\mathbb{R}^q)^T$:

$$r(\sigma)w = 0 \quad \iff \quad \mathscr{M}_T(r)w = 0. \qquad (r(\sigma)w \leftrightarrow \mathscr{M}_T(r)w)$$

The generalization of $(r(\sigma)w \leftrightarrow \mathscr{M}_T(r)w)$ to a polynomial operator $R(\sigma)$ defined by a matrix polynomial $R \in \mathbb{R}^{g\times q}[z]$ requires R to be row-proper, equivalently the kernel representation to be shortest-lag [13].

2.7 Nonparameteric representation by a basis of the finite-horizon behavior

The input/state/output and kernel representations describe the infinite-horizon behavior of a bounded complexity linear time-invariant system $\mathscr{B}$ via recursive equations. In this section, we fix $T \in \mathbb{N}$ and consider the horizon-T behavior $\mathscr{B}|_T$ of a system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$. Since $\mathscr{B}|_T$ is a linear subspace of dimension (see (ℓ_i) and [13, Corollary 5])

$$r := \dim \mathscr{B}|_T = \boldsymbol{m}(\mathscr{B})T + \boldsymbol{n}(\mathscr{B}) \leq qT, \quad \text{for } T \geq \boldsymbol{\ell}(\mathscr{B}), \qquad (\dim \mathscr{B}|_T)$$

it can be represented as

$$\mathscr{B}|_T = \operatorname{image} B_T, \tag{B_T}$$

where $B_T \in \mathbb{R}^{qT\times r}$ is a suitably chosen full column rank matrix. Equivalently, for any $w \in \mathscr{B}|_T$, there is a vector $\alpha \in \mathbb{R}^r$, such that $w = B_T\alpha$ and $w := B_T\alpha \in \mathscr{B}|_T$ for any $\alpha \in \mathbb{R}^r$. The representation (B_T) is nonparametric because its parameter B_T grows with the horizon T. The infinite-horizon behavior $\mathscr{B}$ however is fully characterized by its restriction $\mathscr{B}|_T$ to horizon $T \geq \boldsymbol{\ell}(\mathscr{B})+1$, thus (B_T) is a representation of $\mathscr{B}$ [13, Lemma 13].

The finite-horizon behavior $\mathscr{B}|_T$ as well as any basis B_T for it have the "hidden" bounded complexity linear time-invariant structure presented in Section 2.3. The sets of finite-horizon behaviors $\mathscr{B}|_T$'s and the set of their parameters B_T that correspond to bounded complexity linear time-invariant systems $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ are lower-dimensional manifolds of $(\mathbb{R}^q)^T$ and $\mathbb{R}^{qT\times(mT+n)}$, respectively. Therefore, randomly selected subspace of $\mathbb{R}^{qT}$ with dimension $mT+n$ and full column rank matrix $B_T \in \mathbb{R}^{qT\times(mT+n)}$ with probability one do not correspond to a $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$.

The representation (B_T) exists for any linear system and makes no a priori separation of the variables into inputs and outputs. In contrast, classical non-parametric representations of linear systems, such as convolution and frequency response function, view a dynamical system as a map $u \mapsto y$ from an input u to an output y. Thus, 1) they require an input/output partitioning of the variables and 2) assume zero initial conditions. While the variables can always be separated into inputs and outputs, the zero initial conditions assumption is restrictive. Another difference between (B_T) and classical non-parametric representations is that the parameter B_T of (B_T) is finite-dimensional while classical non-parametric representations require in general infinite-dimensional parameters, such as the impulse response and the frequency response functions. Finally, (B_T) can be obtained from a finite-length trajectory of the system, which makes it a stepping stone towards direct data-driven methods. This is the topic of the next section.

2.8 Data-driven representation by the image of a Hankel matrix

The representation (B_T) can be obtained from another representation of the system or from raw data—trajectories of the system. Obtaining (B_T) from data is based on the fundamental lemma, which gives a data-driven representation of the system, parameterized by the data itself. Consider a trajectory $w_\mathrm{d} \in \mathscr{B}|_{T_\mathrm{d}}$ of a linear time-invariant system $\mathscr{B} \in \mathscr{L}^q$ and an integer L, $1 \leq L \leq T_\mathrm{d}$. The *Hankel matrix* of w_d with L block-rows is defined as

$$\mathscr{H}_L(w_\mathrm{d}) := \begin{bmatrix} w_\mathrm{d}(1) & w_\mathrm{d}(2) & \cdots & w_\mathrm{d}(T_\mathrm{d}-L+1) \\ w_\mathrm{d}(2) & w_\mathrm{d}(3) & \cdots & w_\mathrm{d}(T_\mathrm{d}-L+2) \\ \vdots & \vdots & & \vdots \\ w_\mathrm{d}(L) & w_\mathrm{d}(L+1) & \cdots & w_\mathrm{d}(T_\mathrm{d}) \end{bmatrix}. \tag{$\mathscr{H}_L(w_\mathrm{d})$}$$

An alternative way to define the Hankel matrix is via consecutive application of the shift and restrict operators on w_d:

$$\mathscr{H}_L(w_\mathrm{d}) = \begin{bmatrix} (\sigma^0 w_\mathrm{d})|_L & (\sigma^1 w_\mathrm{d})|_L & \cdots & (\sigma^{T_\mathrm{d}-L} w_\mathrm{d})|_L \end{bmatrix} \in \mathbb{R}^{qL\times(T_\mathrm{d}-L+1)}.$$

The former definition shows the structure of the Hankel matrix—the block elements on the anti-diagonals are equal—while the latter definition shows a systems theoretic interpretation of the Hankel matrix—for exact data, generated by a linear time-invariant system, the columns of $\mathscr{H}_L(w_\mathrm{d})$ are L-samples long trajectory of the system. Thus, using one T_d-samples long trajectories, the Hankel operator $\mathscr{H}_L$ constructs $T_\mathrm{d}-L+1$, L-samples long trajectories.

