Recent progress in structured low-rank approximation

Ivan Markovsky

Department ELEC, Vrije Universiteit Brussel
Pleinlaan 2, Building K, B-1050 Brussels, Belgium
Tel.: +32-2-6292947, Email: imarkovs@vub.ac.be

Abstract

Rank deficiency of a data matrix is equivalent to the existence of an exact linear model for the data. For the purpose of linear static modeling, the matrix is unstructured and the corresponding modeling problem is an approximation of the matrix by another matrix of a lower rank. In the context of linear time-invariant dynamic models, the appropriate data matrix is Hankel and the corresponding modeling problems becomes structured low-rank approximation. Low-rank approximation has applications in: system identification; signal processing, machine learning, and computer algebra, where different types of structure and constraints occur.

This paper gives an overview of recent progress in efficient local optimization algorithms for solving weighted mosaic-Hankel structured low-rank approximation problems. In addition, the data matrix may have missing elements and elements may be specified as exact. The described algorithms are implemented in a publicly available software package. Their application to system identification, approximate common divisor, and data-driven simulation problems is described in the paper and is illustrated by reproducible simulation examples. As a data modeling paradigm the low-rank approximation setting is closely related to the the behavioral approach in systems and control, total least squares, errors-in-variables modeling, principal component analysis, and rank minimization.

Keywords: low-rank approximation, total least squares, structured matrix, variable projection, missing data, system identification, reproducible research, literate programing, noweb.

1 Introduction

Science is aiming at discovering patterns in empirical observations. The simplest and most basic form of a pattern is the linear relation and a convenient data structure for organizing the observations is the two dimensional array of numbers—the matrix. The problem of finding linear relations among the columns or rows of a matrix is closely related to the numerical linear algebra problem of computing the rank of a matrix. Linear relations exist if and only if the matrix is "low-rank", *i.e.*, its rank is less than the smaller of its dimensions.

The problem of discovering more complicated data patterns, such as nonlinear and dynamical relations, can also be posed and solved as a rank computation problem for a matrix obtained from the data via a problem dependent transformation. We call such matrices structured. There is a direct correspondence between types of patterns considered in the applications and types of structured matrices used in the mathematical problem formulation described in the paper. Specific examples illustrating this statement are given in Section 2.

Noise in the data, however, can disguise the exact properties present (or hypothesized) in the noise-free data. This makes the estimation and approximation issues critical in real-life data modeling applications. In Section 2, we show that the problem of discovering *exact patterns* in the data is equivalent to computing the rank of a structured matrix, constructed from the data. In Section 3, we make a transition to the more realistic problem of structured low-rank approximation, where the "best" approximation of the given data by data that is consistent with the desired exact property is aimed at. An effective approach for solving this latter problem is outlined in Section 5. Details about the method in the case of missing data and efficient computations for mosaic-Hankel structured matrices are given in Section 7.

The emphasis in the paper is on applications in systems, control, and signal processing. General introduction to application of the structured low-rank approximation methods developed is given in Section 4. Details about applications in signal processing and numerical examples presented in a literate programming style are given in Section 6.

The examples include sum-of-damped exponentials modeling, using prior knowledge about pole location, data modeling with missing data, and approximate common divisor computation. An interesting application of missing data estimation, shown in Section 6.5, is data-driven simulation and control, *i.e.*, direct computation of a signal of interest (simulated response or feed-forward control) from data without explicit model identification.

Links among structured low-rank approximation and other data modeling approaches—structured total least-squares, behavioral paradigm, errors-in-variables modeling, principal component analysis, and rank minimization—are presented in Section 8. We refer to reader to Section 8, the overview paper [Mar08], and the book [IWVD06], for extensive literature survey and bibliography. This paper is an update of [Mar08] and [MV07] with new applications (Sections 4 and 6), methods for missing data estimation (Section 7.1), fast approximation of mosaic-Hankel-like matrices (Section 7.2), and availability of a software package [MU12a] implementing the theory.

2 Examples of properties equivalent to rank deficiency of a matrix

The general concept outlined in the introduction is illustrated and motivated in this section on five specific examples of data patterns:[1]

E1. N, q-dimensional vectors $d_1, \ldots, d_N$ belong to a subspace of dimension $\mathtt{m} < q$,

E2. N, 2-dimensional vectors $d_1, \ldots, d_N$ belong to a conic section,

E3. the scalar sequence $\big(y(1), \ldots, y(T)\big)$ is a sum of $\mathtt{n} < T/2$ damped exponentials,

E4. the q-dimensional vector sequence $\big(w(1), \ldots, w(T)\big)$ is a trajectory of a bounded complexity linear time-invariant system, and

E5. two polynomials a and b have a common divisor of degree at least ℓ.

Each example is expressed as a rank constraint

$$\operatorname{rank}(D) \leq r, \tag{1}$$

for a suitably chosen matrix D and a natural number r. The matrix D is structured, *i.e.*, it is a function of the data. The observation that as diverse examples as E1–E5 lead to (1) motivates the statement that the low-rank property of a structured matrix constructed from the data is a general property in data modeling.

Each example is next briefly discussed. In the simplest example E1, D is the matrix of the (horizontally) stacked vectors

$$\begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$$

and the upper bound r on the rank of D is equal to the subspace dimension $\mathtt{m}$. Indeed, the equivalence of statement E1 and the rank condition (1) is a basic linear algebra fact. In the case of subspace fitting, the data matrix D is unstructured, *i.e.*, the observed variables d_{ij} enter directly as elements of the data matrix D. As shown next, this is not the case in the more complicated examples.

The notion of rank plays also a central role in the problem of detecting whether the points $\{ d_1, \ldots, d_N \} \subset \mathbb{R}^2$ belong to a conic section $\mathscr{B} \subset \mathbb{R}^2$. Although a conic section $\mathscr{B}$ is a nonlinear curve, it is linearly parameterized

$$\mathscr{B} = \Big\{ d = \begin{bmatrix} a \\ b \end{bmatrix} \;\big|\; \begin{bmatrix} a^2 & ab & a & b^2 & b & 1 \end{bmatrix} \theta = 0 \Big\}, \qquad \text{for some } \theta \neq 0,$$

by monomials a^2, ab, a, b^2, b, and 1 of degree up to 2. We have,

$$\{ d_1, \ldots, d_N \} \subset \mathscr{B} \quad \iff \quad \theta^\top D = 0,$$

[1] In data modeling, a pattern of the data is called a *model* and a collection of related patterns (*e.g.*, linear, conic section, *etc.*) is called a *model class*.

where D is the $6 \times N$ matrix

$$\begin{bmatrix} a_1^2 & a_2^2 & \cdots & a_N^2 \\ a_1 b_1 & a_2 b_2 & \cdots & a_N b_N \\ a_1 & a_2 & \cdots & a_N \\ b_1^2 & b_2^2 & \cdots & b_N^2 \\ b_1 & b_2 & \cdots & b_N \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \tag{2}$$

whose columns are nonlinear transformations of the corresponding data vectors $d_i = \left[\begin{smallmatrix} a_i \\ b_i \end{smallmatrix}\right]$. Thus, statement E2 is equivalent to the rank condition (1) with D given in (2) and $r = 5$.

The matrix (2) is an example of a nonlinear structure. In the rest of the paper, we consider only affinely structured matrices, which allow effective solution methods (see Section 5). As illustrated by examples E3–E5, the affine structure is general enough to cover important classes of applications in systems, control, and signal processing.[2]

A sum-of-damped-exponentials sequence $\big(y(1), \ldots, y(T)\big)$, *i.e.*, a sequence of the form

$$y(t) = c_1 z_1^t + \cdots + c_{\mathtt{n}} z_{\mathtt{n}}^t, \qquad \text{for} \quad t = 1, \ldots, T, \tag{3}$$

where $c_1, \ldots, c_{\mathtt{n}}$ and $z_1, \ldots, z_{\mathtt{n}}$ are given complex numbers, satisfies a difference equation

$$R_0 y(t) + R_1 y(t+1) + \cdots + R_{\mathtt{n}} y(t+\mathtt{n}) = 0, \qquad \text{for} \quad t = 1, \ldots, T - \mathtt{n}, \tag{4}$$

and parameter vector

$$R := \begin{bmatrix} R_0 & R_1 & \cdots & R_{\mathtt{n}} \end{bmatrix} \neq 0$$

The relation (4) is again linear in the parameters, so that the condition for existence of solution is a rank constraint of the type (1). The difference equation (4), however, results in a matrix equation $RD = 0$, with a $(\mathtt{n}+1) \times (T - \mathtt{n})$ Hankel structured matrix D

$$\mathscr{H}_{\mathtt{n}+1, T-\mathtt{n}}(y) := \begin{bmatrix} y(1) & y(2) & y(3) & \cdots & y(T-\mathtt{n}) \\ y(2) & y(3) & \iddots & & y(T-\mathtt{n}+1) \\ y(3) & \iddots & & & \vdots \\ \vdots & & & & \\ y(\mathtt{n}+1) & y(\mathtt{n}+2) & \cdots & & y(T) \end{bmatrix}, \tag{5}$$

i.e., the elements on the antidiagonals of the matrix are equal to each other. The upper bound r on the rank of D is equal to the number $\mathtt{n}$ of damped exponentials.

