Chapter 8
Exact Identification

With this chapter, we start to consider identification problems. The first problem is the simplest of this type: given a trajectory of an LTI system, find a representation of the system that produced this trajectory. The problem is defined and motivated in Sections 8.1–8.3.

Exact identification is closely related to the construction of the most powerful unfalsified model (MPUM). In Section 8.2, we define the MPUM, and in Section 8.3, we define the identifiability property. Under identifiability, the MPUM of the data, which is explicitly constructible from the data, coincides with the data generating system. This allows us to find the data generating system from data. An identifiability test in terms of the given data is presented in Section 8.4. This key result is repeatedly used in what follows and is called the fundamental lemma.

In Section 8.5, we review algorithms for exact identification. Section 8.6 presents algorithms for passing from data to a convolution representation. Section 8.7 reviews realization theory and algorithms. Section 8.8 presents algorithms for computation of sequential free responses, which are a key ingredient of direct algorithms for the construction of an input/state/output representation of the MPUM.

In Section 8.9, we explain the relation of the algorithms presented to classical algorithms for deterministic subspace identification. In particular, the orthogonal and oblique projections correspond to computation of, respectively, free responses and sequential free responses of the system. We comment on the inherent inefficiency of the orthogonal and oblique projections for the purpose of exact identification. Simulation results that compare the efficiency of various exact identification algorithms are shown in Section 8.10.

8.1 Introduction

In this chapter, we consider the following problem:

Given a trajectory w_{d} of an LTI system $\mathscr{B}$, find a representation of $\mathscr{B}$.

We refer to this most basic identification problem as an exact identification problem. It is of interest to find algorithms that make the transition from w_{d} directly to any one of the various possible representations of $\mathscr{B}$; cf. Figure 7.1.

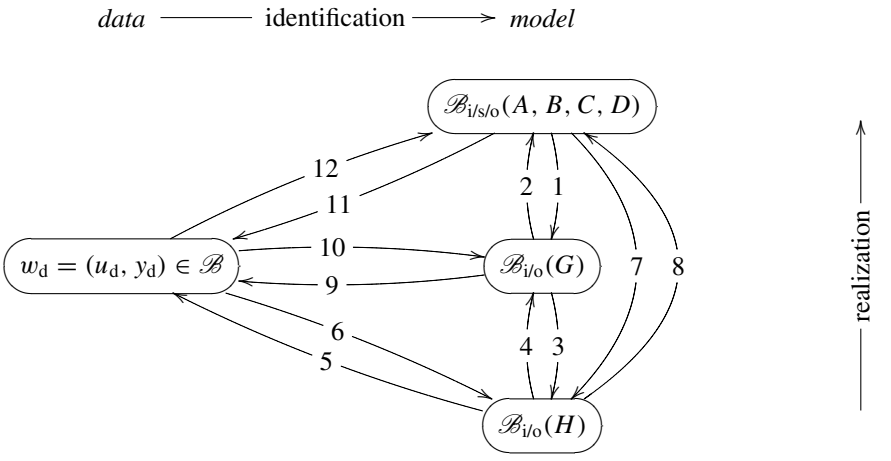

Figure 8.1. *Data, input/output model representations, and links among them.*

1. $G(z) = C(Iz - A)^{-1}B + D$
2. Realization of a transfer function
3. $H = \mathscr{Z}^{-1}(G)$
4. $G = \mathscr{Z}(H) = \sum_{t=0}^{\infty} H(t)z^{-t}$
5. Convolution $y_{\mathrm{d}}(t) = \sum_{\tau=0}^{t} H(\tau)u_{\mathrm{d}}(t - \tau)$
6. Exact identification; see Algorithms 8.6 and 8.7
7. $H(0) = D$, $H(t) = CA^{t-1}B$, for $t \geq 1$
8. Realization of an impulse response; see Algorithm 8.8
9. Simulation of the response under the input u_{d}
10. Exact identification; see Algorithm 8.1
11. Simulation of the response under the input u_{d} and initial conditions $x(1) = x_{\mathrm{ini}}$
12. Exact identification; see Algorithms 8.4 and 8.5

Figure 8.1 shows the representations with an input/output partition of the variables that we considered before and the trajectory $w_{\mathrm{d}} =: (u_{\mathrm{d}}, y_{\mathrm{d}})$. The transitions from w_{d} to convolution, transfer function, and input/state/output representations are exact identification problems. The transitions among the representations themselves are representation problems. Most notable of the representation problems are the realization ones: passing from an impulse response or transfer function to an input/state/output representation.

The exact identification problem is an important system theoretic problem. It includes as a special case the classical impulse response realization problem and is a prerequisite for the study of more involved approximate, stochastic, and stochastic/approximate identification problems (e.g., the GlTLS misfit minimization problem, which is an approximate identification problem). In addition, numerical algorithms for exact identification are useful

computational tools and appear as subproblems in other identification algorithms. By itself, however, exact identification is not a practical identification problem. The data is assumed to be exact and unless $\mathscr{B}$ is the trivial system $\mathscr{B} = (\mathbb{R}^{\mathrm{w}})^{\mathbb{N}}$, a randomly chosen time series $w_{\mathrm{d}} \in (\mathbb{R}^{\mathrm{w}})^{\mathbb{N}}$ is a trajectory of $\mathscr{B}$ with probability zero.

Modified exact identification algorithms can be applied on data that is not necessarily generated by a finite dimensional LTI system by replacing exact linear algebra operations with approximate operations. For example, rank determination is replaced by numerical rank determination (via SVD) and solution of a system of linear equations by LS or TLS approximation. A lot of research is devoted to the problem of establishing alternatives to $w_{\mathrm{d}} \in \mathscr{B}$, under which such modified algorithms have desirable properties. Often this problem is treated in the stochastic setting of the ARMAX model and the properties aimed at are consistency and asymptotic efficiency.

Note 8.1 (Multiple time series) In general, the given data for identification is a finite set of time series $w_{\mathrm{d},1}, \ldots, w_{\mathrm{d},N}$. In the presentation, however, we define and solve the identification problems for a single time series. The generalization for multiple time series of equal length is trivial and the one for nonequal length is an open problem.

Note 8.2 (Finite amount of data) An important aspect of the identification problems that we address is the finiteness of the available data. Previous studies of exact identification either assume an infinite amount of data or do not address the issue of finiteness of the data.

Note 8.3 (Given input/output partitioning) Although the exact identification problem is defined in the behavioral setting, most of the established results are in the input/output setting. In our treatment, some problems are also solved in the input/output setting.

Software implementation of the algorithms presented in this and the following chapter is described in Appendix B.3.

8.2 The Most Powerful Unfalsified Model

The notion of the most powerful unfalsified model (MPUM) is introduced in [Wil86b, Definition 4]. It plays a fundamental role in the exact identification problem.

Definition 8.4 (MPUM in the model class $\mathscr{L}^{\mathrm{w}}$ [Wil86b]). *The system $\mathscr{B} \subseteq (\mathbb{R}^{\mathrm{w}})^{\mathbb{N}}$ is an MPUM of the time series $w_{\mathrm{d}} \in (\mathbb{R}^{\mathrm{w}})^{T}$ in the model class $\mathscr{L}^{\mathrm{w}}$ if it is*

1. *finite dimensional LTI, i.e., $\mathscr{B} \in \mathscr{L}^{\mathrm{w}}$,*
2. *unfalsified, i.e., $w_{\mathrm{d}} \in \mathscr{B}|_{[1,T]}$, and*
3. *most powerful among all finite dimensional LTI unfalsified systems, i.e.,*

$$\mathscr{B}' \in \mathscr{L}^{\mathrm{w}} \text{ and } w_{\mathrm{d}} \in \mathscr{B}'|_{[1,T]} \quad \Longrightarrow \quad \mathscr{B}|_{[1,T]} \subseteq \mathscr{B}'|_{[1,T]}.$$

The MPUM of w_{d} is denoted by $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$. We skip the explicit dependence on w_{d} when w_{d} is understood from the context.

The existence and uniqueness of the MPUM are proven in the following theorem.

Theorem 8.5 (Existence and uniqueness of the MPUM [Wil86b]). *The MPUM of* $w_{\mathrm{d}} \in (\mathbb{R}^{\mathrm{w}})^T$ *exists and is unique. Moreover,*

$$\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}}) = \cap_{\substack{w_{\mathrm{d}} \in \mathscr{B}|_{[1,T]} \\ \mathscr{B} \in \mathscr{L}^{\mathrm{w}}}} \mathscr{B};$$

i.e., $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ *is the smallest shift-invariant closed in the topology of pointwise convergence subspace of* $(\mathbb{R}^{\mathrm{w}})^{\mathbb{N}}$ *that contains* w_{d}.

Proof. Define $\mathscr{B}' := \cap_{\substack{w_{\mathrm{d}} \in \mathscr{B}|_{[1,T]} \\ \mathscr{B} \in \mathscr{L}^{\mathrm{w}}}} \mathscr{B}$. We will show that $\mathscr{B}'$ is an MPUM.

Lemma 8.6 (Intersection property of $\mathscr{L}^{\mathrm{w}}$). $\mathscr{B}_1, \mathscr{B}_2 \in \mathscr{L}^{\mathrm{w}} \implies \mathscr{B}_1 \cap \mathscr{B}_2 \in \mathscr{L}^{\mathrm{w}}$.

Proof. See [Wil86b, Proposition 11]. ∎

Lemma 8.6 implies that $\mathscr{B}' \in \mathscr{L}^{\mathrm{w}}$. Obviously, $w_{\mathrm{d}} \in \mathscr{B}'$, so that $\mathscr{B}'$ is unfalsified. Moreover, $\mathscr{B}'$ is in the intersection of all finite dimensional LTI unfalsified models, so that it is most powerful. Therefore, $\mathscr{B}'$ is an MPUM.

We proved the existence of an MPUM. In order to prove uniqueness, assume that there is $\mathscr{B}'' \neq \mathscr{B}'$ that is also an MPUM of w_{d}. By Lemma 8.6, $\mathscr{B} := \mathscr{B}'' \cap \mathscr{B}' \in \mathscr{L}^{\mathrm{w}}$ and $\mathscr{B}$ is obviously unfalsified. But $\mathscr{B} \subset \mathscr{B}'$, so that $\mathscr{B}'$ is not an MPUM, which is a contradiction. ∎

The next proposition shows another characterization of the MPUM for infinite w_{d}.

Proposition 8.7. *Let* $w_{\mathrm{d}} \in (\mathbb{R}^{\mathrm{w}})^{\mathbb{N}}$. *Then*

$$\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}}) = \mathrm{closure}\big(\,\mathrm{image}(w_{\mathrm{d}}, \sigma w_{\mathrm{d}}, \sigma^2 w_{\mathrm{d}}, \ldots)\big);$$

i.e., $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ *is the closure of the span of* w_{d} *and all its shifts.*

Proof. Let $\mathscr{B}' := \mathrm{closure}\big(\,\mathrm{image}(w_{\mathrm{d}}, \sigma w_{\mathrm{d}}, \sigma^2 w_{\mathrm{d}}, \ldots)\big)$. By definition, $\mathscr{B}'$ is a closed, linear, and shift-invariant subspace. Then [Wil86a, Theorem 5] implies that $\mathscr{B}' \in \mathscr{L}^{\mathrm{w}}$. By definition, $w_{\mathrm{d}} \in \mathscr{B}'$, so that $\mathscr{B}'$ is unfalsified. From conditions 1 and 2 of Definition 8.4, it is easy to see that any unfalsified model contains $\mathscr{B}'$. Therefore, $\mathscr{B}'$ is the MPUM of w_{d}. ∎

Note 8.8 (Algorithms for construction of the MPUM) Proposition 8.7 shows that the MPUM $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ is explicitly constructible from the given data w_{d}. However, algorithms that pass from w_{d} to concrete representations of $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ are needed. Such algorithms are described in Section 8.5.