By the assumptions that $w_\mathrm{d} \in \mathscr{B}|_{T_\mathrm{d}}$ and $\mathscr{B} \in \mathscr{L}^q$, it follows from the definition of the Hankel matrix that

$$\operatorname{image} \mathscr{H}_L(w_\mathrm{d}) \subseteq \mathscr{B}|_L. \tag{$*$}$$

As shown in [13, Corollary 21], if $\mathscr{B}$ is moreover bounded complexity and

$$\operatorname{rank} \mathscr{H}_L(w_\mathrm{d}) = \boldsymbol{m}(\mathscr{B})L + \boldsymbol{n}(\mathscr{B}), \tag{GPE}$$

the non-strict inequality $(*)$ holds with equality, *i.e.*,

$$\mathscr{B}|_L = \operatorname{image} \mathscr{H}_L(w_\mathrm{d}). \tag{DDR}$$

Equation (DDR) is called a *data-driven representation* of the finite-horizon behavior $\mathscr{B}|_L$ and (GPE) is called a *generalized persistency of excitation* condition for (DDR).

The generalized persistency of excitation condition (GPE) is necessary and sufficient for the data-driven representation (DDR). Alternative sufficient conditions are given by the fundamental lemma [25, Theorem 1]. They require an input/output partitioning of the variables, persistency of excitation of the input component u_d of w_d and controllability of the data generating system $\mathscr{B}$. These conditions are suitable for input design as they restrict the input and the data generating system in order to guarantee (DDR) for any initial condition, under which the data collection experiment is performed. In contrast, the generalized persistency of excitation is suitable for checking when a given trajectory w_d can represent $\mathscr{B}|_L$. Since the behavior $\mathscr{B}$ is fully characterized by the restriction $\mathscr{B}|_T$ to horizon $T = \boldsymbol{\ell}(\mathscr{B}) + 1$ [13, Lemma 13], an exact finite trajectory $w_\mathrm{d} \in \mathscr{B}|_{T_\mathrm{d}}$ that satisfies the (GPE) condition describes fully the system $\mathscr{B}$.

The data-driven representation (DDR) can be trivially extended for data consisting of multiple trajectories

$$\mathscr{W}_\mathrm{d} := \{ w_\mathrm{d}^1, \ldots, w_\mathrm{d}^N \}, \quad \text{where } w_\mathrm{d}^i \in \mathscr{B}|_{T_\mathrm{d}^i}.$$

The only modification needed in (GPE) and (DDR) is to replace the Hankel matrix ($\mathscr{H}_L(w_\mathrm{d})$) by the *mosaic-Hankel matrix* [4, 16]

$$\mathscr{H}_L(\mathscr{W}_\mathrm{d}) := \begin{bmatrix} \mathscr{H}_L(w_\mathrm{d}^1) & \cdots & \mathscr{H}_L(w_\mathrm{d}^N) \end{bmatrix} \in \mathbb{R}^{qL \times \sum_{i=1}^N (T_\mathrm{d}^i - L)}. \qquad (\mathscr{H}_L(\mathscr{W}))$$

Similarly, generalization of the fundamental lemma [25, Theorem 1] to data consisting of multiple trajectories requires only replacement of the Hankel matrix by the mosaic-Hankel matrix in the persistency of excitation condition [19].

Using (DDR), a basis B_T for the finite-horizon behavior $\mathscr{B}|_T$ can be computed from a given trajectory w_d of the system. This involves a rank revealing factorization of the Hankel matrix $\mathscr{H}_T(w_\mathrm{d})$—a classical problem in linear algebra, for which different methods exist. The most reliable one is the singular value decomposition (SVD) combined with thresholding of the singular values. Indeed, the SVD truncation is a method for computing an optimal low-rank approximation. The Hankel structure of $\mathscr{H}_T(w_\mathrm{d})$ however is not preserved in the approximation obtained by the SVD truncation. Preserving the Hankel structure while reducing the rank is a nonconvex optimization problem called *structured low-rank approximation* [9].