The sum-of-damped-exponentials model (3) is a special case of a linear time-invariant dynamical system. In the behavioral approach to system theory [Wil87], a dynamical system is defined as a set of trajectories. A system can be specified by different representations. For example, a linear time-invariant system can always be represented by a vector constant coefficients difference equation

$$R_0 w(t) + R_1 w(t+1) + \cdots + R_\ell w(t+\ell) = 0, \qquad \text{for} \quad t = 1, \ldots, T - \ell, \tag{6}$$

where the coefficients $R_i \in \mathbb{R}^{\mathtt{p} \times q}$ are such that the matrix polynomial $R(z) := \sum_{i=0}^{\ell} R_i z^i$ is full row rank. In this case, (6) is a system of $\mathtt{p}$ linearly independent equations.

Using the notation $\mathscr{H}$, defined in (5) for scalar sequences, the difference equation (6) can be written as a matrix equation

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_{\mathtt{n}} \end{bmatrix} D = 0, \qquad \text{where} \quad D = \mathscr{H}_{\ell+1, T-\ell}(w).$$

Note that, in this case, the data matrix D is Hankel with $q \times 1$ block elements (a block Hankel matrix). This shows that statement E4 is also a special case of (1) for D block Hankel and $r = (\ell+1)q - \mathtt{p}$. The bound r on the rank of D bounds the complexity of the model.

[2] As explained in Section 3, the affine structure is still too general for development of practically efficient algorithms. The best compromise between generality and efficiency is the mosaic-Hankel-like structure.

Finally, in the polynomial common divisor example E5, there is a relation

$$\operatorname{rank}(D) \leq r := n_a + n_b - 2\ell + 1$$

between the degree ℓ of a common divisor of the polynomials

$$a(z) = a_0 + a_1 z + \cdots + a_{n_a} z^{n_a} \qquad \text{and} \qquad b(z) = b_0 + b_1 z + \cdots + b_{n_b} z^{n_b}$$

and the rank of the Sylvester sub-resultant matrix [ASÅ04] of a and b:

$$\begin{bmatrix} S_{n_b-\ell+1}(a) \\ S_{n_a-\ell+1}(b) \end{bmatrix}, \qquad \text{where} \quad S_k(a) := \begin{bmatrix} a_0 & a_1 & \cdots & a_{n_a} & & \\ & \ddots & \ddots & & \ddots & \\ & & a_0 & a_1 & \cdots & a_{n_a} \end{bmatrix} \in \mathbb{R}^{k \times (n_a+k)}. \tag{7}$$

(By default, all blank entries in a matrix are zeros.) A summary of the matrices D and their maximum ranks r, in the subspace, conic section, sum-of-damped-exponentials, linear time-invariant dynamical system, and common divisor examples is given in Table 1.

The examples have the following common feature:

> a property of the data is equivalent to the low-rank property of a matrix constructed from the data.

The problem of checking the presence or absence of a property of the data is then reduced to the standard numerical linear algebra problem of computing the rank of a matrix. Existing algorithms and software are, therefore, readily available for solving the original problem. Note that a range of applications are equivalent to a single abstract problem, for which robust and efficient methods and software are readily available. The key step in applying the approach sketched above and further developed in the paper is to find the relevant matrix D and rank constraint r in (1) for the data property of interest.

	example	data matrix D	structure	rank bound r
E1	subspace	$\begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$	unstructured	subspace dimension $\mathtt{m}$
E2	conic section	(2)	polynomial	5
E3	sum of damped exponentials	$\mathscr{H}_{\ell+1,T-\ell}(y)$	Hankel	number of exponents $\mathtt{n}$
E4	linear time-invariant system	$\mathscr{H}_{\ell+1,T-\ell}(w)$	block Hankel	$\ell q + \mathtt{m}$
E5	greatest common divisor	$\begin{bmatrix} S_{n_b-\ell+1}(a) \\ S_{n_a-\ell+1}(b) \end{bmatrix}$	Sylvester	$n_a + n_b - 2\ell + 1$ $\ell =$ GCD degree

Table 1: Summary of data matrices D and rank bounds r in the examples.

3 Structured low-rank approximation

Examples E1–E5 are qualitative statements about properties of the data. In practice the data is often noise corrupted and the properties are generically not satisfied. Even for exact data, in finite precision arithmetic, the problem of computing the rank of a matrix is notoriously difficult and should be avoided.

A more general and more useful problem than the one of establishing the presence or absence of an exact property is the one of finding a quantitative measure of how far the data is from satisfying the property. One way to formalize this problem is to define a distance measure from the data to the subset of the data space where the property of interest is satisfied. As shown above, the subset of the data space for which the property is satisfied is a manifold of low-rank matrices, so that the distance computation problem is a low-rank approximation problem.

For the examples considered, the low-rank approximation problem leads to classical problems: principal component analysis [Hot33] and total least squares [GV80, MV07], in the case of E1; orthogonal regression, also called geometric fitting [GGS94, Nie01], in the case of E2; output error system identification [SS89, Lju99], in the case of E3; errors-in-variables system identification [Söd07], in the case of E4; and approximate common divisor [KL98], in the case of E5. Similarities and differences of the low-rank approximation setting to other modeling paradigms are explained in Section 8 and an overview of applications is given in Section 4. Next, we define formally the low-rank approximation problem.

The general problem considered in the paper is to find the nearest (in some specified sense) matrix $\widehat{D}$ to a given matrix D, where $\widehat{D}$ has rank less than or equal to a specified number r and the same structure as the one of D. A matrix structure is a class of matrices, which can be defined as the image $\{ \mathscr{S}(p) \mid p \in \mathbb{R}^{n_p} \}$ of a function $\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m \times n}$, *e.g.*, the affine structures

$$\mathscr{S}(p) = \sum_{i=1}^{n_p} S_i p_i, \qquad \text{where} \quad S_i \in \mathbb{R}^{m \times n}. \tag{8}$$

An example of affine matrix structure $\mathscr{S}$ is the Hankel structure (5).[3] For example, $\mathscr{H}_2(p) = \left[\begin{smallmatrix} p_1 & p_2 \\ p_2 & p_3 \end{smallmatrix}\right]$ is represented by (8) with

$$S_0 = 0, \quad S_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad S_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \text{and} \quad S_3 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

For a given structure $\mathscr{S}$, a vector $p \in \mathbb{R}^{n_p}$, refered to as the structure parameter vector, specifies a structured matrix $\mathscr{S}(p)$. The distance between matrices with the same structure is expressed equivalently as the distance between their structure parameter vectors. In what follows, we use the weighted (semi-)norm

$$\|p\|_w^2 := \sum_{i=1}^{n_p} w_i p_i^2,$$

defined by the nonnegative vector $w = \begin{bmatrix} w_1 & \cdots & w_{n_p} \end{bmatrix}^\top$.

The structured low-rank approximation problem is defined as follows.

Problem 1 (Structured low-rank approximation problem)**.** Given matrix structure specification $\mathscr{S}$, vector of structure parameters p, nonnegative vector w, and desired rank r, find a structure parameter vector $\widehat{p}$, such that the corresponding matrix $\mathscr{S}(\widehat{p})$ has rank at most r, and is as close as possible to p in the sense of the weighted semi-norm $\| \cdot \|_w$, *i.e.*,

$$\text{minimize} \quad \text{over } \widehat{p} \in \mathbb{R}^{n_p} \quad \|p - \widehat{p}\|_w^2 \quad \text{subject to} \quad \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r. \tag{SLRA}$$

Without loss of generality, it is assumed that $m \leq n$.

A solution method for general affine structure (8) is presented in Sections 5. As illustrated by examples E3–E5, applications in signal processing, system identification, and computer algebra lead to special linearly structured low-rank approximation problems. The block-Hankel and Sylvester structure, appearing in system identification and computer algebra, are instances of a *mosaic-Hankel-like* structure [Hei95]

$$\mathscr{S}(p) := \Phi \mathscr{H}_{\mathbf{m},\mathbf{n}}(p), \tag{9}$$

where $\mathscr{H}_{\mathbf{m},\mathbf{n}}$ is the block matrix with Hankel blocks (see (5))

$$\mathscr{H}_{\mathbf{m},\mathbf{n}}(p) = \begin{bmatrix} \mathscr{H}_{m_1,n_1}(p^{(11)}) & \cdots & \mathscr{H}_{m_1,n_N}(p^{(1N)}) \\ \vdots & & \vdots \\ \mathscr{H}_{m_q,n_1}(p^{(q1)}) & \cdots & \mathscr{H}_{m_q,n_N}(p^{(qN)}) \end{bmatrix}, \qquad \begin{matrix} \mathbf{m} := \begin{bmatrix} m_1 & \cdots & m_q \end{bmatrix} \\ \mathbf{n} := \begin{bmatrix} n_1 & \cdots & n_N \end{bmatrix} \end{matrix} \tag{10}$$

[3] This representation of a Hankel matrix is memory and computation inefficient and is not used in practical implementations.

Φ is a full row rank matrix, and the $p^{(ij)}$'s are subvectors of p. A fast algorithm for mosaic-Hankel low-rank approximation is outlined in Section 7.2.