Note 8.9 (Generically $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}}) = (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ for infinite data $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$) The existence of the MPUM is guaranteed in the model class $\mathscr{L}^{\mathtt{w}}$ of unbounded complexity. For "rough" data $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ (the generic case in $(\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$), the MPUM is the trivial system $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}}) = (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$, i.e., a system with $\mathtt{w}$ inputs. Therefore, generically the MPUM of an infinite time series does not exist in a model class $\mathscr{L}^{\mathtt{w}}_{\mathtt{m}}$ with $\mathtt{m} < \mathtt{w}$. Therefore, an approximation is needed in order to find a nontrivial model. Approximate identification is treated in Chapter 11.

Note 8.10 (Generically $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})|_{[1,T]} = (\mathbb{R}^{\mathtt{w}})^{T}$ for finite data $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{T}$) For finite data $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{T}$, the MPUM always exists in a model class $\mathscr{L}^{\mathtt{w}}_{\mathtt{m}}$ with any number $0 \leq \mathtt{m} \leq \mathtt{w}$ of inputs. For rough data the solution is still a trivial system $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})|_{[1,T]} = (\mathbb{R}^{\mathtt{w}})^{T}$. Now, however, the possibility of fitting an arbitrary T samples long time series is achieved by the initial conditions as well as the inputs. Indeed, any observable system $\mathscr{B} \in \mathscr{L}^{\mathtt{w}}$ of order $\mathbf{n}(\mathscr{B}) \geq \mathbf{p}(\mathscr{B})T$ is unfalsified by any T samples long time series $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{T}$.

8.3 Identifiability

Not every trajectory w_{d} of a system $\mathscr{B} \in \mathscr{L}^{\mathtt{w}}$ allows the reconstruction of $\mathscr{B}$ from w_{d}. For example, the trajectory $w_{\mathrm{d}} = 0 \in \mathscr{B}$ does not carry any information about $\mathscr{B}$ because any LTI system is compatible with the zero trajectory. The possibility of identifying $\mathscr{B}$ from w_{d} is a property of both w_{d} and $\mathscr{B}$. In order to formalize the notion of the "possibility of identifying a system from exact data", we define the identifiability property as follows.

Definition 8.11 (Identifiability). *The system $\mathscr{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ is identifiable from the data $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{T}$ in the model class $\mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$ if*

1. $\mathscr{B} \in \mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$,
2. $w_{\mathrm{d}} \in \mathscr{B}|_{[1,T]}$, *and*
3. *there is no other system $\mathscr{B}' \in \mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$, $\mathscr{B}' \neq \mathscr{B}$, that fits the data, i.e.,*

$$\mathscr{B}' \in \mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}} \quad \text{and} \quad w_{\mathrm{d}} \in \mathscr{B}'|_{[1,T]} \quad \Longrightarrow \quad \mathscr{B}' = \mathscr{B}.$$

Identifiability in $\mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$ implies that the MPUM of the data w_{d} is in $\mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$ and coincides with the data generating system $\mathscr{B}$.

Theorem 8.12. *If $\mathscr{B} \subseteq (\mathbb{R}^{\mathtt{w}})^{\mathbb{N}}$ is identifiable in the model class $\mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$ from the data $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^{T}$ in the model class $\mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$, then $\mathscr{B} = \mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$.*

Proof. The first condition for $\mathscr{B}$ being identifiable from w_{d} implies the first condition for $\mathscr{B}$ being the MPUM of w_{d}, and the second conditions are equivalent. Condition 3 for $\mathscr{B}$ being identifiable from w_{d} implies that there is a unique unfalsified system in the model class $\mathscr{L}^{\mathtt{w},\mathtt{n}}_{\mathtt{m},\mathtt{l}}$. Therefore, $\mathscr{B}$ is the MPUM of w_{d}. □

Since the MPUM is explicitly computable from the given data (see Note 8.8), identifiability indeed implies the "possibility of identifying the system from exact data". In Section 8.5, we list algorithms for passing from w_{d} to kernel, convolution, and input/state/output

representations of the MPUM. For example, consider Algorithm 8.1, which constructs a kernel representation of the MPUM $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$.

Next, we define the considered exact identification problem.

Problem 8.13 (Exact identification). *Given $w_{\mathrm{d}} \in \mathscr{B} \in \mathscr{L}^{\mathtt{w}}$ and complexity specification $(\mathtt{m}, \mathtt{l}_{\max}, \mathtt{n}_{\max})$, determine whether $\mathscr{B}$ is identifiable from w_{d} in the model class $\mathscr{L}^{\mathtt{w},\mathtt{n}_{\max}}_{\mathtt{m},\mathtt{l}_{\max}}$, and if so, find an algorithm that computes a representation of $\mathscr{B}$.*

8.4 Conditions for Identifiability

The block-Hankel matrix with t_1 block rows and t_2 block columns, constructed from (in general matrix valued) time series $w = \big(w(1), w(2), \dots\big)$, is denoted by

$$\mathscr{H}_{t_1,t_2}(w) := \begin{bmatrix} w(1) & w(2) & w(3) & \cdots & w(t_2) \\ w(2) & w(3) & w(4) & \cdots & w(t_2+1) \\ w(3) & w(4) & w(5) & \cdots & w(t_2+2) \\ \vdots & \vdots & \vdots & & \vdots \\ w(t_1) & w(t_1+1) & w(t_1+2) & \cdots & w(t_1+t_2-1) \end{bmatrix}. \qquad (\mathscr{H})$$

If both block dimensions t_1 and t_2 are infinite, we skip them in the notation; i.e., we define $\mathscr{H}(w) := \mathscr{H}_{\infty,\infty}(w)$. If the time series is finite, i.e., $w = \big(w(1), \dots, w(T)\big)$, then $\mathscr{H}_{t_1}(w)$ denotes the Hankel matrix with t_1 block rows and as many block columns as the finite time horizon T allows; i.e., $\mathscr{H}_{t_1}(w) := \mathscr{H}_{t_1,t_2}(w)$, where $t_2 = T - t_1 + 1$.

With some abuse of notation (w is viewed as both the matrix $\begin{bmatrix} w(1) & w(2) & \cdots \end{bmatrix}$ and the vector $\operatorname{col}\big(w(1), w(2), \dots\big)$), the infinite Hankel matrix $\mathscr{H}(w)$ can be block partitioned in the following two ways:

$$\mathscr{H}(w) = \begin{bmatrix} w \\ \sigma w \\ \sigma^2 w \\ \vdots \end{bmatrix} = \begin{bmatrix} w & \sigma w & \sigma^2 w & \cdots \end{bmatrix},$$

which shows that it is composed of w and its shifts $\sigma^t w$, $t \geq 1$, stacked next to each other. Therefore, $w \in \mathscr{B}$ implies that $\operatorname{col\,span}\big(\mathscr{H}(w)\big) \subseteq \mathscr{B}$. We establish conditions on w and $\mathscr{B}$ under which equality holds, i.e., conditions under which w specifies $\mathscr{B}$ exactly.

Definition 8.14 (Persistency of excitation). *The time series $u_{\mathrm{d}} = \big(u_{\mathrm{d}}(1), \dots, u_{\mathrm{d}}(T)\big)$ is persistently exciting of order L if the Hankel matrix $\mathscr{H}_L(u_{\mathrm{d}})$ is of full row rank.*

Lemma 8.15 (Fundamental lemma [WRMM05]). *Let*

1. *$w_{\mathrm{d}} = (u_{\mathrm{d}}, y_{\mathrm{d}})$ be a T samples long trajectory of the LTI system $\mathscr{B}$, i.e.,*

$$w_{\mathrm{d}} = \begin{bmatrix} u_{\mathrm{d}} \\ y_{\mathrm{d}} \end{bmatrix} = \left(\begin{bmatrix} u_{\mathrm{d}}(1) \\ y_{\mathrm{d}}(1) \end{bmatrix}, \dots, \begin{bmatrix} u_{\mathrm{d}}(T) \\ y_{\mathrm{d}}(T) \end{bmatrix} \right) \in \mathscr{B}|_{[1,T]};$$

2. *the system $\mathscr{B}$ be controllable; and*

3. *the input sequence u_d be persistently exciting of order $L + \mathbf{n}(\mathscr{B})$.*

Then any L samples long trajectory $w = (u, y)$ of $\mathscr{B}$ can be written as a linear combination of the columns of $\mathscr{H}_L(w_\mathrm{d})$ and any linear combination $\mathscr{H}_L(w_\mathrm{d})g$, $g \in \mathbb{R}^{T-L+1}$, is a trajectory of $\mathscr{B}$, i.e.,

$$\operatorname{col\,span}\left(\mathscr{H}_L(w_\mathrm{d})\right) = \mathscr{B}|_{[1,L]}.$$

Proof. See Appendix A.3. □

The fundamental lemma gives conditions under which the Hankel matrix $\mathscr{H}_L(w_\mathrm{d})$ has the "correct" image (and as a consequence the "correct" left kernel). For sufficiently large L, namely $L \geq \mathbf{l}(\mathscr{B}) + 1$, it answers the identifiability question.

Theorem 8.16 (Identifiability conditions). *The system $\mathscr{B} \in \mathscr{L}^\mathtt{w}$ is identifiable from the exact data $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d}) \in \mathscr{B}$ if $\mathscr{B}$ is controllable and u_d is persistently exciting of order $\mathbf{l}(\mathscr{B}) + 1 + \mathbf{n}(\mathscr{B})$.*

Note that for applying Theorem 8.16, we need to know a priori the order and the lag of $\mathscr{B}$ and that $\mathscr{B}$ is controllable. These assumptions can be relaxed as follows. Knowledge of upper bounds $\mathtt{n}_{\max}$ and $\mathtt{l}_{\max}$ of, respectively, $\mathbf{n}(\mathscr{B})$ and $\mathbf{l}(\mathscr{B})$ suffice to verify identifiability. Moreover, the condition "$\mathscr{B}$ controllable and u_d persistently exciting of order $\mathtt{l}_{\max} + 1 + \mathtt{n}_{\max}$" is the sharpest necessary condition for identifiability that is verifiable from the data, $\mathtt{n}_{\max}$, and $\mathtt{l}_{\max}$ only. In other words, if u_d is not persistently exciting of order $\mathtt{l}_{\max}+1+\mathtt{n}_{\max}$, then there is a controllable system $\mathscr{B} \in \mathscr{L}^{\mathtt{w},\mathtt{n}_{\max}}_{\mathtt{m},\mathtt{l}_{\max}}$, such that $w_\mathrm{d} \in \mathscr{B}$ and $\mathscr{B}$ is not identifiable from w_d.

We will need the following corollary of the fundamental lemma.