$w \in (\mathbb{R}^q)^\mathbb{N}$, $w : \mathbb{N} \to \mathbb{R}^q$	q-variate real discrete-time signal with time axis $\mathbb{N}$
$w\|_T := \big(w(1), \ldots, w(T)\big)$	restriction of w to the interval $[1, T]$
$w = w_\mathrm{ini} \wedge w_\mathrm{f}$	concatenation of trajectories w_ini and w_f
σ, $(\sigma w)(t) := w(t+1)$	unit shift operator
$\mathscr{B} \subset (\mathbb{R}^q)^\mathbb{N}$	discrete-time dynamical system with q variables
$\mathscr{B}\|_T := \{ w\|_T \mid w \in \mathscr{B} \}$	restriction of $\mathscr{B}$ to the interval $[1, T]$
$\mathscr{L}^q$	set of linear time-invariant systems with q variables
$\boldsymbol{m}(\mathscr{B})$ / $\boldsymbol{\ell}(\mathscr{B})$ / $\boldsymbol{n}(\mathscr{B})$	number of inputs / lag / order of $\mathscr{B}$
$\boldsymbol{c}(\mathscr{B}) := \big(\boldsymbol{m}(\mathscr{B}), \boldsymbol{\ell}(\mathscr{B}), \boldsymbol{n}(\mathscr{B})\big)$	complexity of $\mathscr{B}$
$\mathscr{L}^q_c := \{ \mathscr{B} \in \mathscr{L}^q \mid \boldsymbol{c}(\mathscr{B}) \leq c \}$	set of bounded complexity linear time-invariant systems
$\ker R(\sigma)$	kernel representation
$\mathscr{B}_\mathrm{i/s/o}(A, B, C, D)$	input/state/output representation
$\mathscr{M}_T(R)$	multiplication matrix with T block columns, constructed from R
$\mathscr{H}_T(w)$	Hankel matrix with T block rows, constructed from w
$\lceil a \rceil$ / $\lfloor a \rfloor$	rounding to the nearest integer larger / smaller than a

Table 1: Summary of notation.

3 Problems and methods for their solution

Problems in systems and control are usually classified into analysis and design. Examples of analysis problems are finding the lag structure, finding an input/output partitioning of the variables, and checking controllability. Examples of control problems are state transfer, optimal tracking, and control by interconnection. In addition to analysis and

design, other classes of problems related to dynamical systems are identification, missing data recovery, and estimation. Identification aims to obtain a system from data—trajectories of the unknown system. Identification problems can be further on classified into exact, approximate, and stochastic. In missing data recovery, an incomplete trajectory of the system is given and the goal is to complete the missing values. Special cases of missing data recovery are simulation, finding initial/final conditions, and recovering some variables of the system from other variables. Closely related problems to the one of exact recovery of missing values, are the one of approximate recovery of missing values and approximation of given inexact values. Stochastic equivalents of these deterministic approximation problems are errors-in-variables smoothing, Kalman smoothing, and input estimation. Missing data recovery and approximation are unified in [12] by a problem of trajectories interpolation and approximation. In the problem setup considered in [12], the data-generating system is implicitly specified by a given complete trajectory of the system. A more general (and harder) problem of system identification from data with missing values is posed and solved in [10, 1].

Using the behavioral language, we define problems without reference to a representation of the system. However, for solving the problems in practice, we use representations. The choice of the representation is the most important first step in solving a problem because it determines the analytical and computational tools used. For example, choosing the kernel representation (KER) naturally leads to polynomial algebra because the parameter $R(z)$ is a polynomial matrix. Indeed, the results lead to computational problems such as checking rank over the ring of polynomials, checking left/right primness, Smith decomposition, *etc*. The algorithms for solving problems in polynomial algebra however are arguably less developed and not as widely available as the ones for matrix computations.

In addition to determining the analytical and computational tools for solving problems in systems and control, some representation have limitations of what systems they can represent. For example, the transfer function has the limitations of representing controllable systems, under zero initial conditions, with a priori given input/output partitioning of the variables. This excludes autonomous systems, the response of which is only due to the initial conditions. Also, the assumption that the data comes separated into inputs and outputs is more restrictive than generally perceived—it is unrealistic in applications where input selection is part of the problem formulation, *e.g.*, dynamic networks.

A problem can be solved using a different representation than the given one. The power of systems theory is the knowledge of transitioning from any given representation to any other representation. Behind a transition there is a theoretical result that reveals the connection. Some of the results are trivial—formulas relating the corresponding parameters, others are more involved and require a chain of steps. For example, the conversion from an impulse response to an input/state/output representation is the realization problem [5]. The large number of possible transitions among the various representations of linear time-invariant systems makes the subject a fruitful area for development of numerical methods. On the one hand, connections among the various representations lead to deeper understanding of systems theory. On the other hand, the methods resulting from these connections are essential for solving analysis and design problems. Different representations allow us to access a broader set of analytical and computational tools.

> "The best way to come up with a good solution is to come up with many solutions." T. Kailath [6]

The representations introduced in Section 2:

- minimal input/state/output (I/S/O),
- minimal / shortest-lag kernel (KER),
- basis for the finite-horizon behavior (B_T), and
- data-driven (DDR)

rank from most structured to least structured as listed above. The data-driven representation (DDR) is parametrized by the raw data and does not display any structure of the system. The basis representation (B_T) is a "compressed" version of (DDR). Everything that can be done with (DDR), can be done also with (B_T), but more efficiently, especially if (B_T) is an orthonormal basis. Note that (B_T) displays the dimension of $\mathscr{B}|_T$, which gives a partial information about the complexity of $\mathscr{B}$. The kernel representation is parametric and, when minimal, it displays both the number of inputs $\boldsymbol{m}(\mathscr{B})$ and the lag $\boldsymbol{\ell}(\mathscr{B})$ of the system: $\boldsymbol{m}(\mathscr{B}) = q - \operatorname{rowdim} R$ and $\boldsymbol{\ell}(\mathscr{B}) = \deg R$. If the kernel representation is, in addition shortest-lag, then it displays also the structure indices $(\ell_1, \ldots \ell_p)$ of the system. A minimal (I/S/O) is parametric and displays the number of inputs $\boldsymbol{m}(\mathscr{B})$, the order $\boldsymbol{n}(\mathscr{B})$, and an input/output partitioning of the variables.