In regular problems, the weight vector w is positive, however, it is useful to consider also the extreme cases of infinite and zero weights in the approximation criterion, so that, in general, $w \in [0, +\infty) \cup \{+\infty\}$. An infinite weight $w_i = +\infty$ imposes the equality constraint $\widehat{p}_i = p_i$ on the optimization problem (SLRA) and a zero weight $w_i = 0$ makes the parameter p_i irrelevant. Indeed, in case of an infinite weight $w_i = +\infty$, the cost function $\|p - \widehat{p}\|_w$ is infinite unless $\widehat{p}_i = p_i$, so that the infinite weight specifies implicitly a hard constraint of a fixed element in the approximation vector $\widehat{p}$.

In the case of zero weight $w_i = 0$, the error term $p_i - \widehat{p}_i$ is excluded from the cost function, so that p_i does not influence the solution. Parameter values corresponding to zero weights are marked with the symbol `NaN` (not a number) and are considered as missing values.

4 Applications

Many data-driven modeling problems in systems and control, signal processing, and machine learning as well as problems in computer algebra (see Figure 1) can be posed as a structured low-rank approximation problem (SLRA) for suitably defined structure $\mathscr{S}$, weight vector w, and rank constraint r. As illustrated by examples E1–E5, there is a correspondence between the matrix structure $\mathscr{S}$ of the low-rank approximation problem (SLRA) and the model class in the data modeling problems (see Table 2).

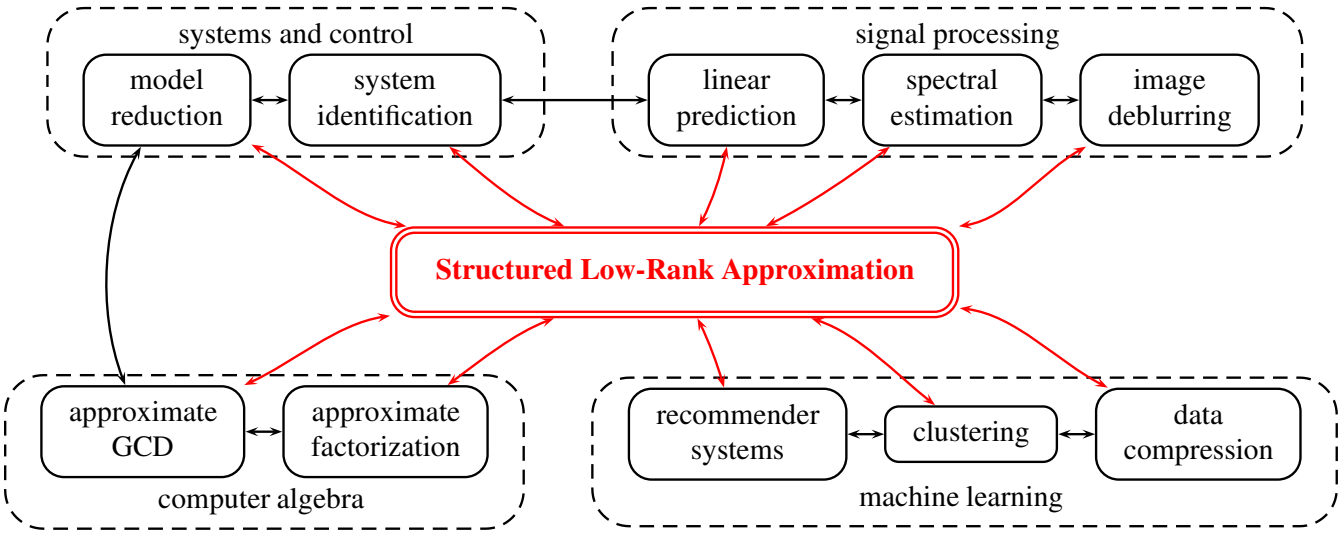


Figure 1: Structured low-rank approximation as a generic problem for data modeling.

Table 2: Correspondence between the matrix structure $\mathscr{S}$ of the low-rank approximation problem (SLRA) and the model class in the data modeling problem.

Structure $\mathscr{S}$	$\leftrightarrow$	Model class
unstructured	$\leftrightarrow$	linear static
Hankel	$\leftrightarrow$	scalar linear time-invariant
$q \times 1$ Hankel	$\leftrightarrow$	q-variate linear time-invariant
$q \times N$ Hankel	$\leftrightarrow$	N equal length trajectories
mosaic-Hankel	$\leftrightarrow$	N general trajectories
$\begin{bmatrix}\text{Hankel} & \text{unstructured}\end{bmatrix}$	$\leftrightarrow$	finite impulse response
$\begin{matrix}\text{block-Hankel} \\ \text{Hankel-block}\end{matrix}$	$\leftrightarrow$	2-dimensional linear shift-invariant

Advantages of reformulating problems as the structured low-rank approximation problem (SLRA) are: 1) links among seemingly unrelated problems are established, 2) generic methods, algorithms, and software can be reused

in different application domains. In order to justify this claim, we apply the generic software for solving (SLRA), described in Section 5 of the paper, to three vastly different applications: system identification, data-driven simulation, and computation of an approximate common divisor of three polynomials.

In system theory, it is well known that there is a connection between model reduction and system identification. At the heart of both problems is the fact (illustrated by example E3) that a time series is a trajectory of a linear time-invariant system of bounded complexity if and only if a Hankel matrix constructed from the time-series is rank deficient. Also, there are known connections among signal processing methods, such as forward-backward linear prediction [KT82] and harmonic retrieval [SM05], on one hand, and system identification methods, such as the prediction error methods [Lju99], on the other hand.

Cross-disciplinary links among computer algebra, machine learning and system theory however are less explored. The low-rank approximation setting offers a platform for establishing such links on a global scale. Indeed, formulating a new problem as a low-rank approximation problem with a specific structure relates this problem to all problems that are known to be equivalent to low-rank approximation with the same type of structured matrix.

Another advantage of formulating a problem as the structured low-rank approximation problem (SLRA) is that a range of different solution methods becomes readily available for that problem. The main types of methods, with a few representatives for each type, are:

- global solution methods (see [UM12, Section 3]),
  - semi-definite programming relaxations of rational function minimization [JdK06],
  - methods based on solving systems of polynomial equations [Ste04];
- local optimization methods,
  - variable projection [MVP05],
  - alternating projections [WAH$^+$97]; and
- heuristics based on convex relaxations,
  - multistage methods [VD96],
  - nuclear norm heuristic [Faz02].

The methods have different properties, which may be advantages or disadvantages in different applications. Multistage methods are, in general, faster but less accurate than the optimization based methods. Local optimization methods require initial approximations. The solution obtained and the amount of computations may be affected strongly by the bad initialization. Global optimization methods are currently impractical for realistic signal processing applications (*e.g.*, large data sets or real-time applications), where the required computational effort is an important consideration.

In our experience the most promising approach for batch model identification is local optimization using an approach similar to the variable projection principle [GP03]. This approach is very effective in data modeling problems where the model complexity (number of inputs and lag or order of the system) is much smaller than the amount of data. This is a typical situation in control and signal processing. The method described in the following section is of the variable projection family and is therefore intended for data approximation by low-complexity models. In Section 6, we show numerical examples with a publicly available software implementation of the method.

5 Variable projection-type algorithms for structured low-rank approximation

The rank constraint in (SLRA) can be expressed as a condition that the dimension of the left kernel of $\mathscr{S}(\widehat{p})$ is at least $m-r$

$$\operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r \quad \Longleftrightarrow \quad \begin{array}{l}\text{there is a full row rank matrix } R \in \mathbb{R}^{(m-r)\times m},\\ \text{such that } R\mathscr{S}(\widehat{p}) = 0.\end{array}$$

Then (SLRA) can be rewritten in the following equivalent form

$$\begin{array}{ll}\text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} \text{ and } R \in \mathbb{R}^{(m-r)\times m} \quad \|p-\widehat{p}\|_w^2\\ \text{subject to} & R\mathscr{S}(\widehat{p}) = 0 \quad \text{and} \quad R \text{ is full row rank.}\end{array} \tag{11}$$

Problem (11) is a nonlinear least squares problem, which, in general, admits no analytic solution. However, the variable $\widehat{p}$ can be eliminated by representing (11) as a double minimization problem:

$$\text{minimize} \quad \text{over full row rank } R \in \mathbb{R}^{(m-r)\times m} \quad M(R), \tag{12}$$

where

$$M(R) := \min_{\widehat{p}} \|p-\widehat{p}\|_w^2 \quad \text{subject to} \quad R\mathscr{S}(\widehat{p}) = 0. \tag{13}$$

The inner minimization problem (13) is a linear least squares problem

$$M(R) = \min_{\widehat{p}} \|\operatorname{diag}(\sqrt{w})(p-\widehat{p})\|_2^2 \quad \text{subject to} \quad G(R)(\widehat{p}-p) = \operatorname{vec}\big(R\mathscr{S}(p)\big), \tag{14}$$

where

$$G(R) := \begin{bmatrix} \operatorname{vec}(RS_1) & \cdots & \operatorname{vec}(RS_{n_p}) \end{bmatrix} \in \mathbb{R}^{(m-r)n\times n_p}.$$

Problem (14) admits an analytic solution, though special care is needed for fixed values ($w_i = +\infty$) and missing values ($w_i = 0$), see Section 7.1. Furthermore, in Section 7.2, it is shown that fast evaluation of M can be performed for mosaic-Hankel-like structured matrices (9).