Corollary 8.17 (Willems et al. [WRMM05]). *Consider the minimal input/state/output representation of the controllable system $\mathscr{B}$, $\mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$, and let x_d be the state sequence of $\mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$, corresponding to the trajectory $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d})$ of $\mathscr{B}$.*

(i) *If u_d is persistently exciting of order $\mathbf{n}(\mathscr{B}) + 1$, then*

$$\operatorname{rank}\left(\begin{bmatrix} x_\mathrm{d}(1) & x_\mathrm{d}(2) & \cdots & x_\mathrm{d}(T) \end{bmatrix}\right) = \mathbf{n}(\mathscr{B})$$

and

$$\operatorname{rank}\begin{bmatrix} u_\mathrm{d}(1) & \cdots & u_\mathrm{d}(T) \\ x_\mathrm{d}(1) & \cdots & x_\mathrm{d}(T) \end{bmatrix} = \mathbf{n}(\mathscr{B}) + \mathtt{m}.$$

(ii) *If u_d is persistently exciting of order $\mathbf{n}(\mathscr{B}) + L$, then*

$$\operatorname{rank}\begin{bmatrix} X_\mathrm{d} \\ \mathscr{H}_L(u_\mathrm{d}) \end{bmatrix} = \mathbf{n}(\mathscr{B}) + L\mathtt{m}, \quad \textit{where} \quad X_\mathrm{d} := \begin{bmatrix} x_\mathrm{d}(1) & \cdots & x_\mathrm{d}(T-L+1) \end{bmatrix}.$$

The rest of the chapter is devoted to the second part of the exact identification problem: algorithms that compute a representation of the MPUM.

8.5 Algorithms for Exact Identification

If the conditions of Theorem 8.16 are satisfied, then there are algorithms that compute a representation of the data generating system $\mathscr{B}$ from the data w_{d}. In fact, such algorithms compute the MPUM of the data w_{d}. In this section, we outline four classes of algorithms for exact identification. The first one derives a kernel representation and the second one derives a convolution representation. Composed with realization algorithms, they give (indirect) algorithms for the computation of state space representations. The last two classes of algorithms construct (directly) an input/state/output representation.

Algorithms for Computation of a Kernel Representation

Under the assumption of the fundamental lemma,

$$\ker\left(\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}})\right) = \mathscr{B}|_{[0,\,\mathtt{l}_{\max}]}.$$

Therefore, a basis for the left kernel of $\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}})$ defines a kernel representation of $\mathscr{B} \in \mathscr{L}^{\mathtt{w},\mathtt{n}_{\max}}_{\mathtt{m},\mathtt{l}_{\max}}$. Let

$$\begin{bmatrix}\tilde{R}_0 & \tilde{R}_1 & \cdots & \tilde{R}_{\mathtt{l}_{\max}}\end{bmatrix}\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}}) = 0,$$

where $\tilde{R}_i \in \mathbb{R}^{g\times \mathtt{w}}$ with $g = \mathtt{p}(\mathtt{l}_{\max}+1) - \mathbf{n}(\mathscr{B})$. Then

$$\mathscr{B} = \ker\left(\tilde{R}(\sigma)\right), \quad \text{where} \quad \tilde{R}(z) = \sum_{i=0}^{\mathtt{l}_{\max}} \tilde{R}_i z^i.$$

This (in general nonminimal) kernel representation can be made minimal by standard polynomial linear algebra algorithms: find a unimodular matrix $\tilde{U} \in \mathbb{R}^{g\times g}[z]$, such that $\tilde{U}\tilde{R} = \left[\begin{smallmatrix} R \\ 0 \end{smallmatrix}\right]$, where R is full row rank. Then $\ker\left(R(\sigma)\right) = 0$ is a minimal kernel representation of $\mathscr{B}$.

The above procedure is summarized in Algorithm 8.1.

Note 8.18 (Approximate identification) The SVD in step 2 of Algorithm 8.1 is used for the computation of the left kernel of the block-Hankel matrix $\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}})$. Other algorithms can be used for the same purpose as well. The SVD, however, has an important advantage when an approximate model is desired.

Suppose that $\operatorname{rank}\left(\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}})\right) = \mathtt{w}(\mathtt{l}_{\max}+1)$, so that $\mathscr{B}_{\mathrm{mpum}}$ is the trivial model $(\mathbb{R}^{\mathtt{w}})^T$. Nevertheless, one can proceed heuristically with steps 5 and 6 in order to compute a nontrivial approximate model. The parameter g can either be chosen from the decay of the singular values (e.g., the number of singular values smaller than a user-given tolerance) or be fixed. The selection of g determines the number of inputs of the identified model and thus its complexity. The motivation for this heuristic for approximate modeling is that U_2 spans a space that in a certain sense is an "approximate left kernel" of $\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}})$.

In [Wil86b, Section 15], Algorithm 8.1 is refined. An efficient recursive algorithm for the computation of a kernel representation of the MPUM is proposed. Moreover, the algorithm of [Wil86b] computes a shortest lag kernel representation and as a byproduct finds an input/output partition of the variables.

Algorithm 8.1 Kernel representation of the MPUM w2r

Input: $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^T$ and $\mathtt{l}_{\max}$.

1: Compute the SVD of $\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}}) = U\Sigma V^\top$ and let r be the rank of $\mathscr{H}_{\mathtt{l}_{\max}+1}(w_{\mathrm{d}})$.
2: **if** $r = \mathtt{w}(\mathtt{l}_{\max}+1)$ **then**
3: $R(z) = 0_{1\times \mathtt{w}}$ {the MPUM is the trivial model $(\mathbb{R}^{\mathtt{w}})^T$}.
4: **else**
5: Let $U := \begin{bmatrix} \overset{r}{U_1} & \overset{g}{U_2} \end{bmatrix}$ and define $U_2^\top =: \begin{bmatrix} \tilde{R}_0 & \tilde{R}_1 & \cdots & \tilde{R}_{\mathtt{l}_{\max}} \end{bmatrix}$, where $\tilde{R}_i \in \mathbb{R}^{g\times \mathtt{w}}$.
6: Compute a unimodular matrix $\tilde{U} \in \mathbb{R}^{g\times g}[z]$, such that

$$\tilde{U}(z)\left(\sum_{i=0}^{\mathtt{l}_{\max}} \tilde{R}_i z^i\right) = \begin{bmatrix} R(z) \\ 0 \end{bmatrix}, \qquad \text{where } R \text{ is full row rank.}$$

7: **end if**

Output: $R(z)$—a minimal kernel representation of the MPUM.

Algorithm 8.2 is an indirect algorithm for computation of an input/state/output representation of the MPUM that uses Algorithm 8.1 for computing a kernel representation first. The transition from a kernel representation to an input/state/output representation is a standard one. First, a maximal-degree, full-rank submatrix $P \in \mathbb{R}^{\mathtt{p}\times\mathtt{p}}$ of R is selected and Q is defined as the complementary to P submatrix of R. Then the left matrix fraction description (P, Q) is realized by standard realization algorithms.

Algorithm 8.2 I/S/O representation of the MPUM via a kernel representation w2r2ss

Input: $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{w}})^T$ and $\mathtt{l}_{\max}$.

1: Compute a minimal kernel representation of the MPUM via Algorithm 8.1.
2: Select a maximal-degree, full-rank submatrix $P \in \mathbb{R}^{\mathtt{p}\times\mathtt{p}}$ of R and let Q be the complementary to P submatrix of R {select an input/output partition of the variables}.
3: Realize (P, Q) via a state space system $\mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$.

Output: (A, B, C, D)—a minimal input/state/output representation of the MPUM.

If an input/output partition of the time series w_{d} is a priori given, then step 2 is skipped. For the computation of the transfer function $P^{-1}(z)Q(z)$ of $\mathscr{B}$, matrix polynomial linear operations are needed that are not an integral part of most popular numerical linear algebra packages and libraries such as MATLAB.

Algorithms for Computation of a Convolution Representation

The convolution representation is parameterized by the impulse response. Algorithm 8.7 from Section 8.6 computes the impulse response directly from data. This algorithm is a consequence of the fundamental lemma with the refinement that iteratively sequential pieces of the impulse response are computed.

The impulse response is used in the algorithms for balanced model identification, presented in Chapter 9. Previously proposed algorithms for balanced model identification

compute a Hankel matrix of the Markov parameters and thus recompute most samples of the impulse response many times. The algorithm presented in Section 8.6 avoids this and as a result is more efficient.

Algorithm 8.3 is an indirect algorithm for computation of an input/state/output representation of the MPUM that uses Algorithm 8.7 for computing a convolution representation first. The transition from a convolution representation to an input/state/output representation is a standard problem of realization theory; see Section 8.7.

Algorithm 8.3 I/S/O representation of the MPUM via an impulse response `uy2h2ss`

Input: u_d, y_d, $\mathtt{n}_{\max}$, and $\mathtt{l}_{\max}$.

1: Compute the first $\mathtt{l}_{\max}+1+\mathtt{n}_{\max}$ samples of the impulse response H of the MPUM via Algorithm 8.7.

2: Compute a realization $\mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$ of H via Algorithm 8.8.

Output: (A, B, C, D)—a minimal input/state/output representation of the MPUM.

Algorithms Based on Computation of an Observability Matrix

Let $\mathscr{B} = \mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$. If, in addition to $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d})$, the extended observability matrix $\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)$ were known, we could find (A, B, C, D) by solving two linear systems of equations. The first block row of $\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)$ immediately gives C, and A is computed from the so-called shift equation

$$\big(\sigma^*\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)\big)A = \big(\sigma\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)\big).$$

(σ and σ^*, acting on a block matrix, remove, respectively, the first and the last block rows.) Once A and C are known, computing D, B, and the initial condition x_ini, under which w_d is obtained, is also a linear problem. The system of equations (see (VC))

$$y_\mathrm{d}(t) = CA^t x_\mathrm{ini} + \sum_{\tau=1}^{t-1} CA^{t-1-\tau}Bu_\mathrm{d}(\tau) + D\delta(t+1), \quad \text{for } t = 1, \ldots, \mathtt{l}_{\max}+1, \quad (8.1)$$

is linear in the unknowns D, B, and x_ini and can be solved explicitly by using Kronecker products.

Thus the identification problem boils down to the computation of $\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)$. Observe that the columns of $\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)$ are $\mathbf{n}(\mathscr{B})$ linearly independent free responses of $\mathscr{B}$. Moreover, *any* $\mathbf{n}(\mathscr{B})$ linearly independent free responses $y_1, \ldots, y_{\mathbf{n}(\mathscr{B})}$ of $\mathscr{B}$, stacked next to each other, determine the extended observability matrix up to a similarity transformation. Let $x_1, \ldots, x_{\mathbf{n}(\mathscr{B})}$ be the initial conditions for $y_1, \ldots, y_{\mathbf{n}(\mathscr{B})}$. The matrix

$$X_\mathrm{ini} := \begin{bmatrix} x_1 & \cdots & x_{\mathbf{n}(\mathscr{B})} \end{bmatrix} \in \mathbb{R}^{\mathbf{n}(\mathscr{B})\times\mathbf{n}(\mathscr{B})}$$

is full rank because, by assumption, the corresponding responses are linearly independent. Then

$$Y_0 := \begin{bmatrix} y_1 & \cdots & y_{\mathbf{n}(\mathscr{B})} \end{bmatrix} = \mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)X_\mathrm{ini},$$

which shows that Y_0 is equivalent to $\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)$.

We have further reduced the identification problem to the problem of computing $\mathbf{n}(\mathscr{B})$ linearly independent free responses of the MPUM. Under the assumptions of the fundamental lemma, such responses can be computed in the same way as the one used for the computation of the impulse response directly from data. The details are described in Section 8.8.