In the conversion of one representation to another, the direction of going from a less structured one to a more structured one is harder because it involves finding additional structure of the system. In particular, going from a non-parametric to a parametric representation requires finding the system's complexity. Going from a trajectory of the system to (B_T), (KER), or (I/S/O), are *exact identification* methods, while the other directions—going from (B_T), (KER), or (I/S/O) to a trajectory of the system—are *simulation* methods.

Next, we present methods for the tasks of finding the system's complexity, finding an input/output partitioning of the variables, and checking controllability. In all cases, we start from raw data of the system and the methods are easily implementable by standard numerical linear algebra operations and lead to practical numerical algorithms.

4 Examples

Using the perspective brought by the behavioral approach and the nonparametric representation of the finite-horizon behavior (B_T), we derive new methods for solving problems in systems theory. The first example is finding the integer invariants $\boldsymbol{m}(\mathscr{B})$ and $\boldsymbol{n}(\mathscr{B})$ directly from data. This is a more general problem than the classical one of order selection. Indeed, in the classical setting, the number of inputs $\boldsymbol{m}(\mathscr{B})$ is assumed a priori given since the given data is partitioned into inputs and outputs. The second example is a logical continuation of the first one: once we have detected the number of inputs, the question occurs of how to find an input/output partitioning of the variables. We present a direct data-driven test to verify if a given partitioning of the variables is a valid input/output partitioning for the data-generating system. Finally, we consider the problem of checking controllability. First, we present a method based on a kernel representation, which is a direct consequence of the controllability definition using ($r(\sigma)w \leftrightarrow \mathscr{M}_T(r)w$). Then, we present a direct data-driven test based on a decomposition of a system into an autonomous subsystem and a controllable subsystem. The computation of the controllable subsystem is based on (B_T).

4.1 Finding the system's complexity

Recall from Section 2.3, that the complexity $\boldsymbol{c}(\mathscr{B})$ of a linear time-invariant system $\mathscr{B}$ is defined as the triple: number of inputs $\boldsymbol{m}(\mathscr{B})$, lag $\boldsymbol{\ell}(\mathscr{B})$, and order $\boldsymbol{n}(\mathscr{B})$. Although in the classical setting the order $\boldsymbol{n}(\mathscr{B})$ is defined via a state-space representation, it is a property of the system $\mathscr{B}$ and can be computed directly from the restricted behavior $\mathscr{B}|_T$, provided that $T \geq \boldsymbol{\ell}(\mathscr{B}) + 1$. The number of inputs $\boldsymbol{m}(\mathscr{B})$ also can be computed from $\mathscr{B}|_T$ without a reference to an input/output representation. Evaluating $\dim \mathscr{B}|_{t_i}$ for $t_1 \neq t_2 \geq \boldsymbol{\ell}(\mathscr{B})$, *e.g.*, $t_1 = T$, $t_2 = T-1$, and using ($\dim \mathscr{B}|_T$), we have the system of equations

$$\begin{bmatrix} T & 1 \\ T-1 & 1 \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} = \begin{bmatrix} \dim \mathscr{B}|_T \\ \dim \mathscr{B}|_{T-1} \end{bmatrix},$$

from which $m = \boldsymbol{m}(\mathscr{B})$ and $n = \boldsymbol{n}(\mathscr{B})$ can be found. The evaluation of $\dim \mathscr{B}|_t$ requires a rank computation. Denoting with $B_T|_t$ the selection of the first t block-rows (*i.e.*, the first qt rows) of B_T, we have

$$\mathscr{B}|_t = \operatorname{image} B_T|_t, \quad \text{for } t = 1, \ldots, T,$$

and therefore

$$\dim \mathscr{B}|_t = \operatorname{rank} B_T|_t, \quad \text{for } t = 1, \ldots, T. \tag{dim-rank-B_T}$$

If a trajectory $w_\mathrm{d} \in \mathscr{B}|_{T_\mathrm{d}}$ is given that satisfies the (GPE) condition instead of a representation (B_T) of the system, $\dim \mathscr{B}|_t$ is computed by a rank test of the Hankel matrix $\mathscr{H}_t(w_\mathrm{d})$, *i.e.*, in (dim-rank-$B_T$) $B_T|_t$ is replaced by $\mathscr{H}_t(w_\mathrm{d})$:

$$\dim \mathscr{B}|_t = \operatorname{rank} \mathscr{H}_t(w_\mathrm{d}), \quad \text{for } t = 1, \ldots, T. \tag{dim-rank-w_d}$$

Implementation details about the above outlined method for finding the system's complexity from data are the selection of the horizon T and the algorithm for the rank computation. If there is no prior knowledge about the lag of the data-generating system, the horizon T should be chosen as large as possible in order to maximize the chance that the condition $T \geq \boldsymbol{\ell}(\mathscr{B}) + 1$ is satisfied. The largest possible horizon given a finite trajectory $w_\mathrm{d} \in \mathscr{B}|_{T_\mathrm{d}}$ is

$$T_\mathrm{max} := \left\lfloor \frac{T_\mathrm{d} + 1}{q + 1} \right\rfloor - 1.$$

When an upper bound $\bar{\ell}$ for the lag of the system is a priori known, the horizon T is chosen as $T = \bar{\ell} + 1$, in order to minimize the computational cost.