The remaining problem (12) of minimizing the function M over R, referred to as outer minimization problem, has less optimization variables than problem (11) — $m(m-r)$ versus $n_p + m(m-r)$). In applications, typically m is much smaller than the number of parameters n_p, so that the reduction in number of variables is significant.

The approach, described above, for solving (11) is similar to the variable projection method [GP03] for solution of separable unconstrained non-linear least squares problems

$$\text{minimize} \quad \text{over } \alpha \text{ and } a \quad \|y-\Phi(\alpha)a\|_2^2. \tag{15}$$

The low-rank approximation problem (11), however, is different from the nonlinear least squares problem (15). Instead of the explicit function $\widehat{y} = \Phi(\alpha)a$, where a is unconstrained, an implicit relation $R\mathscr{S}(\widehat{p}) = 0$ is considered, where the variable R is constrained to have full row rank. This fact requires new type of algorithms where the nonlinear least squares problem is an optimization problem on a Grassmann manifold, see [AMS08, AMSD02].

In (12), the cost function M is minimized over the set of full row rank matrices $R \in \mathbb{R}^{(m-r)\times m}$. It can be seen from the definition (13) of $M(R)$ that the cost function depends only on the space spanned by the rows of R:

$$\text{rows of } R' \text{ and } R'' \text{ span the same subspace} \quad \Longrightarrow \quad M(R') = M(R'').$$

Therefore, (12) is a minimization problem on the Grassmann manifold $\operatorname{Gr}(m-r,m)$ of all $(m-r)$-dimensional subspaces of $\mathbb{R}^m$. In order to find a minimum of M, the search space in (12) can be replaced by a set of matrices $R \in \mathbb{R}^{(m-r)\times m}$ that represent all subspaces from $\operatorname{Gr}(m-r,m)$, *e.g.*, matrices satisfying the constraint

$$RR^\top = I_{m-r}. \tag{16}$$

In the software package of [MU12a], problem (SLRA) is solved numerically by a function with interface

```
[ph, info] = slra(p, s, r, opt)
```

The compulsory input arguments `p`, `s`, and `r` correspond to the vector of the structure parameters p, the structure specification $\mathscr{S}$, and the bound r on the rank of the approximating matrix, respectively. The output argument `ph` is a locally optimal solution $\widehat{p}$ of (SLRA). The argument `s` is a structure with fields `m`, `n`, `phi`, and `w`. The fields `m` and `n` are the vectors **m** and **n** that specify the mosaic-Hankel matrix (10), and `phi` is the Φ matrix (default value I). The vector `w` is the weight vector $w \in \mathbb{R}^{n_p}$ (default value is vector of all ones).

The arguments `opt` and `info` contain input options and output information for the optimization solver, respectively. For example, the optional field `opt.Rini` is used as an initial approximation for the optimization method. The minimizer $\widehat{R}$ of (14) is provided in the field `info.Rh`.

In the software package, problem (12) is solved by local minimization over parameters $\theta \in \mathbb{R}^{n_\theta}$, such that $R = \mathscr{R}(\theta)$

$$\text{minimize} \quad \text{over } \theta \in \mathbb{R}^{n_\theta} \quad M\big(\mathscr{R}(\theta)\big) \quad \text{subject to} \quad \mathscr{R}(\theta)\mathscr{R}^\top(\theta) = I_{m-r}.$$

The function $\mathscr{R}$ allows specification of linear structure of the matrix R

$$R = \mathscr{R}(\theta) := \mathrm{vec}_{m-r}^{-1}(\theta\Psi), \tag{17}$$

where $\mathrm{vec}^{-1}(\cdot)$ is the inverse of the column-wise vectorization operator and $\Psi \in \mathbb{R}^{n_\theta \times (m-r)m}$ is a full row-rank matrix that is passed by the optional parameter `opt.Psi` (default value I).

6 Numerical examples

In this section, we describe the application of (SLRA) to system identification (Section 6.1–6.4), data-driven simulation (Section 6.5), and approximate common divisor computation (Section 6.6). Numerical examples with the variable projection method, described in the paper and implemented in [MU12a], are presented using a literate programing style [Knu92]. Matlab is the programming language and `noweb` [Ram94] is the literate programming tool used.

6.1 Autonomous system identification

In system theoretic and signal processing terminology, the difference equation (4) defines an autonomous linear time-invariant dynamical model. The problem of finding the model from observed trajectory of the system is called a system identification problem [Lju99, SS89]. The particular identification problem considered in this section is defined as follows: Given a time series

$$y = \big(y(1), \ldots, y(T)\big) \in \mathbb{R}^T$$

and a natural number ℓ,

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{y} \in \mathbb{R}^T \text{ and } \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \neq 0 \quad \|y - \widehat{y}\|_2^2 \\ \text{subject to} & R_0\widehat{y}(t) + R_1\widehat{y}(t+1) + \cdots + R_\ell\widehat{y}(t+\ell) = 0, \quad \text{for } t = 1, \ldots, T-\ell. \end{array} \tag{18}$$

The constraints of (18) are equivalent to

$$\mathrm{rank}\,\big(\mathscr{H}_{\ell+1,T-\ell}(\widehat{y})\big) \leq \ell,$$

so that, (18) is equivalent to the Hankel low-rank approximation problem

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{y} \in \mathbb{R}^T \quad \|y - \widehat{y}\|_2^2 \\ \text{subject to} & \mathrm{rank}\,\big(\mathscr{H}_{\ell+1,T-\ell}(\widehat{y})\big) \leq \ell, \end{array} \tag{19}$$

which is a special case of (SLRA) with

$$\mathbf{m} = \ell + 1 \qquad \text{and} \qquad \mathbf{n} = T - \ell.$$

The problem is solved numerically with a call `slra(y, s, r)` to the `slra` solver, with input arguments

⟨`slra` *arguments for scalar Hankel problem*⟩≡
```
s.m = ell + 1; s.n = T - ell; r = ell;
```

and `y` being the vector of the consecutive output samples $y(1), \ldots, y(T)$.

Numerical example

The sequence $\bar{y}$ (`y0` in the Matlab code), shown in Figure 2, left, (solid line) is an exact trajectory of an autonomous linear time-invariant system with lag $\ell = 6$ (`ell` in the Matlab code).

⟨*example autonomous SYSID*⟩≡
```
clear all; randn('seed', 0); load aut_sys_traj; opt.solver = 'c';
```

Indeed, the left kernel of the Hankel matrix $\mathscr{H}_{\ell+1,T-\ell}(\bar{y})$ has dimension equal to one:

⟨*example autonomous SYSID*⟩+≡
```
size(null(blkhank(y0, ell + 1)')', 1)
```

and a basis vector $\bar{R}$ (`R0` in the Matlab code) for the left kernel of $\mathscr{H}_{\ell+1,T-\ell}(\bar{y})$ gives a difference equation representation (4) with lag ℓ of the exact model for $\bar{y}$. The mapping $\bar{y} \mapsto \bar{R}$ (an exact identification problem) is implemented in Matlab by the function `y2R`.

⟨*example autonomous SYSID*⟩+≡
```
y2R = @(y, ell) null(blkhank(y, ell + 1)')'; R0 = y2R(y0, ell);
```

In the simulation example, however, the identification data $y = \bar{y} + \widetilde{y}$ is corrupted by zero mean white Gaussian noise $\widetilde{y}$ with standard deviation `nl`, multiplied by the empirical standard deviation of $\bar{y}$.

⟨*example autonomous SYSID*⟩+≡
```
nl = 0.4; y = y0 + nl * std(y0) * randn(T, 1);
```

This makes the data matrix $\mathscr{H}_{\ell+1,T-\ell}(y)$ full rank.

⟨*example autonomous SYSID*⟩+≡
```
rank(blkhank(y, ell + 1))
```

Therefore, an exact model does not exists. A heuristic method for approximate autonomous system identification is Kung's method [Kun78], implemented in the function `h2ss` [Mar12, Section 3.1].

⟨*define* `y2R_appr`⟩≡
```
y2R_appr = @(y, ell) fliplr(poly(eig(h2ss(y, ell))));
```

The model obtained by Kung's method is suboptimal in the output error criterion $\|y - \widehat{y}\|$ and is used as an initial approximation for the structured low-rank optimization.

⟨*example autonomous SYSID*⟩+≡
```
⟨define y2R_appr⟩, opt.Rini = y2R_appr(y, ell);
```

Figure 2, left, shows the optimal fit (dashed line) obtained by the `slra` function and the noisy data (dotted line).

⟨*example autonomous SYSID*⟩+≡
```
⟨slra arguments for scalar Hankel problem⟩
[yh, info] = slra(y, s, r, opt);
plot(yh, 'b-'), hold on, plot(y, 'k:'), plot(y0, 'r-')
```

⟨*example autonomous SYSID*⟩+≡
```
ax = axis; axis([1 T ax(3:4)]), print_fig('slra-f2')
```

In order to validate the predictive ability of the identified model, we forecast the next $T_{\text{val}} = 100$ samples

$$y_{\text{v}} := \big(y_{\text{v}}(T+1), \ldots, y_{\text{v}}(T+T_{\text{v}})\big) \qquad (\texttt{yv} \text{ in the Matlab code})$$

of the time series y, shown in Figure 2, right (solid line).