Since $\mathbf{n}(\mathscr{B})$ is unknown, however, $\mathtt{n}_{\max}$ free responses $y_1, \ldots, y_{\mathtt{n}_{\max}}$ are computed such that the corresponding matrix $Y_0 := \begin{bmatrix} y_1 & \cdots & y_{\mathtt{n}_{\max}} \end{bmatrix}$ has its maximal possible rank $\mathbf{n}(\mathscr{B})$. The matrix Y_0 in this case can be viewed as an extended observability matrix $\mathscr{O}_{\mathtt{l}_{\max}+1}(\tilde{A}, \tilde{C})$ for a nonminimal input/state/output representation of $\mathscr{B}$ with $\tilde{A} \in \mathbb{R}^{\mathtt{n}_{\max} \times \mathtt{n}_{\max}}$ and $\tilde{C} \in \mathbb{R}^{\mathtt{p} \times \mathtt{n}_{\max}}$. In order to find a minimal representation, a rank revealing factorization $Y_0 = \Gamma X_{\mathrm{ini}}$ of Y_0 is computed. The matrix Γ is equal to $\mathscr{O}_{\mathtt{l}_{\max}+1}(A, C)$ up to a similarity transformation. The nonuniqueness of the state space basis in which Γ and X_{ini} are obtained corresponds precisely to the nonuniqueness of the rank revealing factorization.

The procedure outlined above is summarized in Algorithm 8.4.

Algorithm 8.4 I/S/O representation of the MPUM via an observability matrix `uy2o2ss`

Input: u_{d}, y_{d}, $\mathtt{l}_{\max}$, and $\mathtt{n}_{\max}$.

1: Compute $\mathtt{n}_{\max}$, $\mathtt{l}_{\max}+1$ samples long free responses Y_0 of the MPUM via Algorithm 8.9.
2: Compute a rank revealing factorization $Y_0 = \Gamma X_{\mathrm{ini}}$.
3: Solve the linear system of equations $(\sigma^*\Gamma)A = (\sigma\Gamma)$ for A and define C to be the first block entry of Γ.
4: Solve the linear system of equations (8.1) for D, B, and x_{ini}.

Output: (A, B, C, D)—a minimal input/state/output representation of the MPUM.

Algorithms Based on Computation of a State Sequence

If a state sequence $x_{\mathrm{d}}(1), \ldots, x_{\mathrm{d}}(\mathbf{n}(\mathscr{B}) + \mathtt{m} + 1)$ of an input/state/output representation of the MPUM were known, then the parameters (A, B, C, D) could be computed by solving the linear system of equations

$$\begin{bmatrix} x_{\mathrm{d}}(2) & \cdots & x_{\mathrm{d}}(\mathbf{n}(\mathscr{B}) + \mathtt{m} + 1) \\ y_{\mathrm{d}}(1) & \cdots & y_{\mathrm{d}}(\mathbf{n}(\mathscr{B}) + \mathtt{m}) \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_{\mathrm{d}}(1) & \cdots & x_{\mathrm{d}}(\mathbf{n}(\mathscr{B}) + \mathtt{m}) \\ u_{\mathrm{d}}(1) & \cdots & u_{\mathrm{d}}(\mathbf{n}(\mathscr{B}) + \mathtt{m}) \end{bmatrix}. \tag{8.2}$$

Therefore, the identification problem is reduced to the problem of computing a state sequence of the MPUM. This can be done by computing $\mathbf{n}(\mathscr{B}) + \mathtt{m} + 1$ *sequential* free responses. By "sequential" we mean that the corresponding sequence of initial conditions for the responses is a valid state sequence. Under the conditions of the fundamental lemma, such responses can be computed from data by an algorithm similar to the ones used for the computation of the impulse response and free responses. Since $\mathbf{n}(\mathscr{B})$ is unknown, however, $\mathtt{n}_{\max} + \mathtt{m} + 1$ sequential free responses should be computed. The details are described in Section 8.8.

The procedure outlined above is summarized in Algorithm 8.5. An alternative approach for computing a state sequence directly from data, based on the shift-and-cut map [RW97], is presented in [MWD05].

Algorithm 8.5 I/S/O representation of the MPUM via a state sequence `uy2x2ss`

Input: u_d, y_d, $\mathtt{l}_{\max}$, and $\mathtt{n}_{\max}$.

1: Compute $\mathtt{n}_{\max}$, $\mathtt{l}_{\max}+1$ samples long sequential free responses Y_0 of the MPUM via Algorithm 8.9.
2: Compute a rank revealing factorization $Y_0 = \Gamma X_\mathrm{d}$.
3: Solve the system of equations (8.2) for A, B, C, D, where

$$\begin{bmatrix} x_\mathrm{d}(1) & \cdots & x_\mathrm{d}(\mathtt{n}_{\max}+\mathtt{m}+1) \end{bmatrix} := X_\mathrm{d}.$$

Output: (A, B, C, D)—a minimal input/state/output representation of the MPUM.

8.6 Computation of the Impulse Response from Data

In this section, we consider the following problem:

> Given a trajectory $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d})$ of a system $\mathscr{B} \in \mathscr{L}^{\mathtt{w}}$, find the first t samples of the impulse response of $\mathscr{B}$.

Under the conditions of the fundamental lemma, we have that

$$\operatorname{col\,span}\left(\mathscr{H}_t(w_\mathrm{d})\right) = \mathscr{B}|_{[1,t]}.$$

This implies that there exists a matrix G, such that $\mathscr{H}_t(y_\mathrm{d})G = H$. Thus the problem reduces to finding a particular G.

Define U_p, U_f, Y_p, and Y_f as follows:

$$\mathscr{H}_{\mathtt{l}_{\max}+t}(u_\mathrm{d}) =: \begin{bmatrix} U_\mathrm{p} \\ U_\mathrm{f} \end{bmatrix}, \qquad \mathscr{H}_{\mathtt{l}_{\max}+t}(y_\mathrm{d}) =: \begin{bmatrix} Y_\mathrm{p} \\ Y_\mathrm{f} \end{bmatrix}, \tag{8.3}$$

where $\operatorname{row\,dim}(U_\mathrm{p}) = \operatorname{row\,dim}(Y_\mathrm{p}) = \mathtt{l}_{\max}$ and $\operatorname{row\,dim}(U_\mathrm{f}) = \operatorname{row\,dim}(Y_\mathrm{f}) = t$.

Theorem 8.19 (Impulse response from data). *Let $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d})$ be a trajectory of a controllable LTI system $\mathscr{B} \in \mathscr{L}^{\mathtt{w},\mathtt{n}_{\max}}_{\mathtt{m},\mathtt{l}_{\max}}$ and let u_d be persistently exciting of order $t + \mathtt{l}_{\max} + \mathtt{n}_{\max}$. Then the system of equations*

$$\begin{bmatrix} U_\mathrm{p} \\ U_\mathrm{f} \\ Y_\mathrm{p} \end{bmatrix} G = \begin{bmatrix} 0_{\mathtt{m}\mathtt{l}_{\max}\times\mathtt{m}} \\ \begin{bmatrix} I_\mathtt{m} \\ 0_{\mathtt{m}(t-1)\times\mathtt{m}} \end{bmatrix} \\ 0_{\mathtt{p}\mathtt{l}_{\max}\times\mathtt{m}} \end{bmatrix} \tag{8.4}$$

is solvable for $G \in \mathbb{R}^{\bullet\times\mathtt{m}}$. Moreover, for any particular solution $\bar{G}$, the matrix $Y_\mathrm{f}\bar{G}$ contains the first t samples of the impulse response of $\mathscr{B}$, i.e.,

$$Y_\mathrm{f}\bar{G} = H.$$

Proof. Under the assumptions of the theorem, we can apply the fundamental lemma with $L = \mathtt{l}_{\max} + t$. Thus

$$\operatorname{col\,span}\left(\mathscr{H}_{\mathtt{l}_{\max}+t}(w_\mathrm{d})\right) = \mathscr{B}|_{[1,\mathtt{l}_{\max}+t]}.$$

First, we show that (8.4) is solvable. The impulse response $(\begin{bmatrix} I_{\mathtt{m}} \\ 0_{\mathtt{m}(t-1)\times\mathtt{m}} \end{bmatrix}, H)$ is a (matrix valued) response of $\mathscr{B}$ obtained under zero initial conditions. Because of the zero initial conditions, $(\begin{bmatrix} I_{\mathtt{m}} \\ 0_{\mathtt{m}(t-1)\times\mathtt{m}} \end{bmatrix}, H)$ preceded by any number of zeros remains a response of $\mathscr{B}$. Therefore, there exists a matrix $\bar{G}$, such that

$$\begin{bmatrix} U_{\mathrm{p}} \\ U_{\mathrm{f}} \\ Y_{\mathrm{p}} \\ Y_{\mathrm{f}} \end{bmatrix} \bar{G} = \begin{bmatrix} 0_{\mathtt{m}\mathtt{l}_{\max}\times\mathtt{m}} \\ \begin{bmatrix} I_{\mathtt{m}} \\ 0_{\mathtt{m}(t-1)\times\mathtt{m}} \end{bmatrix} \\ 0_{\mathtt{p}\mathtt{l}_{\max}\times\mathtt{m}} \\ H \end{bmatrix}.$$

This shows that there exists a solution $\bar{G}$ of (8.4) and therefore $Y_{\mathrm{f}}\bar{G}$ is the impulse response.

Conversely, let G be a solution of (8.4). We have

$$\begin{bmatrix} U_{\mathrm{p}} \\ U_{\mathrm{f}} \\ Y_{\mathrm{p}} \\ Y_{\mathrm{f}} \end{bmatrix} G = \begin{bmatrix} 0_{\mathtt{m}\mathtt{l}_{\max}\times\mathtt{m}} \\ \begin{bmatrix} I_{\mathtt{m}} \\ 0_{\mathtt{m}(t-1)\times\mathtt{m}} \end{bmatrix} \\ 0_{\mathtt{p}\mathtt{l}_{\max}\times\mathtt{m}} \\ Y_{\mathrm{f}}G \end{bmatrix} \tag{8.5}$$

and the fundamental lemma guarantees that the right-hand side of (8.5) is a response of $\mathscr{B}$. The response is identically zero during the first $\mathtt{l}_{\max}$ samples, which (using the assumption $\mathtt{l}_{\max} \geq \mathbf{l}(\mathscr{B})$) guarantees that the initial conditions are set to zero. The input $\begin{bmatrix} I_{\mathtt{m}} \\ 0_{\mathtt{m}(t-1)\times\mathtt{m}} \end{bmatrix}$ is a matrix valued impulse, so that the corresponding output $Y_{\mathrm{f}}G$ is indeed the impulse response H. □

Theorem 8.19 gives the following block algorithm for the computation of H.

Algorithm 8.6 Block computation of the impulse response from data `uy2hblk`

Input: u_{d}, y_{d}, $\mathtt{l}_{\max}$, and t.

1: Solve the system of equations (8.4). Let $\bar{G}$ be the computed solution.
2: Compute $H = Y_{\mathrm{f}}\bar{G}$.

Output: the first t samples of the impulse response H of the MPUM.