Concerning the method for the rank computation, the most reliable one is thresholding the singular values. There is no general rule for the choice of the tolerance. This is a major issue in computations involving inexact / noisy data. The tolerance is a hyper-parameter that reflects prior knowledge about the expected perturbation on the data.

The method described above is implemented in the Behavioral Toolbox. The function `c_mpum` computes the complexity from a given trajectory $w_{\mathrm{d}} \in (\mathbb{R}^q)^{T_{\mathrm{d}}}$ while the function `BT2c` computes the complexity from a given representation (B_T). Here is an example of applying `c_mpum` on a randomly generated trajectory of a random system:

```
m = 3; p = 4; n = 5; Td = 100;   % simulation parameters
B = drss(n, p, m);               % random stable LTI system
wd = B2w(B, Td);                 % random data trajectory
[c, mh, ell, nh] = c_mpum(wd);   % complexity computation from data
[m == mh, n == nh] % -> [1 1]    % check if the computed result is correct
```

4.2 Finding an input/output partitioning of the variables

A partitioning of the variables $w(t) \in \mathbb{R}^q$ into inputs $u(t) \in \mathbb{R}^m$ and outputs $y(t) \in \mathbb{R}^p$ is defined by a permutation matrix $\Pi \in \mathbb{R}^{q \times q}$ as follows:

$$w \mapsto (u,y) : \begin{bmatrix} u \\ y \end{bmatrix} := \Pi w \qquad \text{and} \qquad (u,y) \mapsto w : \; w = \Pi^\top \begin{bmatrix} u \\ y \end{bmatrix}. \tag{I/O}$$

Πw reorders the variables w, so that the first m variables are the inputs and the remaining $p := q - m$ variables are the outputs. The number of inputs m is uniquely defined by the behavior. However, an input/output partitioning of the variables is in general not unique. The currently available methods in the literature for finding an input/output partitioning of a system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ are based on parametric representations of the system, *e.g.*, the kernel representation.

Next, we describe a method for testing if a partitioning (I/O) is a valid input/output partitioning of a system specified by (B_T) or a trajectory $w_{\mathrm{d}} \in \mathscr{B}|_{T_{\mathrm{d}}}$ satisfying (GPE). As in Section 4.1, the solution for the latter case follows trivially from the former, so we will skip it.

By definition, (I/O) is an input/output partitioning of $\mathscr{B}$ if and only if it is possible to simulate any trajectory of $\mathscr{B}$ by choosing the input u and the initial conditions that are specified by a past trajectory w_{ini} [15, Lemma 1]. Let $T_{\mathrm{ini}} \geq \ell$ and $T_{\mathrm{f}} \geq \ell$. We have that

$$\begin{gathered} \text{(I/O) is an input/output partitioning of } \mathscr{B} \\ \Updownarrow \\ \text{for all } w_{\mathrm{ini}} \in \mathscr{B}|_{T_{\mathrm{ini}}} \text{ and } u_{\mathrm{f}} \in (\mathbb{R}^m)^{T_{\mathrm{f}}}, \text{ there is a unique } y_{\mathrm{f}} \in (\mathbb{R}^p)^{T_{\mathrm{f}}}, \\ \text{such that } w_{\mathrm{ini}} \wedge \Pi^\top \begin{bmatrix} u_{\mathrm{f}} \\ y_{\mathrm{f}} \end{bmatrix} \in \mathscr{B}|_{T_{\mathrm{ini}}+T_{\mathrm{f}}}. \end{gathered} \tag{valid-I/O}$$

Let $B_{T_{\mathrm{ini}}+T_{\mathrm{f}}}$ be the matrix of basis vectors for $\mathscr{B}|_{T_{\mathrm{ini}}+T_{\mathrm{f}}}$ and let $\begin{bmatrix} W_{\mathrm{ini}} \\ U_{\mathrm{f}} \end{bmatrix}$ be the submatrix of $B_{T_{\mathrm{ini}}+T_{\mathrm{f}}}$, corresponding to the initial trajectory w_{ini} and the input u_{f}. Then, (valid-I/O) is equivalent to

$$\operatorname{rank} \begin{bmatrix} W_{\mathrm{ini}} \\ U_{\mathrm{f}} \end{bmatrix} = m(T_{\mathrm{ini}} + T_{\mathrm{f}}) + n.$$

The method is implemented in the function `is_io` of the Behavioral Toolbox. The function `BT2IO` finds all possible input/output partitionings of the system by exhaustively trying all possible permutations Π. In order to test `is_io` and `BT2IO`, consider the following single-input single-output system:

```
B = ss(tf([0 1], [1 1], 1)); q = 2;
```

that has an input/output partitioning $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ but not $w = \left[\begin{smallmatrix} y \\ u \end{smallmatrix}\right]$. First, using the function `B2BT` we obtain (B_T)

```
T = 10; BT = B2BT(Bp, T);
```

Then, using (B_T) we test if the partitionings $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ and $w = \left[\begin{smallmatrix} y \\ u \end{smallmatrix}\right]$ are valid:

```
is_io(BT, q, [1 2]) % -> 1
is_io(BT, q, [2 1]) % -> 0
```

Finally, we do an exhaustive search of valid input/output partitionings:

```
BT2IO(BT, q)        % -> [1 2]
```

4.3 Checking controllability

In this section, we present both model-based and data-driven methods for checking if a system is controllable. The model-based method is based on a kernel representation of the system and the data-driven method is based on the representation (B_T). The methods require only standard numerical linear algebra operations.