⟨*example autonomous SYSID*⟩+≡
```
R = info.Rh; yini = yh(end - ell + 1:end); ⟨forecast⟩
figure, plot(yvh, 'b-'), hold on, plot(yv(3:end), 'r-')
```

The prediction $\widehat{y}_{\text{v}}$ (`yvh` in the Matlab code) of the model is obtained by iterating the recursion (4) with initial conditions $\widehat{y}(T-\ell+1), \ldots, \widehat{y}(T)$.

⟨*forecast*⟩≡
```
X = - R(1:ell) / R(end); yvh = yini;
for t = 1:Tv
  yvh = [yvh; X * yvh(end - ell + 1:end)];
end
yvh = yvh(ell + 1:end);
```

Figure 2, right, shows the fit of the validation data (dashed line) and the prediction (dashed-dotted line) by a model obtained in the next section.

⟨*example autonomous SYSID*⟩+≡
```
ax = axis; axis([1 T ax(3:4)]), print_fig('slra-f2v')
```

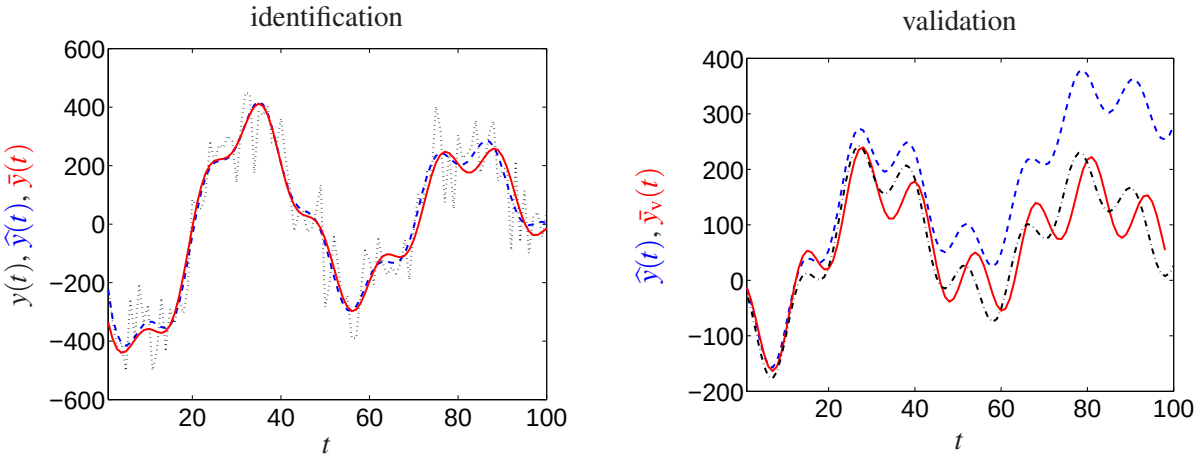


Figure 2: Autonomous system identification example.

6.2 Autonomous system identification with fixed poles

The complex numbers $z_1, \ldots, z_{\mathrm{n}}$ in the sum-of-damped-exponentials model (3) describe the behavior of the model and are called *poles*. The poles of an autonomous linear time-invariant model, defined by a difference equation (4), are equal to the roots of the polynomial

$$R(z) := R_0 + R_0 z + \cdots + R_\ell z^\ell.$$

An $\ell+1$ dimensional vector $R = \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix}$ corresponds to a degree ℓ polynomial $R(z)$ and, vice verse, a polynomial of degree ℓ corresponds to the $\ell+1$ dimensional vector formed of its coefficients in, say, increasing order of the degree.

Consider the Hankel low-rank approximation problem (19) and let R be a basis for the left kernel of the approximating matrix $\mathscr{H}_{\ell+1,T-\ell}(\widehat{y})$. Denote by $z_1, \ldots, z_\ell$ the zeros of $R(z)$. Since R is real, the zeros appear in complex conjugate pairs. Otherwise, the poles $z_1, \ldots, z_\ell$ are unconstrained by the approximation problem (19). In this section, we impose a constraint that some zeros of $R(z)$ are fixed at predefined locations $z_{\mathrm{f},1}, \ldots, z_{\mathrm{f},\ell_{\mathrm{f}}}$ in the complex plane (observing the complex conjugate symmetry) and the other zeros are unconstrained. The fixed zeros define a real polynomial

$$R_{\mathrm{f}}(z) := \prod_{i=1}^{\ell_{\mathrm{f}}} (z - z_{\mathrm{f},i})$$

and the free zeros define a polynomial $\theta(z)$ of degree $\ell - \ell_{\mathrm{f}}$. With this notation, the fixed zeros constraint is

$$R(z) = \theta(z) R_{\mathrm{f}}(z),$$

or in matrix notation

$$R = \theta S_{n_\theta}(R_{\mathrm{f}}), \qquad \text{where} \quad n_\theta = \ell - \ell_{\mathrm{f}} + 1.$$

The resulting Hankel low-rank approximation problem with fixed poles is of the form (17), with the Ψ matrix being the polynomial multiplication matrix $S_{n_\theta}(R_{\mathrm{f}})$, see (7) (constructed in Matlab by the function `mult_mat`).

⟨`opt.psi` *for fixed poles*⟩≡

```
opt.psi = mult_mat(poly(zf), 1, ell - length(zf) + 1);
```

Numerical example

As a numerical example of identification with fixed poles, consider the problem of trend estimation. The data y is modeled as the sum of a ramp function and a trajectory of an autonomous linear time-invariant system. The full model is an autonomous linear time-invariant system with a double pole at one. Therefore, it has $\ell - 2$ free poles and two poles fixed at one.

The true time series $\bar{y}$ considered in the previous section is generated as a sum of a ramp function and a trajectory of a 4th order autonomous linear time-invariant system. Using this prior knowledge, we obtain a better approximation of the true data generating system.

⟨*example autonomous SYSID*⟩+≡
```
zf = [1 1]; ⟨opt.psi for fixed poles⟩
[yh_f, info_f] = slra(y, s, ell, opt);
```

The identified model is validated by predicting the validation data y_{val}.

⟨*example autonomous SYSID*⟩+≡
```
R = info_f.Rh; yini = yh_f(end - ell + 1:end); ⟨forecast⟩, plot(yvh, 'k-.')
```

Figure 2, right, shows the result obtained (dashed-dotted line). Although the fixed double pole at one leads to worse fit of the identification data, it improves the fit of the validation data.

⟨*example autonomous SYSID*⟩+≡
```
ax = axis; axis([1 T ax(3:4)]), print_fig('slra-f2v')
```

6.3 Autonomous system identification with missing data

A block of ℓ or more missing values splits the data into two independent datasets. With $N-1$ such blocks of missing data the problem is equivalent to identification from N datasets. Note, however, that in general the datasets have different number of samples $T_1, \ldots, T_N$. System identification from multiple datasets with different lengths is a mosaic-Hankel low-rank approximation problem with

$$\mathbf{m} = \ell \qquad \text{and} \qquad \mathbf{n} = \begin{bmatrix} T_1 - \ell & \cdots & T_N - \ell \end{bmatrix}.$$

In the more general case of arbitrary distributed missing values, the identification problem is posed as a weighted Hankel low-rank approximation problem. Let $\mathscr{I}_{\text{m}}$ be the indices of the missing samples and $\overline{\mathscr{I}}_{\text{m}}$ be the indices of the given samples. The appropriate weight vector is:

$$w \in \mathbb{R}^T, \qquad \text{with} \quad w(\mathscr{I}_{\text{m}}) = 0 \quad \text{and} \quad w(\overline{\mathscr{I}}_{\text{m}}) = 1.$$

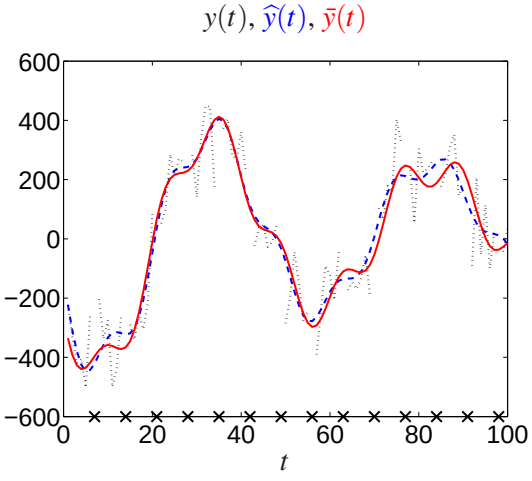
Numerical example

The following experiment shows identification with periodically missing data, in which case standard identification methods are not applicable.

⟨*example autonomous SYSID*⟩+≡
```
⟨slra arguments for scalar Hankel problem⟩
Im = (ell + 1):(ell + 1):T; opt.psi = [];
y(Im) = NaN; ⟨s.w for missing values⟩
[yh_m, info_m] = slra(y, s, r, opt);
```

missing data locations	—	crosses
noisy trajectory	—	dotted black
optimal approximation	—	dashed blue
true trajectory	—	solid red

⟨*example autonomous SYSID*⟩+≡
```
figure(3), plot(yh_m, 'b-'), hold on,
plot(y, 'k:'), plot(y0, 'r-')
ax = axis;
plot(Im, ax(3) * ones(size(Im)), ...
     'Xk', 'markersize', 15)
axis(ax), print_fig('slra-f3')
```



6.4 Identification of an input/output system

The examples presented so far illustrate different aspects of the scalar autonomous linear time-invariant system identification problem (example E3 from the introduction). Next, we consider the general case of a dynamical system with inputs (example E4). In (6), $\mathtt{m} := q - \mathtt{p}$ variables (elements of w) are free, *i.e.*, unrestricted by the model. In system theory, the free variables are called inputs and the remaining variables outputs of the system. Although the separation of the variables into inputs and outputs is, in general, not unique, the number of inputs is invariant of the input/output partition. The pair of integers $(\ell, \mathtt{m})$—lag ℓ of the equation (6) and number of inputs $\mathtt{m}$—describes the complexity of the system.