Note 8.20 (Efficient implementation via QR factorization) The system of equations (8.4) of step 1 of Algorithm 8.6 can be solved efficiently by first "compressing the data" via the QR factorization

$$\left[\begin{array}{c} U_{\mathrm{p}} \\ U_{\mathrm{f}} \\ \hline Y_{\mathrm{p}} \\ Y_{\mathrm{f}} \end{array}\right]^{\top} = QR, \qquad R^{\top} =: \left[\begin{array}{ccc} R_{11} & 0 & 0 \\ \hline R_{21} & R_{22} & 0 \end{array}\right],$$

where $R_{11} \in \mathbb{R}^{j\times j}$, $j = \mathtt{m}(\mathtt{l}_{\max} + t) + \mathtt{p}\mathtt{l}_{\max}$, and then computing the pseudoinverse of the R_{11} block. We have

$$H = Y_{\mathrm{f}} \begin{bmatrix} U_{\mathrm{p}} \\ U_{\mathrm{f}} \\ Y_{\mathrm{p}} \end{bmatrix}^{\dagger} \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix} = R_{21}R_{11}^{\dagger} \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}.$$

We proceed to point out an inherent limitation of Algorithm 8.6 when dealing with finite amount of data. Let a T samples long trajectory be given. The persistency of excitation assumption in Theorem 8.19 requires that $\mathscr{H}_{t+\mathtt{l}_{\max}+\mathtt{n}_{\max}}(u_{\mathrm{d}})$ be full row rank, which implies that

$$\mathtt{m}(t+\mathtt{l}_{\max}+\mathtt{n}_{\max}) \leq T-(t+\mathtt{l}_{\max}+\mathtt{n}_{\max})+1 \quad \Longrightarrow \quad t \leq \frac{T+1}{\mathtt{m}+1}-\mathtt{l}_{\max}-\mathtt{n}_{\max}.$$

Thus, using Algorithm 8.6, we are limited in the number of samples of the impulse response that can be computed. Moreover, for efficiency and accuracy (in the presence of noise), we want to have Hankel matrices U_{p}, U_{f}, etc., with many more columns than rows, which implies small t.

In fact, according to Theorem 8.16, u_{d} persistently exciting of order $1+\mathtt{l}_{\max}+\mathtt{n}_{\max}$ is sufficient for computation of the whole impulse response of the system. Indeed, this can be done by weaving trajectories. (See Figure 8.2 for an illustration.)

Lemma 8.21 (Weaving trajectories). *Consider a system $\mathscr{B} \in \mathscr{L}^{\mathrm{w}}$ and let*

1. *$w_{\mathrm{d},1}$ be a T_1 samples long trajectory of $\mathscr{B}$, i.e., $w_{\mathrm{d},1} \in \mathscr{B}|_{[1,T_1]}$;*
2. *$w_{\mathrm{d},2}$ be a T_2 samples long trajectory of $\mathscr{B}$, i.e., $w_{\mathrm{d},2} \in \mathscr{B}|_{[1,T_2]}$; and*
3. *the last $\mathtt{l}_{\max}$ samples, where $\mathtt{l}_{\max} \geq \mathbf{l}(\mathscr{B})$, of $w_{\mathrm{d},1}$ coincide with the first $\mathtt{l}_{\max}$ samples of $w_{\mathrm{d},2}$, i.e.,*

$$\big(w_{\mathrm{d},1}(T_1-\mathtt{l}_{\max}+1),\dots,w_{\mathrm{d},1}(T_1)\big) = \big(w_{\mathrm{d},2}(1),\dots,w_{\mathrm{d},2}(\mathtt{l}_{\max})\big).$$

Then the trajectory

$$w := \big(w_{\mathrm{d},1}(1),\dots,w_{\mathrm{d},1}(T_1),\, w_{\mathrm{d},2}(\mathtt{l}_{\max}+1),\dots,w_{\mathrm{d},2}(T_2)\big) \tag{8.6}$$

obtained by weaving together $w_{\mathrm{d},1}$ and $w_{\mathrm{d},2}$ is a trajectory of $\mathscr{B}$, i.e., $w \in \mathscr{B}|_{[1,T_1+T_2-\mathtt{l}_{\max}]}$.

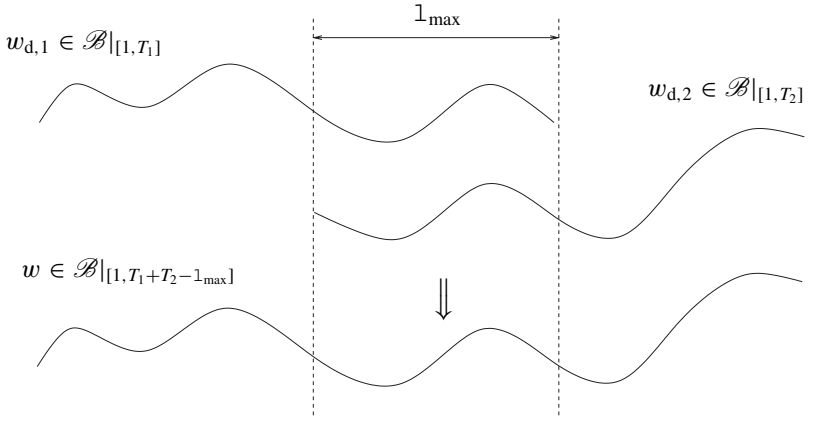

Figure 8.2. *Weaving trajectories.*

Proof. Let $x_{\mathrm{d},1} := \big(x_{\mathrm{d},1}(1), \ldots, x_{\mathrm{d},1}(T_1+1)\big)$ and $x_{\mathrm{d},2} := \big(x_{\mathrm{d},2}(1), \ldots, x_{\mathrm{d},2}(T_2+1)\big)$ be state sequences of $\mathscr{B}$ associated with $w_{\mathrm{d},1}$ and $w_{\mathrm{d},2}$, respectively. Assumption 3 implies that $x_{\mathrm{d},1}(T_1+1) = x_{\mathrm{d},2}(\mathtt{l}_{\max}+1)$. Therefore, (8.6) is a trajectory of $\mathscr{B}$. ∎

Algorithm 8.7 overcomes the above-mentioned limitation of the block algorithm by iteratively computing blocks of L consecutive samples, where

$$1 \le L \le \frac{T+1}{\mathtt{m}+1} - \mathtt{l}_{\max} - \mathtt{n}_{\max}. \tag{8.7}$$

Moreover, monitoring the decay of H (provided the system is stable) while computing it gives a heuristic way to determine a value for t that is sufficiently large to show the transient.

Algorithm 8.7 Iterative computation of the impulse response from data `uy2h`

Input: u_{d}, y_{d}, $\mathtt{n}_{\max}$, $\mathtt{l}_{\max}$, and either t or a convergence tolerance ε.

1: Choose the number of samples L computed in one iteration step according to (8.7).

2: Initialization: $k := 0$, $F_{\mathrm{u}}^{(0)} := \begin{bmatrix} 0_{\mathtt{m}\mathtt{l}_{\max}\times\mathtt{m}} \\ \begin{bmatrix} I_{\mathtt{m}} \\ 0_{\mathtt{m}(L-1)\times\mathtt{m}} \end{bmatrix} \end{bmatrix}$, and $F_{\mathrm{y,p}}^{(0)} := 0_{\mathtt{p}\mathtt{l}_{\max}}$.

3: **repeat**

4: Solve the system $\begin{bmatrix} U_{\mathrm{p}} \\ U_{\mathrm{f}} \\ Y_{\mathrm{p}} \end{bmatrix} G^{(k)} = \begin{bmatrix} F_{\mathrm{u}}^{(k)} \\ F_{\mathrm{y,p}}^{(k)} \end{bmatrix}$.

5: Compute the response $H^{(k)} := F_{\mathrm{y,f}}^{(k)} := Y_{\mathrm{f}} G^{(k)}$.

6: Define $F_{\mathrm{y}}^{(k)} := \begin{bmatrix} F_{\mathrm{y,p}}^{(k)} \\ F_{\mathrm{y,f}}^{(k)} \end{bmatrix}$.

7: Shift F_{u} and F_{y}: $F_{\mathrm{u}}^{(k+1)} := \begin{bmatrix} \sigma^L F_{\mathrm{u}}^{(k)} \\ 0_{\mathtt{m}L\times\mathtt{m}} \end{bmatrix}$, $F_{\mathrm{y,p}}^{(k+1)} := \sigma^L F_{\mathrm{y}}^{(k)}$.

8: Increment the iteration counter $k := k+1$.

9: **until** $\begin{cases} kL < t & \text{if } t \text{ is given,} \\ \|H^{(k-1)}\|_{\mathrm{F}} \le \varepsilon & \text{otherwise.} \end{cases}$

Output: $H = \operatorname{col}\big(H^{(0)}, \ldots, H^{(k-1)}\big)$.

In the recursive algorithm, the matrices U_{p}, U_{f}, Y_{p}, and Y_{f} defined above are redefined as follows:

$$\mathscr{H}_{\mathtt{l}_{\max}+L}(u_{\mathrm{d}}) =: \begin{bmatrix} U_{\mathrm{p}} \\ U_{\mathrm{f}} \end{bmatrix}, \qquad \mathscr{H}_{\mathtt{l}_{\max}+L}(y_{\mathrm{d}}) =: \begin{bmatrix} Y_{\mathrm{p}} \\ Y_{\mathrm{f}} \end{bmatrix},$$

where $\operatorname{row\,dim}(U_{\mathrm{p}}) = \operatorname{row\,dim}(Y_{\mathrm{p}}) = \mathtt{l}_{\max}$ and $\operatorname{row\,dim}(U_{\mathrm{f}}) = \operatorname{row\,dim}(Y_{\mathrm{f}}) = L$.

Proposition 8.22. *Let $w_{\mathrm{d}} = (u_{\mathrm{d}}, y_{\mathrm{d}})$ be a trajectory of a controllable LTI system $\mathscr{B}$ of order $\mathbf{n}(\mathscr{B}) \le \mathtt{n}_{\max}$ and lag $\mathbf{l}(\mathscr{B}) \le \mathtt{l}_{\max}$, and let u_{d} be persistently exciting of order $L + \mathtt{l}_{\max} + \mathtt{n}_{\max}$. Then Algorithm* 8.7 *computes the first t samples of the impulse response of $\mathscr{B}$.*

Proof. Under the assumptions of the proposition, we can apply Theorem 8.19, with the parameter t replaced by the parameter L, selected in step 1 of the algorithm. Steps 4 and 5 of

the recursive algorithm correspond to steps 1 and 2 of the block algorithm (Algorithm 8.6). The right-hand side $\begin{bmatrix} F_{\mathrm{u}}^{(k)} \\ F_{\mathrm{y,p}}^{(k)} \end{bmatrix}$ of the system of equations, solved in step 4, is initialized so that $H^{(0)}$ is indeed the matrix of the first L samples of the impulse response.

The response computed on the $(k+1)$st iteration step, $k \geq 1$, is a response due to zero input, and its first $\mathtt{l}_{\max}$ samples overlap the last $\mathtt{l}_{\max}$ samples of the response computed on the kth iteration step. By the weaving lemma (Lemma 8.21), their concatenation is a valid response. Applying this argument recursively, we have that H computed by the algorithm is the impulse response of the system. □

With $L = 1$, the persistency of excitation condition required by Proposition 8.22 is $\mathtt{l}_{\max} + 1 + \mathtt{n}_{\max}$, which is the identifiability condition of Theorem 8.16 (with the unknown lag $\mathbf{l}(\mathscr{B})$ and order $\mathbf{n}(\mathscr{B})$ replaced by their given upper bounds $\mathtt{l}_{\max}$ and $\mathtt{n}_{\max}$).

Note 8.23 (Data driven simulation) In [MWRM05], Theorem 8.19 and Algorithms 8.6 and 8.7 are modified to compute an arbitrary response directly from data. This procedure is called data driven simulation and is shown to be related to deterministic subspace identification algorithms.

Note 8.24 (Efficient implementation via QR factorization) The most expensive part of Algorithm 8.7 is solving the system of equations in step 4. It can be solved efficiently via the QR factorization, as described in Note 8.20. Moreover, since the matrix on the left-hand side of the system is fixed, the pseudoinverse can be computed outside the iteration loop and used for all iterations.