Model-based test, using a kernel representation

Recall the definition of controllability in the behavioral setting, given in Section 2.2. For a bounded complexity linear time-invariant system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$, the "past" and the "future" trajectories can be restricted to length $\boldsymbol{\ell}(\mathscr{B})$, while t_c can be chosen as the order of the system.[1] Thus, we have the following definition.

> *State transfer problem for bounded complexity linear time-invariant system:* Given a system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$, a "past" trajectory $w_p \in \mathscr{B}|_\ell$, which specifies an initial condition and a "future" trajectory $w_f \in \mathscr{B}|_\ell$, which specifies a terminal condition, find a "control" trajectory $w_c \in \mathscr{B}|_n$, such that $w_p \wedge w_c \wedge w_f \in \mathscr{B}|_{2\ell+n}$.

Consider a shortest-lag kernel representation (KER) of $\mathscr{B}$. Using $(r(\sigma)w \leftrightarrow \mathscr{M}_T(r)w)$, we have that

$$w_p \wedge w_c \wedge w_f \in \mathscr{B}|_{2\ell+n} \quad \Longleftrightarrow \quad \mathscr{M}_{2\ell+n}(R)w = 0 \quad \Longleftrightarrow : \quad \begin{bmatrix} M_p & M_c & M_f \end{bmatrix} \begin{bmatrix} w_p \\ w_c \\ w_f \end{bmatrix} = 0. \qquad (M_c)$$

Finding w_c then reduces to solving the system of linear equations

$$M_c w_c = -\begin{bmatrix} M_p & M_f \end{bmatrix} \begin{bmatrix} w_p \\ w_f \end{bmatrix}.$$

The existence of a solution w_c for arbitrary w_p and w_f is equivalent to M_c being full row-rank, which is an algebraic test for controllability.

Theorem 1 (Controllability test using shortest-lag kernel representation)**.** *The system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ with a shortest-lag kernel representation (KER) is controllable if and only if the matrix M_c defined in (M_c) is full row-rank.*

Note that in Theorem 1 the given kernel representation has to be shortest-lag. This is due to the fact that in the derivation of the result we used $(r(\sigma)w \leftrightarrow \mathscr{M}_T(r)w)$, which requires R to be row-proper.

The matrix M_c is the *generalized Sylvester matrix* of $R(z)$ [2, 3]. Full row-rank of the generalized Sylvester matrix is equivalent to left primness of R. In the special case of a single-input single-output system $\mathscr{B} \in \mathscr{L}^2_{(1,\ell,n)}$,

$$R(z) =: \begin{bmatrix} R_1(z) & R_2(z) \end{bmatrix} \in \mathbb{R}^{1\times 2}[z],$$

M_c is the *Sylvester matrix* of $R_1(z)$ and $R_2(z)$, and M_c being full row-rank is equivalent to $R_1(z)$ and $R_2(z)$ being co-prime.

[1] The smallest value of t_c that can be chosen is the controllability index of the system.

Data-driven test, using (B_T)

Data-driven analysis of controllability from input-state data is considered in [20, Section III]. Theorem 8 in [20] gives a Hautus-like controllability test in terms of input-state data. The restriction to input-state data is relaxed in [17, Section IV]. Theorem 3 in [17] gives a Hautus-like controllability test that involves Hankel matrices constructed from a general trajectory of the system.

Although the approach of [20] is aimed at data-driven analysis, it is essentially based on the view of a system as a state-space representation and controllability as a property of this representation. The resulting Hautus-like tests in [20] and [17] require checking the rank of a pencil $A+\lambda B$ for all complex numbers λ, which in turn requires solving a generalized eigenvalue problem, in addition to checking the rank of multiple real matrices.

In this section, we start from the definition of controllability in the behavioral setting as a property of the system and derive the test as a consequence of this definition—an important conceptual difference. The resulting method for checking controllability is different from existing controllability tests in the literature. It requires a single rank test of a real matrix computed from a general trajectory of the system. Thus, the method proposed here is computationally advantageous in comparison with the method of [17].

The method is based on the following result from [22, Proposition V.8]: Any system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ admits a direct sum decomposition into an autonomous subsystem $\mathscr{B}_{\mathrm{aut}} \in \mathscr{L}^q_{(0,\ell,n)}$ and a controllable subsystem $\mathscr{B}_{\mathrm{c}} \in \mathscr{L}^q_{(m,\ell,n)}$

$$\mathscr{B} = \mathscr{B}_{\mathrm{aut}} \oplus \mathscr{B}_{\mathrm{c}}. \qquad \text{(aut}\oplus\text{ctrb)}$$

Using (aut$\oplus$ctrb), we can check if $\mathscr{B}$ is controllable by checking if $\mathscr{B}_{\mathrm{aut}} = \{0\}$, or equivalently, $\mathscr{B} = \mathscr{B}_{\mathrm{c}}$. The problem of checking controllability therefore reduces to the one of finding $\mathscr{B}_{\mathrm{c}}$.