Given a time series

$$w = \big(w(1), \ldots, w(T)\big) \in (\mathbb{R}^q)^T$$

and a complexity, specified by a pair of natural numbers $(\mathtt{m}, \ell)$,

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{w} \in (\mathbb{R}^q)^T \text{ and } \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \neq 0 \quad \|w - \widehat{w}\|_2^2 \\ \text{subject to} & R_0\widehat{w}(t) + R_1\widehat{w}(t+1) + \cdots + R_\ell\widehat{w}(t+\ell) = 0, \quad \text{for } t = 1, \ldots, T-\ell. \end{array} \tag{20}$$

In the special case of $\mathtt{m} = 0$, (20) reduces to the output-only identification problem (18) and in the special case of $\ell = 0$, (20) reduces to static modeling (example E1). This seamless transition from one problem to another is an important advantage of the adopted framework.

Although, from the application point of view, the input-output identification problem is a significant generalization of the output-only identification problem, from the numerical linear algebra point of view, it is a trivial one. The general identification problem (20) is also a Hankel low-rank approximation problem:

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{w} \in (\mathbb{R}^q)^T \quad \|w - \widehat{w}\|_2^2 \\ \text{subject to} & \text{rank}\big(\mathscr{H}_{\ell+1, T-\ell}(\widehat{w})\big) \leq \ell. \end{array} \tag{21}$$

More specifically, problem (21) is a special case of (SLRA) with

$$\mathbf{m} = \underbrace{\begin{bmatrix} \ell+1 & \cdots & \ell+1 \end{bmatrix}}_{q} \qquad \text{and} \qquad \mathbf{n} = T - \ell,$$

so that it can be solved numerically with a call `slra(w, s, r)` to the `slra` solver, with input arguments

⟨*structure specification for input/output identification*⟩≡

```
s.m = (ell + 1) * ones(q, 1); s.n = T - ell; r = ell * q + m;
```

and `w` set equal to the vector of consecutive output samples $w(1), \ldots, w(T)$.

Numerical example

A linear time-invariant system, with an input/output partition of the variables $w = (u, y)$, can be represented by the difference equation

$$P_0 y(t) + P_1 y(t+1) + \cdots + P_\ell y(t+\ell) = Q_0 u(t) + Q_1 u(t+1) + \cdots + Q_\ell u(t+\ell). \tag{22}$$

We consider a simulation example with the second order single-input single-output system with parameters

$$\bar{Q} = \begin{bmatrix} 1 & -1 & 1 \end{bmatrix} \qquad \text{and} \qquad \bar{P} = \begin{bmatrix} 0.81 & -1.456 & 1 \end{bmatrix}.$$

⟨*data generating system*⟩≡

```
Q0 = [1 -1 1]; P0 = [0.81 -1.456 1];
```

The identification data $w = (u, y)$ is a random trajectory of the system, perturbed by additive noise. The Control Toolbox [Mat] of Matlab is used to simulate a response of the model.

⟨*simulate w*⟩≡

```
sys0 = tf(fliplr(Q0), fliplr(P0), 1);
u0 = rand(T, m); y0 = lsim(sys0, u0); E = rand(T, q);
w  = [u0 y0] + nl * E / norm(E, 'fro') * norm([u0 y0], 'fro');
```

The identified model is obtained from the parameter $\widehat{R}$ as follows:

⟨$\widehat{R} \mapsto (22)$⟩≡
```
Qh = fliplr(info.Rh(1:ell + 1)); Ph = -fliplr(info.Rh(ell + 2:end));
sysh = tf(Qh, Ph, 1);
```

The true data generating system and the identified system are compared by plotting their step responses:

⟨*compare the step responses of* `sys0` *and* `sysh`⟩≡
```
s0 = step(sys0, T2 - ell); sh = step(sysh, T2 - ell);
```

⟨*compare the step responses of* `sys0` *and* `sysh`⟩+≡
```
plot(sh(2:end), '-b'), hold on, plot(s0(2:end), '-r')
ax = axis; axis([1, T2 - 1, ax(3:4)]), print_fig('f-sysid')
```

In a specific example with noise standard deviation 0.05 the result is shown in Figure 3.

⟨*input/output identification example*⟩≡
```
clear all, randn('seed', 0); opt.solver = 'c';
```
⟨*data generating system*⟩ `ell = 2; m = 1; q = 2; T = 100; T2 = 40; nl = 0.05;`
⟨*simulate w*⟩
⟨*structure specification for input/output identification*⟩
```
[wh1, info] = slra(vec(w), s, q * ell + 1);
```
⟨$\widehat{R} \mapsto (22)$⟩
⟨*compare the step responses of* `sys0` *and* `sysh`⟩

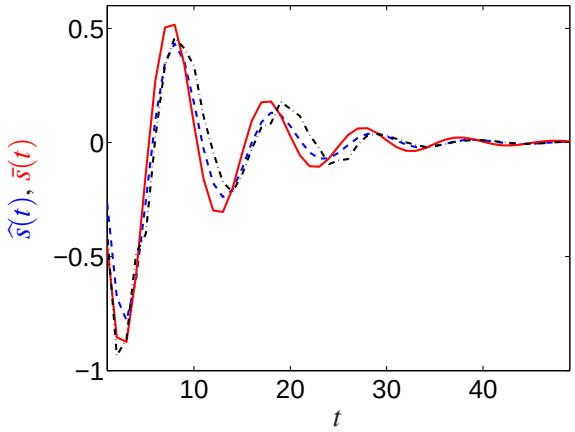


Figure 3: Comparison of the true $\bar{s}(t)$ (solid line) and identified $\widehat{s}(t)$ model step responses by the suboptimal (dashed-dotted) and optimal (dashed) methods.

6.5 Data-driven simulation

The numerical example in the previous section shows model based simulation: first, a model is identified from data and, second, the model's step response is simulated. In this section, we show how an arbitrary response of an unknown system can be obtained directly from (noisy) data of the system without identifying the model in an intermediate step.

Consider two input/output trajectories

$$w' = \begin{bmatrix} u' \\ y' \end{bmatrix} \in (\mathbb{R}^q)^{T'} \qquad \text{and} \qquad w'' = \begin{bmatrix} u'' \\ y'' \end{bmatrix} \in (\mathbb{R}^q)^{T''}$$

of an unknown linear time-invariant system $\mathscr{B}$ with lag ℓ. The trajectory w' specifies implicitly the system. The trajectory w'' is specified by its first ℓ samples (initial conditions)

$$w''_{\mathrm{p}} = \big(w''(1), \ldots, w''(\ell)\big)$$

and its input component

$$u''_{\mathrm{f}} = \big(u''(\ell+1), \ldots, u''(T'')\big).$$

The problem is to find the output of $\mathscr{B}$

$$y''_{\mathrm{f}} = \big(y''(\ell+1), \ldots, y''(T'')\big),$$

which corresponds to the given initial conditions and input, directly from the data without identifying the system $\mathscr{B}$.

This problem is a mosaic-Hankel structured low-rank approximation problem

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{w}' \text{ and } y''_{\mathrm{f}} \quad \|w' - \widehat{w}'\|_2^2 \\ \text{subject to} & \operatorname{rank}\Big(\begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{w}') & \mathscr{H}_{\ell+1}(w'') \end{bmatrix}\Big) \leq 2\ell+1, \end{array} \tag{23}$$

with exact data, the initial conditions w''_{p} and the input u''_{f}, and with missing data, the to-be-simulated response y''_{f}.

⟨*structure specification for data-driven simulation*⟩≡
```
s.m = [ell + 1, ell + 1]; s.n = [T - ell, T2 - ell];
s.w = [ones(q * T, 1); inf * vec(~isnan(w2))];
```

⟨*extract the estimated response $\widehat{y}''_{\mathrm{f}}$ from $\widehat{w}''$*⟩≡
```
sh = wh(T * q + T2 * m + ell + 1:end);
```

Numerical examples

We use the same simulation setup as in Section 6.4. The to-be-simulated trajectory w'' is the step response s of the system, *i.e.*, the response under zero initial conditions and step input:

$$u'' = (\underbrace{0,\ldots,0}_{\ell},\underbrace{1,1,\ldots,1}_{\text{step input}}), \qquad \text{and} \qquad y'' = \big(\underbrace{0,\ldots,0}_{\ell},\underbrace{s(1),s(2),\ldots,s(T''-\ell)}_{\text{step response}}\big).$$

⟨w'' *step response*⟩≡
```
u2 = [zeros(ell, 1); ones(T2 - ell, 1)];
y2 = [zeros(ell, 1); NaN * ones(T2 - ell, 1)]; w2 = [u2 y2];
```

The step response $\widehat{s}$, estimated by solving problem (23) is equal (up to numerical errors) to the one obtained by the model-based simulation procedure in Section 6.4.