8.7 Realization Theory and Algorithms

The problem of passing from an impulse response to another representation (typically input/state/output or transfer function) is called realization. Given a sequence $H : \mathbb{N} \to \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$, we say that a system $\mathscr{B} \in \mathscr{L}_{\mathtt{m}}^{\mathtt{w}}$, $\mathtt{w} := \mathtt{m}+\mathtt{p}$, realizes H if $\mathscr{B}$ has a convolution representation with an impulse response H. In this case, we say that H is realizable (by a system in the model class $\mathscr{L}_{\mathtt{m}}^{\mathtt{w}}$). A sequence H might not be realizable by a *finite dimensional* LTI system, but if it is realizable, the realization is unique.

Theorem 8.25 (Test for realizability). *The sequence $H : \mathbb{N} \to \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$ is realizable by a finite dimensional LTI system with $\mathtt{m}$ inputs if and only if the two-sided infinite Hankel matrix $\mathscr{H}(\sigma H)$ has a finite rank. Moreover, if the rank of $\mathscr{H}(\sigma H)$ is $\mathtt{n}$, then there is a unique system $\mathscr{B} \in \mathscr{L}_{\mathtt{m}}^{\mathtt{w},\mathtt{n}}$ that realizes H.*

Let H be realizable by a system $\mathscr{B} \in \mathscr{L}_{\mathtt{m}}^{\mathtt{w}}$ with an input/state/output representation $\mathscr{B} = \mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$. We have that

$$\mathscr{H}_{i,j}(\sigma H) = \mathscr{O}_i(A, C)\mathscr{C}_j(A, B),$$

and from the properties of the controllability and observability matrices, it follows that

$$\operatorname{rank}\left(\mathscr{H}_{i,j}(\sigma H)\right) = \begin{cases} \min(\mathtt{p}i, \mathtt{m}j) & \text{for all } i < \mu(\mathscr{B}) \text{ and } j < \nu(\mathscr{B}), \\ \mathbf{n}(\mathscr{B}) & \text{otherwise.} \end{cases}$$

Therefore, if we know that H is an impulse response of a finite dimensional LTI system $\mathscr{B}$ of order $\mathbf{n}(\mathscr{B}) \leq \mathtt{n}_{\max}$ and lag $\mathbf{l}(\mathscr{B}) \leq \mathtt{l}_{\max}$, where $\mathtt{n}_{\max}$ and $\mathtt{l}_{\max}$ are given, we can find $\mathbf{n}(\mathscr{B})$ by a rank computation as follows:

$$\mathbf{n}(\mathscr{B}) = \operatorname{rank}\left(\mathscr{H}_{\mathtt{l}_{\max}+1,\mathtt{n}_{\max}}(\sigma H)\right).$$

This fact is often used in subspace identification. Moreover, the SVD $\mathscr{H}_{t,t}(\sigma H) = U\Sigma V^{\top}$, $t > \mathtt{n}_{\max}$, allows us to find a finite time t balanced approximation of the MPUM, so that the numerical rank computation of the block-Hankel matrix of the Markov parameters is a good heuristic for approximate identification.

Note 8.26 (Realization and exact identification) Clearly, realization is a special exact identification problem. Realization of $H : \mathbb{N} \to \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$ is equivalent to exact identification of the time series

$$\begin{aligned} w_{\mathrm{d},1} = (u_{\mathrm{d},1}, y_{\mathrm{d},1}) &:= \big(\operatorname{col}(0, \delta e_1), \operatorname{col}(0, h_1)\big), \\ &\cdots \\ w_{\mathrm{d},\mathtt{m}} = (u_{\mathrm{d},\mathtt{m}}, y_{\mathrm{d},\mathtt{m}}) &:= \big(\operatorname{col}(0, \delta e_{\mathtt{m}}), \operatorname{col}(0, h_{\mathtt{m}})\big), \end{aligned}$$

where $\begin{bmatrix} h_1 & \cdots & h_{\mathtt{m}} \end{bmatrix} := H$, δ is the Kronecker delta function, $\begin{bmatrix} e_1 & \cdots & e_{\mathtt{m}} \end{bmatrix} := I_{\mathtt{m}}$, and the zero prefix is $\mathtt{l}_{\max}$ samples long. (The zero prefix fixes the initial conditions to be zero, which otherwise are free in the exact identification problem.) Special purpose realization methods, however, are more efficient than a general exact identification algorithm.

Note 8.27 (Realization and exact identification of an autonomous system) An alternative point of view of realization is as an exact identification of an autonomous system: realization of $H : \mathbb{N} \to \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$ is equivalent to exact identification of the time series

$$w_{\mathrm{d},1} = (u_{\mathrm{d},1}, y_{\mathrm{d},1}) := (0, \sigma h_1), \quad \ldots \quad , w_{\mathrm{d},\mathtt{m}} = (u_{\mathrm{d},\mathtt{m}}, y_{\mathrm{d},\mathtt{m}}) := (0, \sigma h_{\mathtt{m}}).$$

Consider the impulse response H of the system

$$\mathscr{B}_{\mathrm{i/s/o}}\left(A, \begin{bmatrix} b_1 & \cdots & b_{\mathtt{m}} \end{bmatrix}, C, \bullet\right)$$

and the responses $y_1, \ldots, y_{\mathtt{m}}$ of the autonomous system $\mathscr{B}_{\mathrm{i/s/o}}(A, C)$ due to the initial conditions $b_1, \ldots, b_{\mathtt{m}}$. It is easy to verify that

$$\sigma H = \begin{bmatrix} y_1 & \cdots & y_{\mathtt{m}} \end{bmatrix}.$$

Thus, with an obvious substitution,

> realization algorithms can be used for exact identification of an autonomous system and vice versa; algorithms for identification of an autonomous systems can be used for realization.

Once we know from Theorem 8.25 or from prior knowledge that a given time series $H := \big(H(0), H(1), \ldots, H(T)\big)$ is realizable in the model class $\mathscr{L}^{\mathrm{w}}_{\mathtt{m},\mathtt{l}_{\max}}$, we can proceed with the problem of finding a representation of the system that realizes H. General exact

Algorithm 8.8 Realization algorithm `h2ss`

Input: H and $\mathtt{l}_{\max}$ satisfying the conditions of Theorem 8.25.

1: Compute a rank revealing factorization of the Hankel matrix $\mathscr{H}_{\mathtt{l}_{\max}+1}(\sigma H) = \Gamma\Delta$.
2: Let $D = H(0)$, C be the first block row of Γ, and B be the first block column of Δ.
3: Solve the shift equation $(\sigma^*\Gamma)A = \sigma\Gamma$.

Output: parameters (A, B, C, D) of a minimal input/state/output realization of H.

identification problems can be used but in the special case at hand there are more efficient alternatives. Algorithm 8.8 is a typical realization algorithm.

The key computational step is the rank revealing factorization of the Hankel matrix $\mathscr{H}_{\mathtt{l}_{\max}+1}(H)$. Moreover, this step determines the state basis in which the parameters A, B, C, D are computed. In case of finite precision arithmetic, it is well known that rank computation is a nontrivial problem. The rank revealing factorization is crucial for the outcome of the algorithm because the rank of $\mathscr{H}_{\mathtt{l}_{\max}+1}(H)$ is the order of the realization.

When the given time series H is not realizable by an LTI system of order $\mathtt{n}_{\max} := \mathtt{pl}_{\max}$, i.e., $\mathscr{H}_{\mathtt{l}_{\max}+1}(\sigma H)$ is full rank, the SVD offers a possibility to find approximate realization in the model class $\mathscr{L}^{\mathrm{w}}_{\mathtt{m},\mathtt{l}_{\max}}$; see also Note 8.18 on page 120. Replace the rank revealing factorization in step 1 of Algorithm 8.8 by the SVD $\mathscr{H}_{\mathtt{l}_{\max}+1}(H) = U\Sigma V^\top$ and the definitions $\Gamma := U\sqrt{\Sigma}$ and $\Delta := \sqrt{\Sigma}V^\top$. This can be viewed as computation of an "approximate rank revealing factorization". Note that in this case the finite time controllability and observability gramians are equal,

$$\Gamma^\top\Gamma = \Delta\Delta^\top = \Sigma,$$

so that the computed realization $\mathscr{B}_{\mathrm{i/s/o}}(A, B, C, D)$ is in a finite time $\mathtt{l}_{\max}$ balanced form. Algorithm 8.8 with the above modification is Kung's algorithm [Kun78].

8.8 Computation of Free Responses

In this section, we consider the following problem:

> Given $w_{\mathrm{d}} = (u_{\mathrm{d}}, y_{\mathrm{d}}) \in \mathscr{B}$, find (sequential) free responses Y_0 of $\mathscr{B}$.

By "sequential", we mean that the initial conditions corresponding to the columns of Y_0 form a valid state sequence of $\mathscr{B}$.

First, we consider computation of general free responses. Using the fundamental lemma, a set of t samples long free responses can be computed from data as follows:

$$\begin{bmatrix} \mathscr{H}_t(u_{\mathrm{d}}) \\ \mathscr{H}_t(y_{\mathrm{d}}) \end{bmatrix} G = \begin{bmatrix} 0 \\ Y_0 \end{bmatrix}. \tag{8.8}$$

Therefore, for any G that satisfies $\mathscr{H}_t(u_{\mathrm{d}})G = 0$, the columns of $Y_0 := \mathscr{H}_t(y_{\mathrm{d}})G$ are free responses. The columns of G are vectors in the null space of $\mathscr{H}_t(u_{\mathrm{d}})$ and can be computed explicitly; however, in general, $\operatorname{rank}(Y_0) \leq \mathbf{n}(\mathscr{B})$. The condition $\operatorname{rank}(Y_0) = \mathbf{n}(\mathscr{B})$ is needed for identification of an input/state/output representation of the MPUM, as outlined in Algorithm 8.3.

In order to ensure the rank condition, we use the splitting of the data into "past" and "future" as defined in (8.3). The blocks in the past allow us to restrict the matrix G, so that the initial conditions X_{ini} under which the responses Y_0 are generated satisfy $\operatorname{rank}(X_{\text{ini}}) = \mathbf{n}(\mathscr{B})$. This implies $\operatorname{rank}(Y_0) = \mathbf{n}(\mathscr{B})$. In turns out, however, that in choosing the initial conditions X_{ini}, we can furthermore produce sequential free responses.

Using the fundamental lemma, we know that the right-hand side of the equation

$$\begin{bmatrix} U_{\text{p}} \\ U_{\text{f}} \\ Y_{\text{p}} \\ Y_{\text{f}} \end{bmatrix} G = \begin{bmatrix} U_{\text{p}} \\ 0 \\ Y_{\text{p}} \\ Y_0 \end{bmatrix}$$

is a trajectory. Therefore, a set of free responses can be computed from data by solving the system of equations

$$\begin{bmatrix} U_{\text{p}} \\ U_{\text{f}} \\ Y_{\text{p}} \end{bmatrix} G = \begin{bmatrix} U_{\text{p}} \\ 0 \\ Y_{\text{p}} \end{bmatrix} \tag{8.9}$$

and setting $Y_0 = Y_{\text{f}} G$. Moreover, the Hankel structure of U_{p} and Y_{p} imply that Y_0 is a matrix of sequential responses. System (8.9) and $Y_0 = Y_{\text{f}} G$ give a block algorithm for the computation of sequential free responses. It is analogous to the block algorithm for the computation of the impulse response and again the computation can be performed efficiently via the QR factorization.

We proceed to present a recursive algorithm for the computation of Y_0, analogous to Algorithm 8.7 for the computation of the impulse response. An advantage of the recursive algorithm over the block one is that one is not restricted by the finite amount of data w_{d} to a finite length response Y_0.