Consider a representation (B_T) of $\mathscr{B}$ with horizon $T = 2\ell + n$. Since $\mathscr{B}_{\mathrm{c}} \subseteq \mathscr{B}$, for checking controllability, it suffices to find only the dimension of $\mathscr{B}_{\mathrm{c}}|_\ell$:

$$\mathscr{B} \text{ is controllable} \quad \Longleftrightarrow \quad \dim \mathscr{B}_{\mathrm{c}}|_\ell = \boldsymbol{m}(\mathscr{B})\ell + \boldsymbol{n}(\mathscr{B}).$$

Moreover,

$$n_{\mathrm{uc}} := \boldsymbol{m}(\mathscr{B})\ell + \boldsymbol{n}(\mathscr{B}) - \dim \mathscr{B}_{\mathrm{c}}|_\ell$$

is the number of uncontrollable modes of $\mathscr{B}$.

In order to find the controllable subspace $\mathscr{B}_{\mathrm{c}}|_\ell$, partition the parameter B_T of (B_T) as follows:

$$B_T =: \begin{bmatrix} B_{\mathrm{p}} \\ B_{\mathrm{m}} \\ B_{\mathrm{f}} \end{bmatrix} \begin{matrix} \} \\ \} \\ \} \end{matrix} \begin{matrix} q\ell \\ qn \\ q\ell \end{matrix}.$$

Let the columns of Z forms a basis for the nullspace of B_{p}. By construction, any $w_{\mathrm{f}} \in \text{image}\, B_{\mathrm{f}} Z$ is obtained from initial conditions that are reachable using state-transfer control from zero initial conditions. Therefore, image $B_{\mathrm{f}} Z \subseteq \mathscr{B}_{\mathrm{c}}|_\ell$. Moreover, image $B_{\mathrm{f}} Z$ contains all trajectories $w_{\mathrm{f}} \in \mathscr{B}_{\mathrm{c}}|_\ell$ because Z is a basis for $\ker B_{\mathrm{p}}$. Therefore, $\mathscr{B}_{\mathrm{c}}|_\ell = \text{image}\, B_{\mathrm{f}} Z$. The method is implemented in the function `isunct` of the Behavioral Toolbox.

Numerical example

We construct an uncontrollable system via the decomposition (aut$\oplus$ctrb) using a random stable autonomous system $\mathscr{B}_{\mathrm{c}} \in \mathscr{L}^q_{(0,\ell_{\mathrm{aut}},n_{\mathrm{aut}})}$ and a random stable controllable system $\mathscr{B}_{\mathrm{c}} \in \mathscr{L}^q_{(m,\ell_{\mathrm{c}},n_{\mathrm{c}})}$:

```
m = 1; p = 1; q = m + p; n_ctr = 3; n_aut = 2; n = n_aut + n_ctr;
B_ctr = drss(n_ctr, p, m); B_aut = drss(n_aut, q, 0);
T = 3 * n; BT = orth([B2BT(B_aut, T), B2BT(B_ctr, T)]);
```

A finite-horizon representation of (aut$\oplus$ctrb) is obtained from the finite-horizon representations of $\mathscr{B}_{\mathrm{aut}}$ and $\mathscr{B}_{\mathrm{c}}$ by appending the matrices of their basis vectors. With probability one, the resulting system $\mathscr{B} \in \mathscr{L}^q_{(m,\ell,n)}$ has order $n = n_{\mathrm{aut}} + n_{\mathrm{c}}$ so that it has n_{aut} uncontrollable modes.

```
[~, m, ell, n] = BT2c(BT, q) % -> 1, 5, 5
```

In order to apply the model-based method, first we derive a kernel representation of $\mathscr{B}$. This is done via the function `BT2R` from the Behavioral Toolbox:

```
R = BT2R(BT, q);
```

Since the system is single output, the computed representation is shortest-lag. In general, however, an additional transformation to a shortest-lag kernel representation is needed before the controllability test is applied. Using the function `multmat`, we construct the matrix M_c:

```
M = multmat(R, q, 3 * ell); Mc = M(:, q * ell + 1:q * 2 * ell);
```

Finally, the test gives

```
n_unctr = size(Mc, 2) - rank(Mc, 1e-10) % -> 2
```

Applying `isunct` on the example, we obtain

```
n_unctr = isunctr(BT, q, 1e-10) % -> 2
```

5 Outlook

In its "early" years, the behavioral approach was a purely theoretical field of research. Due to lack of supporting computational methods, it had no impact on real-life applications. This is now changed with the advent of data-driven methods that are suitable for numerical computations and proved to be effective in real-life applications.

On the one hand, the behavioral approach benefits from the advent of data-driven methods because they make the theory applicable in practice. On the other hand, direct data-driven analysis and design benefits from the behavioral approach as a theoretical framework. The paper illustrated the synergy of the two fields by clarifying the value of separating the notions of dynamical system and representation, which in turn leads to representation-free definition of systems properties and problem formations.

We introduced a nonparametric representation of the finite-horizon behavior of a bounded complexity linear time-invariant system and used it for development of computational methods. Obtaining the representation directly from data is a system identification problem that requires a rank revealing factorization of a Hankel matrix constructed from the given data. For large time-horizons as well as for dealing with noise in the data, it is important to exploit and preserve the Hankel structure. This leads to structure exploiting low-rank approximation methods.