⟨*data-driven simulation example*⟩≡
```
clear all, close all, randn('seed', 0), rand('seed', 0)
⟨data generating system⟩ ell = 2; m = 1; q = 2; T = 100; T2 = 40; nl = 0.05;
⟨simulate w⟩
⟨w'' step response⟩
⟨structure specification for data-driven simulation⟩
opt.solver = 'm'; [wh, info] = slra([vec(w); vec(w2)], s, q * ell + 1, opt);
⟨extract the estimated response ŷ''_f from ŵ''⟩
s0 = step(sys0, T2 - ell - 1);
plot(sh(2:end), '-b'), hold on, plot(s0(2:end), '-r')
ax = axis; axis([1, T2 - ell - 1, ax(3:4)]), print_fig('slra-ext-f2')
```

6.6 Approximate greatest common divisor of N polynomials

Our last example is about polynomials computations with inexact coefficients. Consider N polynomials $p^1,\ldots,p^N$ of degrees $n_1,\ldots,n_N$, respectively. Similarly to the two polynomials case in example E5, the degree of the greatest common divisor of $p^1,\ldots,p^N$ can be expressed as

$$\operatorname{degree}\big(\gcd(p^1,\ldots,p^N)\big) = n_1 + \cdots + n_N - \operatorname{rank}\big(S_0(p^1,\ldots,p^N)\big)$$

where

$$S_\ell(p^1,\ldots,p^N) := \begin{bmatrix} S_{n_1-\ell}(p^2) & \cdots & S_{n_1-\ell}(p^N) \\ S_{n_2-\ell}(p^1) & & \\ & \ddots & \\ & & S_{n_N-\ell}(p^1) \end{bmatrix}. \tag{24}$$

is the generalized (multipolynomial) Sylvester sub-resultant matrix [ASÅ04, KYZ06] for $p^1, \ldots, p^N$ with parameter ℓ. The structured matrix $S_\ell(p^1, \ldots, p^N)$ is a mosaic-Hankel-like matrix $\Phi\mathscr{H}_{\mathbf{m},\mathbf{n}}$ with specification of exact (zero) elements. For example, with $N = 3$ and $n_1 = n_2 = n_3 = 2$,

$$\Phi = \begin{bmatrix} I_4 & & \\ & 0_2 & I_2 \end{bmatrix} \qquad \text{and} \qquad \mathscr{H}_{[2\ 6],6} = \begin{bmatrix} \begin{bmatrix} S_1(p^2) & S_1(p^3) \end{bmatrix} \\ S_6(p^1) \end{bmatrix}.$$

The approximate common divisor problem for N polynomials is defined as follows:

$$\begin{aligned} \text{minimize} \quad & \text{over } \widehat{p}^1 \in \mathbb{R}^{n_1+1}, \ldots, \widehat{p}^N \in \mathbb{R}^{n_N+1} \quad \left\| \begin{bmatrix} p^1 \\ \vdots \\ p^N \end{bmatrix} - \begin{bmatrix} \widehat{p}^1 \\ \vdots \\ \widehat{p}^N \end{bmatrix} \right\|_2^2 \\ \text{subject to} \quad & \text{degree}\left(\gcd(\widehat{p}^1, \ldots, \widehat{p}^N)\right) \geq \ell \end{aligned}$$

and is equivalent to the low-rank approximation problem with rank constraint

$$\text{rank}\left(S_\ell(p^1, \ldots, p^N)\right) \leq n_1 + \cdots + n_N - N\ell - 1.$$

⟨`slra` *arguments for approximate GCD of 3 polynomials*⟩≡
```
s.m = [n1 - ell + 1; n1 + n2 + n3 - 2 * ell + 2];
s.n = 2 * n1 + 2 + n3 - 2 * ell + 2;
s.phi = eye(sum(s.m));
s.phi((s.m(1) + n2 - ell + 2):(s.m(1) + n2 - ell + 1 + n1), :) = [];
z1 = inf * ones(n1 - ell, 1); z2 = inf * ones(s.m(2) - 1, 1);
s.w = [z1; ones(n2 + 1, 1); z1; ones(n3 + 1, 1);
       z1; z2; ones(n1 + 1, 1); z2];
r = s.m(1) + n2  + n3 - 3 * ell + 2;
p = zeros(size(s.w)); p(s.w == 1) = [p1 p2 p3]';
```

Numerical examples

Consider the polynomials

$$\begin{aligned} p^1(z) &= 5 - 6z + z^2 = (1-z)(5-z), \\ p^2(z) &= 5.72 - 6.3z + z^2 = (1.1-z)(5.2-z), \\ p^3(z) &= 6.48 - 6.6z + z^2 = (1.2-z)(5.4-z). \end{aligned}$$

In the example, we are looking for an approximate common divisor of degree 1.

⟨*example GCD*⟩≡
```
clear all, n1 = 2; n2 = 2; n3 = 2; ell = 1;
p1 = conv([1   -1], [5   -1]);
p2 = conv([1.1 -1], [5.2 -1]);
p3 = conv([1.2 -1], [5.4 -1]);
```
⟨`slra` *arguments for approximate GCD of 3 polynomials*⟩
```
opt.solver = 'm'; [ph, info] = slra(p, s, r, opt);
```

The computed approximation polynomials

$$\begin{aligned} \widehat{p}^1(z) &= 4.9989 - 6.0057z + 0.9705z^2, \\ \widehat{p}^2(z) &= 5.7200 - 6.2999z + 1.0004z^2, \\ \widehat{p}^3(z) &= 6.4811 - 6.5944z + 1.0289z^2 \end{aligned}$$

have a common root 0.1924.

⟨*example GCD*⟩+≡
```
ph123 = ph(s.w == 1);
r_ph1 = roots(ph123(1:n1 + 1))
r_ph2 = roots(ph123(n1 + 2:n1 + n2 + 2))
r_ph3 = roots(ph123(n1 + n2 + 3:end))
```

7 Algorithmic details

7.1 Analytical solution of the inner minimization problem

In this section, we use the following additional notation.

- $A_{\mathscr{I},\mathscr{J}}$ is the submatrix of A with rows in $\mathscr{I}$ and columns in $\mathscr{J}$ (the row/column index can be replaced by the symbol ":", in which case all rows/columns are selected).
- n_{m} is the number and $\mathscr{I}_{\mathrm{m}} := \{\, i \mid w_i = 0 \,\}$ is the set of missing values.
- n_{f} is the number and $\mathscr{I}_{\mathrm{f}} := \{\, i \mid w_i = +\infty \,\}$ is the set of fixed values.
- $\mathscr{I}_{\mathrm{mf}} := \mathscr{I}_{\mathrm{m}} \cup \mathscr{I}_{\mathrm{f}}$ is the set of missing and fixed values, and $\overline{\mathscr{I}} := \{1,\dots,n_p\} \setminus \mathscr{I}$.
- $W_{\mathrm{r}} := \operatorname{diag}(w_{\overline{\mathscr{I}}_{\mathrm{mf}}})$ is the part of the weight matrix related to given non-fixed data.
- A^{+} is the pseudoinverse of A and $A^{\perp}$ is a matrix which rows form a basis for the left null space of A.

Consider the inner minimization problem (13). Using the change of variables

$$\Delta p_{\mathrm{r}} := p_{\overline{\mathscr{I}}_{\mathrm{mf}}} - \widehat{p}_{\overline{\mathscr{I}}_{\mathrm{mf}}}$$

and (14), we have

$$R\mathscr{S}(\widehat{p}) = 0 \qquad \Longleftrightarrow \qquad \begin{bmatrix} G_{:,\overline{\mathscr{I}}_{\mathrm{mf}}} & G_{:,\mathscr{I}_{\mathrm{f}}} & G_{:,\mathscr{I}_{\mathrm{m}}} \end{bmatrix} \begin{bmatrix} p_{\overline{\mathscr{I}}_{\mathrm{mf}}} - \Delta p_{\mathrm{r}} \\ p_{\mathscr{I}_{\mathrm{f}}} \\ \widehat{p}_{\mathrm{m}} \end{bmatrix} = 0.$$

Therefore, (13) is equivalent to the generalized least norm problem [Pai79]

$$\begin{aligned} M(R) := \min_{\Delta p_{\mathrm{r}},\, \widehat{p}_{\mathrm{m}}} \ & \|\Delta p_{\mathrm{r}}\|^2_{W_{\mathrm{r}}} \\ \text{subject to} \quad & \begin{bmatrix} G_{:,\overline{\mathscr{I}}_{\mathrm{mf}}} & G_{:,\mathscr{I}_{\mathrm{m}}} \end{bmatrix} \begin{bmatrix} \Delta p_{\mathrm{r}} \\ -\widehat{p}_{\mathrm{m}} \end{bmatrix} = G_{:,\overline{\mathscr{I}}_{\mathrm{m}}} p_{\overline{\mathscr{I}}_{\mathrm{m}}}. \end{aligned} \tag{25}$$

The following theorem is adapted from [MU12b] and gives the solution to (25).