Proposition 8.28. *Under the assumptions of Proposition* 8.22*, Algorithm* 8.9 *computes a matrix of sequential free responses of $\mathscr{B}$ with t block rows.*

Proof. This is similar to the proof of Proposition 8.22. ∎

8.9 Relation to Subspace Identification Methods*

MOESP-type Algorithms

The multivariable output error state space (MOESP)-type subspace identification algorithms correspond to the algorithm based on the computation of free responses as outlined in Section 8.5, Algorithm 8.4. However, in the MOESP algorithms, step 1—the computation of free responses—is implemented via the *orthogonal projection*

$$Y_0 := \mathscr{H}_{1_{\max}+1}(y_{\text{d}}) \underbrace{\Big(I - \mathscr{H}^{\top}_{1_{\max}+1}(u_{\text{d}}) \big(\mathscr{H}_{1_{\max}+1}(u_{\text{d}}) \mathscr{H}^{\top}_{1_{\max}+1}(u_{\text{d}}) \big)^{-1} \mathscr{H}_{1_{\max}+1}(u_{\text{d}}) \Big)}_{\Pi^{\perp}_{u_{\text{d}}}}; \tag{8.10}$$

i.e., the MOESP algorithms compute the orthogonal projection of the rows of $\mathscr{H}_{1_{\max}+1}(y_{\text{d}})$ on the orthogonal complement of the row space of $\mathscr{H}_{1_{\max}+1}(u_{\text{d}})$. In subspace identification

Algorithm 8.9 Iterative computation of sequential free responses `uy2y0`

Input: u_d, y_d, $\mathtt{n}_{\max}$, $\mathtt{l}_{\max}$, and either the desired number of samples t or a convergence tolerance ε.

1: Choose the number of samples L computed in one iteration step according to (8.7).

2: Initialization: $k := 0$, $F_\mathrm{u}^{(0)} := \begin{bmatrix} U_\mathrm{p} \\ 0 \end{bmatrix}$, and $F_\mathrm{y,p}^{(0)} := Y_\mathrm{p}$.

3: **repeat**

4: Solve the system $\begin{bmatrix} U_\mathrm{p} \\ U_\mathrm{f} \\ Y_\mathrm{p} \end{bmatrix} G^{(k)} = \begin{bmatrix} F_\mathrm{u}^{(k)} \\ F_\mathrm{y,p}^{(k)} \end{bmatrix}$.

5: Compute the response $Y_0^{(k)} := F_\mathrm{y,f}^{(k)} := Y_\mathrm{f} G^{(k)}$.

6: Define $F_\mathrm{y}^{(k)} := \begin{bmatrix} F_\mathrm{y,p}^{(k)} \\ F_\mathrm{y,f}^{(k)} \end{bmatrix}$.

7: Shift F_u and F_y: $F_\mathrm{u}^{(k+1)} := \begin{bmatrix} \sigma^L F_\mathrm{u}^{(k)} \\ 0_{\mathtt{m}L \times \mathtt{m}} \end{bmatrix}$ and $F_\mathrm{y,p}^{(k+1)} := \sigma^L F_\mathrm{y}^{(k)}$.

8: Increment the iteration counter $k := k + 1$.

9: **until** $\begin{cases} kL < t & \text{if } t \text{ is given,} \\ \|Y_0^{(k-1)}\|_\mathrm{F} \le \varepsilon & \text{otherwise.} \end{cases}$

Output: $Y_0 = \operatorname{col}\left(Y_0^{(0)}, \ldots, Y_0^{(k-1)}\right)$.

it is customary to think in terms of geometric operations: projection of the rows of a certain matrix onto the row space of another matrix. The fact that these matrices have special (block-Hankel) structure is ignored and the link with system theory is lost. Still, as we show next,

> the orthogonal projection (8.10) has the simple and useful system theoretic interpretation of computing a maximal number of free responses.

Observe that

$$\begin{bmatrix} \mathscr{H}_{\mathtt{l}_{\max}+1}(u_\mathrm{d}) \\ \mathscr{H}_{\mathtt{l}_{\max}+1}(y_\mathrm{d}) \end{bmatrix} \Pi_{u_\mathrm{d}}^{\perp} = \begin{bmatrix} 0 \\ Y_0 \end{bmatrix},$$

which corresponds to (8.8) except that now the projector $\Pi_{u_\mathrm{d}}^{\perp}$ is a square matrix, while in (8.8) G is in general a rectangular matrix. In [VD92, Section 3.3], it is shown that a sufficient condition for $\operatorname{rank}(Y_0) = \mathbf{n}(\mathscr{B})$ is

$$\operatorname{rank}\left(\begin{bmatrix} X_\mathrm{ini} \\ \mathscr{H}_{\mathtt{l}_{\max}+1}(u_\mathrm{d}) \end{bmatrix}\right) = \mathbf{n}(\mathscr{B}) + (\mathtt{l}_{\max} + 1)\mathtt{m}. \tag{8.11}$$

This condition, however, is not verifiable from the data $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d})$. Therefore, given w_d, one cannot check in general whether the data generating system $\mathscr{B}$ is identifiable by the MOESP algorithms. Under the identifiability condition

u_d persistently exciting of order $\mathtt{l}_{\max} + 1 + \mathtt{n}_{\max}$,

which is verifiable from the data, Corollary 8.17 implies (8.11).

Finally, note that the $j = T - \mathtt{l}_{\max}$ free responses that the orthogonal projection (8.10) computes are typically more than necessary for exact identification, i.e., $j \gg \mathbf{n}(\mathscr{B})$. Therefore, in general, the orthogonal projection is a computationally inefficient operation for exact identification. This deficiency of the MOESP algorithms is partially corrected on the level of the numerical implementation. First, the QR factorization

$$\begin{bmatrix} \mathscr{H}_{\mathtt{n}_{\max}}(u_{\mathrm{d}}) \\ \mathscr{H}_{\mathtt{n}_{\max}}(y_{\mathrm{d}}) \end{bmatrix}^{\top} = QR$$

is computed and then only the block entry R_{22} of the R factor is used, where

$$R^{\top} =: \begin{array}{c} \begin{matrix} \mathtt{n}_{\max}\mathtt{m} & \mathtt{n}_{\max}\mathtt{p} & \end{matrix} \\ \begin{bmatrix} R_{11} & 0 & 0 \\ R_{21} & R_{22} & 0 \end{bmatrix} \end{array} \begin{matrix} \\ \mathtt{n}_{\max}\mathtt{m} \\ \mathtt{n}_{\max}\mathtt{p} \end{matrix}.$$

It can be shown (see [VD92, Section 4.1]), that

$$\operatorname{col\,span}(Y_0) = \operatorname{col\,span}(R_{22}).$$

The column dimension of R_{22} is $\mathtt{n}_{\max}\mathtt{p}$, which is (typically) comparable with $\mathtt{n}_{\max}$ and is (typically) much smaller than j.

N4SID-type Algorithms

The numerical algorithms for subspace state space system identification (N4SID) correspond to the algorithm based on the computation of a state sequence as outlined in Section 8.5, Algorithm 8.5. However, in the N4SID-type algorithms, step 1—the computation of sequential free responses—is implemented via the *oblique projection*. Consider the splitting of the data into "past" and "future",

$$\mathscr{H}_{2(\mathtt{l}_{\max}+1)}(u_{\mathrm{d}}) =: \begin{bmatrix} U_{\mathrm{p}} \\ U_{\mathrm{f}} \end{bmatrix}, \qquad \mathscr{H}_{2(\mathtt{l}_{\max}+1)}(y_{\mathrm{d}}) =: \begin{bmatrix} Y_{\mathrm{p}} \\ Y_{\mathrm{f}} \end{bmatrix}, \tag{8.12}$$

with $\operatorname{row\,dim}(U_{\mathrm{p}}) = \operatorname{row\,dim}(U_{\mathrm{f}}) = \operatorname{row\,dim}(Y_{\mathrm{p}}) = \operatorname{row\,dim}(Y_{\mathrm{f}}) = \mathtt{l}_{\max} + 1$, and let

$$W_{\mathrm{p}} := \begin{bmatrix} U_{\mathrm{p}} \\ Y_{\mathrm{p}} \end{bmatrix}.$$

As the key computational step of the MOESP algorithms is the orthogonal projection, the key computational step of the N4SID algorithms is the oblique projection of Y_{f} along the space spanned by the rows of U_{f} onto the space spanned by the rows of W_{p}. This geometric operation, denoted by $Y_{\mathrm{f}}/_{U_{\mathrm{f}}}W_{\mathrm{p}}$, is defined as follows (see [VD96, equation (1.4), page 21]):

$$Y_0 := Y_{\mathrm{f}}/_{U_{\mathrm{f}}}W_{\mathrm{p}} := Y_{\mathrm{f}} \underbrace{\begin{bmatrix} W_{\mathrm{p}}^{\top} & U_{\mathrm{f}}^{\top} \end{bmatrix} \begin{bmatrix} W_{\mathrm{p}}W_{\mathrm{p}}^{\top} & W_{\mathrm{p}}U_{\mathrm{f}}^{\top} \\ U_{\mathrm{f}}W_{\mathrm{p}}^{\top} & U_{\mathrm{f}}U_{\mathrm{f}}^{\top} \end{bmatrix}^{+} \begin{bmatrix} W_{\mathrm{p}} \\ 0 \end{bmatrix}}_{\Pi_{\mathrm{obl}}}. \tag{8.13}$$

Next, we show that

the oblique projection computes sequential free responses of the system.

Note that

$$\begin{bmatrix} W_{\mathrm{p}} \\ U_{\mathrm{f}} \\ Y_{\mathrm{f}} \end{bmatrix} \Pi_{\mathrm{obl}} = \begin{bmatrix} W_{\mathrm{p}} \\ 0 \\ Y_0 \end{bmatrix}$$

corresponds to (8.9) except that the oblique projector Π_{obl} is a square matrix, while in (8.9), G is in general rectangular. Therefore, the columns of the oblique projection Y_0 given in (8.13) are $j := T - 2\mathtt{l}_{\max} - 1$ sequential free responses. However, as with the orthogonal projection, the oblique projection also computes in general more responses than the $\mathtt{n}_{\max} + \mathtt{m} + 2$ ones needed for applying Algorithm 8.5.

In [VD96, Section 2, Theorem 2], it is (implicitly) proven that a sufficient condition for $\operatorname{rank}(X_{\mathrm{d}}) = \mathbf{n}(\mathscr{B})$, which is needed for the exact identification Algorithm 8.5, is

1. u_{d} persistently exciting of order $2\mathtt{n}_{\max}$ and
2. $\operatorname{row span}(X_{\mathrm{d}}) \cap \operatorname{row span}(U_{\mathrm{f}}) = \{0\}$;

see assumptions 1 and 2 of [VD96, Section 2, Theorem 2]. As with assumption (8.11) in the MOESP algorithms, however, assumption 2 is again not verifiable from the given data. Persistency of excitation of u_{d} of order $2(\mathtt{l}_{\max} + 1) + \mathbf{n}(\mathscr{B})$ (i.e., the assumption of the fundamental lemma) is a sufficient condition, verifiable from the data $(u_{\mathrm{d}}, y_{\mathrm{d}})$, for assumptions 1 and 2 of [VD96, Section 2, Theorem 2].

8.10 Simulation Examples

Impulse Response Computation

First, we consider the problem of computing the first t samples of the impulse response H of a system $\mathscr{B}$ from data $w_{\mathrm{d}} := (u_{\mathrm{d}}, y_{\mathrm{d}})$. We choose a random stable system $\mathscr{B}$ of order $\mathtt{n} = 4$ with $\mathtt{m} = 2$ inputs and $\mathtt{p} = 2$ outputs. The data w_{d} is obtained according to the EIV model $w_{\mathrm{d}} = \bar{w} + \tilde{w}$, where $\bar{w} := (\bar{u}, \bar{y}) \in \mathscr{B}|_{[1,T]}$ with $T = 500$, $\bar{u}$ is zero mean unit variance white Gaussian noise, and $\tilde{w}$ is a zero mean white Gaussian noise with variance σ^2. Varying σ, we study empirically the effect of random perturbation on the results.