The synergy of methods developed by the numerical linear algebra community and direct data-driven methods for analysis and design is still a largely unexplored area of research. The structured matrices appearing in behavioral systems theory are: the mosaic Hankel matrix $\mathscr{H}_T(\mathscr{W})$ and the polynomial multiplication matrix $\mathscr{M}_T(R)$. As illustrative examples of the ideas presented in the paper, we considered the analysis problems of computing the systems complexity, finding an input/output partitioning of the variables, and checking controllability. The developed methods are simple to implement and have the potential to be generalized to approximate computations in case of noisy data.

6 Acknowledgment

The research leading to these results has received funding from: the Catalan Institution for Research and Advanced Studies (ICREA), the Fond for Scientific Research Vlaanderen (FWO) project G033822N, the Spanish Ministry of Science State Research Agency MCIU/AEI/10.13039/501100011033, grant PID2023-148952OB-I00, and the European Research Council (ERC) grant agreement No 948679.

References

[1] M. Alsalti, I. Markovsky, V. G. Lopez, and M. A. Müller. "Data-based system representations from irregularly measured data". In: *IEEE Trans. Automat. Contr.* 70 (2025), pp. 143–158.

[2] B. D. O. Anderson and E. I. Jury. "Generalized Bezoutian and Sylvester matrices in multivariable linear control". In: *IEEE Transactions on Automatic Control* 21.4 (1976), pp. 551–556.

[3] R. R. Bitmead, S. Y. Kung, B. D. O. Anderson, and T. Kailath. “Greatest Common Divisors via Generalized Sylvester and Bezout Matrices”. In: *IEEE Transactions on Automatic Control* 23.6 (1978), pp. 1043–1047.

[4] G. Heinig. “Generalized inverses of Hankel and Toeplitz mosaic matrices”. In: *Linear Algebra Appl.* 216 (1995), pp. 43–59.

[5] B. L. Ho and R. E. Kalman. “Effective construction of linear state-variable models from input/output functions”. In: *Regelungstechnik* 14.12 (1966), pp. 545–592.

[6] T. Kailath. *Linear Systems*. Prentice Hall, 1981.

[7] T. Kailath and A. Sayed. “Displacement structure: theory and applications”. In: *SIAM Review* 37.3 (1995), pp. 297–386.

[8] T. Kailath and A. Sayed. *Fast reliable algorithms for matrices with structure*. SIAM, Philadelphia, 1999.

[9] I. Markovsky. *Low-Rank Approximation: Algorithms, Implementation, Applications*. Springer, 2019.

[10] I. Markovsky. “On the most powerful unfalsified model for data with missing values”. In: *Systems & Control Lett.* 95 (2016), pp. 53–61.

[11] I. Markovsky. “The Behavioral Toolbox”. In: *Proc. of Machine Learning Research*. Vol. 242. 2024, pp. 130–141.

[12] I. Markovsky and F. Dörfler. “Data-driven dynamic interpolation and approximation”. In: *Automatica* 135 (2022), p. 110008.

[13] I. Markovsky and F. Dörfler. “Identifiability in the behavioral setting”. In: *IEEE Trans. Automat. Contr.* 68 (3 2023), pp. 1667–1677.

[14] I. Markovsky, L. Huang, and F. Dörfler. “Data-driven control based on behavioral approach: From theory to applications in power systems”. In: *IEEE Control Systems Magazine* 43 (5 2023), pp. 28–68.

[15] I. Markovsky and P. Rapisarda. “Data-driven simulation and control”. In: *Int. J. Control* 81.12 (2008), pp. 1946–1959.

[16] I. Markovsky and K. Usevich. “Software for weighted structured low-rank approximation”. In: *J. Comp. Appl. Math.* 256 (2014), pp. 278–292.

[17] V. Mishra, I. Markovsky, and B. Grossmann. “Data-Driven Tests for Controllability”. In: *Control Systems Letters* 5 (2 2020), pp. 517–522.

[18] J. Polderman and J. C. Willems. *Introduction to Mathematical Systems Theory*. Springer-Verlag, 1998.

[19] H. van Waarde, C. De Persis, M. K. Camlibel, and P. Tesi. “Willems’ Fundamental Lemma for State-Space Systems and Its Extension to Multiple Datasets”. In: *IEEE Control Systems Letters* 4 (2020), pp. 602–607.

[20] H. van Waarde, J. Eising, H. Trentelman, and K. Camlibel. “Data informativity: A new perspective on data-driven analysis and control”. In: *IEEE Trans. Automat. Contr.* 65 (11 2020), pp. 4753–4768.

[21] J. C. Willems. “From time series to linear system—Part I. Finite dimensional linear time invariant systems”. In: *Automatica* 22.5 (1986), pp. 561–580.

[22] J. C. Willems. “Paradigms and puzzles in the theory of dynamical systems”. In: *IEEE Trans. Automat. Contr.* 36 (1991), pp. 259–294.

[23] J. C. Willems. “System theoretic models for the analysis of physical systems”. In: *Ricerche di Automatica* 10.2 (1979), pp. 71–106.

[24] J. C. Willems. “The behavioral approach to open and interconnected systems: Modeling by tearing, zooming, and linking”. In: *IEEE Control Systems Magazine* 27 (2007), pp. 46–99.

[25] J. C. Willems, P. Rapisarda, I. Markovsky, and B. De Moor. “A note on persistency of excitation”. In: *Systems & Control Lett.* 54.4 (2005), pp. 325–329.