We apply Algorithm 8.6 with $t = 27$, $\mathtt{n}_{\max} = \mathtt{n}$, and $\mathtt{l}_{\max} = \lceil \mathtt{n}_{\max}/\mathtt{p} \rceil$. The computed impulse response is denoted by $\hat{H}$ and is compared with the "true" impulse response H obtained from $\mathscr{B}$ by simulation. The comparison is in terms of the Frobenius norm $e = ||H - \hat{H}||_{\mathrm{F}}$ of the approximation error $H - \hat{H}$. We also apply Algorithm 8.7 with parameters $\mathtt{n}_{\max} = \mathtt{n}$, $\mathtt{l}_{\max} = \lceil \mathtt{n}_{\max}/\mathtt{p} \rceil$, $L = 12$, and the function `impulse` from the System Identification Toolbox of MATLAB that estimates impulse response from data.

Table 8.1 shows the approximation errors e and execution times for four different noise levels and for the three compared algorithms. (The efficiency is measured by the execution time and not by the floating point operations (flops) because the function `impulse` is available only in the latter versions of MATLAB that do not support flop counts.)

In the absence of noise, both Algorithm 8.6 and Algorithm 8.7 compute up to numerical errors exactly the impulse response H, while the same is not true for the function `impulse`.

Table 8.1. *Error of approximation* $e = ||H - \hat{H}||_{\mathrm{F}}$ *and execution time in seconds for Algorithm* 8.6*, Algorithm* 8.7 *with* $L = 12$*, and the function* `impulse`*.*

	$\sigma = 0.0$		$\sigma = 0.01$		$\sigma = 0.05$		$\sigma = 0.1$	
Method	e	time, s	e	time, s	e	time, s	e	time, s
Algorithm 8.6	10^{-14}	0.293	0.029	0.277	0.096	0.285	0.251	0.279
Algorithm 8.7	10^{-14}	0.066	0.023	0.086	0.066	0.068	0.201	0.087
`impulse`	0.059	0.584	0.067	0.546	0.109	0.573	0.249	0.558

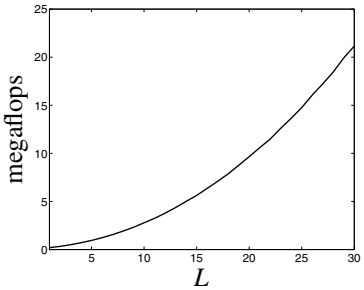

Figure 8.3. *Number of flops as a function of the parameter* L*.*

The simulation results show that the iterative algorithm is faster than the block algorithm. Also, when the given data w_{d} is noisy, the iterative algorithm outperforms the block algorithm and the function `impulse`.

Next, we show the effect of the parameter L on the number of flops and the error of approximation e. The plot in Figure 8.3 shows the number of flops, measured in megaflops, as a function of L. The function is monotonically increasing, so that most efficient is the computation for $L = 1$. The plots in Figure 8.4 show the approximation error e as a function of L for four different noise levels. The results are averaged for 100 noise realizations. The function $e(t)$ is complicated and is likely to depend on many factors. The graphs, however, indicate that in the presence of noise, there is a trade-off between computational efficiency and approximation error. For small L the computational cost is small, but the error e tends to be large.

Comparison of Exact Identification Algorithms

We compare the numerical efficiency of the following algorithms for deterministic identification:

`uy2ssmr` Algorithm of Moonen and Ramos [MR93]; see Algorithm 9.6;

`uy2ssvd` Algorithm of Van Overschee and De Moor [VD96]; see Algorithm 9.5;

"Deterministic algorithm 1" of Section 2.4.1 in [VD96] is combined with the choice of the weight matrices W_1 and W_2 given in Theorem 13, Section 5.4.1. Our implementation, however, differs from the outline of the algorithms given in [VD96]; see Note 9.4 on page 143;

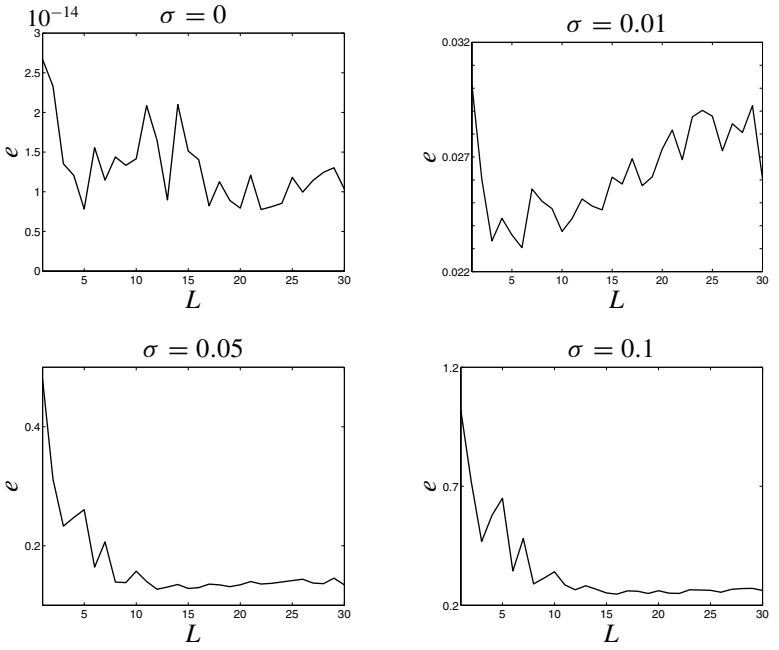

Figure 8.4. *Error of approximation* $e = ||H - \hat{H}||_{\mathrm{F}}$ *as a function of the parameter* L *for different noise levels* σ.

`det_stat` "Deterministic algorithm 1" of [VD96, Section 2.4.1] (implementation `det_stat.m` supplementing the book);

`det_alt` "Deterministic algorithm 2" of [VD96, Section 2.4.2] (implementation `det_alt.m` supplementing the book);

`projec` "Projection algorithm" of [VD96, Section 2.3.1] (implementation `projec.m` supplementing the book);

`intersec` "Intersection algorithm" of [VD96, Section 2.3.2] (implementation `intersec.m` supplementing the book);

`moesp` A deterministic version of the MOESP algorithm;

`uy2ssbal` The algorithm for deterministic balanced subspace identification proposed in Chapter 9 (with parameter $L = 1$); see Algorithm 9.4;

`uy2h2ss` Algorithm 8.7 (with $L = 1$) applied for the computation of the first $2\mathtt{n}_{\max} + 1$ samples of the impulse response, followed by Kung's algorithm for the realization of the impulse response; see Algorithm 8.3.

For the experiments we generate a random stable `n`th order system $\mathscr{B}$ with $\mathtt{m} = 2$ inputs and $\mathtt{p} = 2$ outputs. The input is T samples long, zero mean unit variance white Gaussian sequence and the initial condition x_{ini} is a zero mean random vector. We assume that the

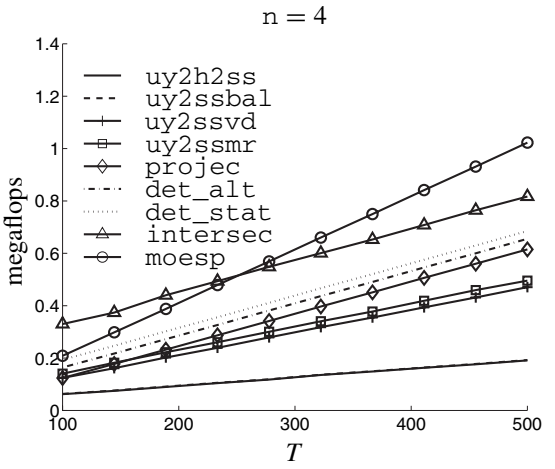

Figure 8.5. *Number of flops for the algorithms as a function of the length T of the given time series.*

true order is known; i.e., $\mathtt{n}_{\max} = \mathtt{n}$ and $\mathtt{l}_{\max}$ is selected as $\lceil \mathtt{n}_{\max}/\mathtt{p} \rceil$. The parameter i (the number of block rows of the Hankel matrix constructed from data) in the subspace identification algorithms is selected as $i = \lceil \mathtt{n}_{\max}/\mathtt{p} \rceil$.

First, we illustrate the amount of work (measured in megaflops) for the compared algorithms as a function of T; see Figure 8.5. The order is chosen as $\mathtt{n} = 4$ and T varies from 100 to 500. The computational complexity of all compared algorithms is linear in T but with different initial cost and different slope. The initial cost and the slope are smallest (almost the same) for `uy2ssbal` and `uy2h2ss`.

The second experiment shows the flops for the compared algorithms as a function of the system order `n`; see Figure 8.6. The length T of the given time series is chosen as 50 and the order `n` is varied from 1 to 18. We deliberately choose T small to show the limitations of the algorithms to identify a system from a finite amount of data. At a certain value of `n`, the graphs in Figure 8.6 stop. The value of `n` where a graph stops is the highest possible order of a system that the corresponding algorithm can identify from the given $T = 50$ data points. (At higher values of `n`, the algorithm either exits with an error message or gives a wrong result.) The flops as a function of `n` are quadratic for all compared algorithms but again the actual number of flops depends on the implementation. Again, most efficient are `uy2ssbal` and `uy2h2ss`. Also, they outperform all other methods except `moesp` in the ability to identify a (high order) system from (small) amount of data. This is a consequence of the fact that Algorithm 8.7 is more parsimonious in the persistency of excitation assumption than Algorithm 8.6.

8.11 Conclusions

We have presented theory and algorithms for exact identification of LTI systems. Although the exact identification problem is not a realistic identification problem, it is interesting and nontrivial from theoretic and algorithmic points of view. In addition, it is an ingredient and

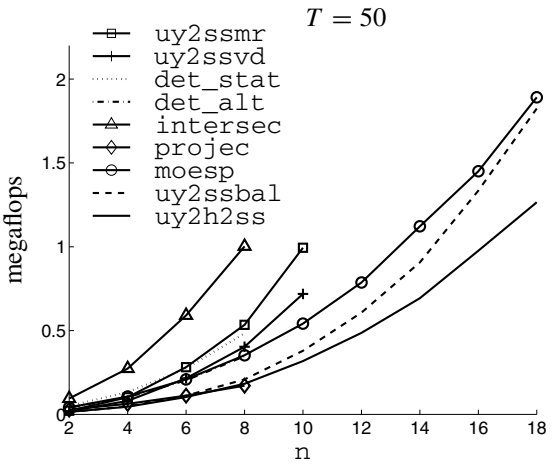

Figure 8.6. *Number of flops for the algorithms as a function of the order* n *of the system.*

prerequisite for proper understanding of other more complicated and realistic identification problems incorporating uncertainty.

The main result is the answer to the identifiability question, Under what conditions, verifiable from w_{d}, does the MPUM coincide with the data generating system? Once this question is answered positively, one can consider algorithms for passing from the data to a representation of the unknown system. In fact, the algorithms compute a representation of the MPUM.

We have presented algorithms for exact identification aiming at kernel, convolution, and input/state/output representations. The latter ones were analyzed in the most detail. We showed links and a new interpretation of the classical MOESP and N4SID deterministic subspace identification algorithms.