Theorem 2. *Under the following assumptions:*

1. $G_{:,\mathscr{I}_{\mathrm{m}}}$ *is full column rank,*
2. $1 \le (m-r)n - n_{\mathrm{m}} \le n_p - n_{\mathrm{f}} - n_{\mathrm{m}}$, *and*
3. $G_{\mathrm{r}} := G^{\perp}_{:,\mathscr{I}_{\mathrm{m}}} G_{:,\overline{\mathscr{I}}_{\mathrm{mf}}}$ *is full row rank,*

problem (25) has a unique minimum,

$$M(R) = \Delta p_{\mathrm{r}}^{\top} W_{\mathrm{r}} \Delta p_{\mathrm{r}} = h^{\top} \left(G_{\mathrm{r}} W_{\mathrm{r}}^{-1} G_{\mathrm{r}}^{\top} \right)^{-1} h, \quad \textit{where} \quad h := G^{\perp}_{:,\mathscr{I}_{\mathrm{m}}} G_{:,\overline{\mathscr{I}}_{\mathrm{m}}} p_{\overline{\mathscr{I}}_{\mathrm{m}}}. \tag{26}$$

The minimum is attained by

$$\widehat{p}_{\mathrm{m}} = -G^{+}_{:,\mathscr{I}_{\mathrm{m}}} \left(G_{:,\overline{\mathscr{I}}_{\mathrm{m}}} p_{\overline{\mathscr{I}}_{\mathrm{m}}} - G_{:,\overline{\mathscr{I}}_{\mathrm{mf}}} \Delta p_{\mathrm{r}} \right) \quad \textit{and} \quad \Delta p_{\mathrm{r}} = W_{\mathrm{r}}^{-1} G_{\mathrm{r}}^{\top} \left(G_{\mathrm{r}} W_{\mathrm{r}}^{-1} G_{\mathrm{r}}^{\top} \right)^{-1} h.$$

7.2 Fast cost function evaluation

> "... in good design, there must be an underlying correspondence between the structure of the problem and the structure of the solution." R. Gabriel

$$\begin{array}{c} \text{structure of the (SLRA) problem} \\ \mathscr{S} \text{ and } w \\ \updownarrow \\ \text{(in the mosaic-Hankel case)} \\ \text{represented by } \mathbf{m}, \mathbf{n}, w \end{array} \quad \leftrightarrow \quad \begin{array}{c} \text{structure of the cost function} \\ \Gamma \text{ (see (27))} \\ \updownarrow \\ \text{(in the mosaic-Hankel case)} \\ \text{represented by } V_k \text{ (see (28))} \end{array}$$

In [UM13] it is shown that for mosaic-Hankel structure (10) and positive weights (no missing values), the cost function M (and its derivatives) can be evaluated efficiently. The algorithm is implemented in a C++ software package [MU12a].

Let $\gamma_i := w_i^{-1}$, for $i = 1, \ldots, n_p$, where $(+\infty)^{-1} := 0$. By (26) the cost function is equal to

$$M(R) = \operatorname{vec}^\top \big(R\mathscr{S}(p)\big)\Gamma^{-1}(R)\operatorname{vec}\big(R\mathscr{S}(p)\big),$$

where

$$\Gamma(R) := G(R)\operatorname{diag}(\gamma)G^\top(R) \in \mathbb{R}^{(m-r)n\times(m-r)n}. \tag{27}$$

The evaluation of $M(R)$, for a given R, requires solution of the system of linear equations

$$\Gamma(R)u = \operatorname{vec}\big(R\mathscr{S}(p)\big).$$

For structure of the form $\Phi\mathscr{H}_{\mathbf{m},\mathbf{n}}$, the matrix $\Gamma(R)$ is block-diagonal

$$\Gamma(R) = \operatorname{diag}\big(\Gamma^{(1)}(R\Phi), \ldots, \Gamma^{(N)}(R\Phi)\big),$$

with $\Gamma^{(k)}(R)$ being the matrix Γ for the structure $\mathscr{H}_{\mathbf{m},n_k}$ and the corresponding subvector of w.

The efficiency of the algorithm is based on the fact that Γ is block banded with bandwidth $s = \max(m_1, \ldots, m_N)$. In addition, for uniform weights, Γ is block-diagonal with block-Toeplitz elements

$$\Gamma(R) := \begin{bmatrix} \Gamma_{1,1} & \cdots & \Gamma_{1,s} & & 0 \\ \vdots & \ddots & \ddots & \ddots & \\ \Gamma_{s,1} & \ddots & \ddots & \ddots & \Gamma_{n-s+1,n} \\ & \ddots & \ddots & \ddots & \vdots \\ 0 & & \Gamma_{n,n-s+1} & \cdots & \Gamma_{n,n} \end{bmatrix}, \quad \text{where} \quad \Gamma_{i,j} = RV_{i,j}R^\top \in \mathbb{R}^{(m-r)\times(m-r)}.$$

The matrices $V_{i,j} \in \mathbb{R}^{m\times m}$ depend only on the structure specification and the weights, so that they can be precomputed outside the optimization loop. For mosaic-Hankel structured problems $\mathscr{H}_{\mathbf{m},n}$, where $\mathbf{m} = \begin{bmatrix} m_1 & \ldots & m_q \end{bmatrix}$, the matrices $V_{i,j}$, $i \geq j$, are

$$V_{i,j} = \begin{cases} (U_{\mathbf{m}})^{j-i} & \text{if } w = \mathbf{1} \text{ (uniform weights) and} \\ \operatorname{diag}(\gamma_{i,j})\,(U_{\mathbf{m}})^{j-i} & \text{otherwise ,} \end{cases} \tag{28}$$

where $\gamma_{i,j} \in \mathbb{R}^m$ is a subvector of γ, and

$$U_{\mathbf{m}} := \operatorname{diag}(U_{m_1}, \ldots, U_{m_q}), \qquad \text{with} \quad U_m := \begin{bmatrix} 0 & 1 & 0 & 0 \\ \vdots & \ddots & \ddots & 0 \\ \vdots & & \ddots & 1 \\ 0 & \ldots & \ldots & 0 \end{bmatrix}.$$

The cost function evaluation is summarized in Algorithm 1.

For general weights, the algorithm has complexity $O(smn)$ and for uniform weights the complexity is $O(mn)$, assuming that an efficient method (*e.g.*, the Schur-type algorithms [KS95]) is used for the Cholesky factorization of the block-Toeplitz matrices $\Gamma^{(i)}(R\Phi)$.

Algorithm 1 Fast cost function evaluation for mosaic-Hankel matrix with positive weights.

Input: R, p, $\mathscr{S}$, $V_{i,j}$

1: Compute $\nu(R) = \operatorname{vec}\big(R\mathscr{S}(p)\big)$.
2: Compute $\Gamma(R)$, using $V_{i,j}$.
3: Compute the Cholesky factorization $\Gamma(R) = C^\top C$ (Γ is an $2s(m-r)$-banded matrix).
4: Compute $F(R) = C^{-\top}\nu(R)$, by solving $s(m-r)$-banded system of linear equations.
5: Set $M(R) := \|F(R)\|_2^2$.

Output: $M(R)$

8 Related paradigms, problems, and algorithms

8.1 Structured total least squares

The total least squares method is introduced by Golub and Van Loan [Gol73, GV80] as a solution technique for an overdetermined system of linear equations $AX \approx B$, where $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times(m-r)}$ are the given data and $X \in \mathbb{R}^{n\times(m-r)}$ is unknown. The total least squares approximate solution $\widehat{X}$ of the system $AX \approx B$ is a minimizer of the following optimization problem

$$\text{minimize} \quad \text{over } X, \widehat{A}, \text{ and } \widehat{B} \quad \Big\| \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \Big\|_{\mathrm{F}} \quad \text{subject to} \quad \widehat{A}X = \widehat{B}, \tag{29}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm.

With $X \in \mathbb{R}^{r\times\bullet}$, problem (29) is generically equivalent to approximation of the matrix $D := \begin{bmatrix} A & B \end{bmatrix}$ by a rank r matrix $\widehat{D} := \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix}$ in the Frobenius norm

$$\text{minimize} \quad \text{over } \widehat{D} \in \mathbb{R}^{m\times n} \quad \|D - \widehat{D}\|_{\mathrm{F}}^2 \quad \text{subject to} \quad \operatorname{rank}(\widehat{D}) \leq r. \tag{30}$$

While a solution to the low-rank approximation problem (30) always exists, the total least squares problem (29) may fail to have a solution. The nongeneric case of lack of total least squares solution occurs when the optimal approximating matrix $\widehat{D}$ of (30) can not be represented in the form $\widehat{A}X \approx \widehat{B}$.

The generalization of the total least squares problem to systems of linear equations with structured matrices A and B is called structured total least squares [DM93]. The structured total least squares problem is generically equivalent to the structured low-rank approximation problem. Many algorithms for solving structured total least squares approximation are proposed in the literature [AMH91, RPG96, LMV00, MLV00, MVP05] . The variable projection-like method presented in Section 5 of the paper is related to the structured total least squares algorithm of [MVP05].

8.2 Behavioral paradigm

The system of equations $\widehat{A}X \approx \widehat{B}$ implies a functional relation among the variables $\widehat{A}$ to $\widehat{B}$ ($\widehat{B}$ is a function of $\widehat{A}$). In system theory and signal processing, such a relation defines what is called an input-output representation of a model. The rank constraint in (30) does not impose an a priori fixed functional relation or input-output partition on the variables of the matrix $\widehat{D}$, however, the rank deficiency of $\widehat{D}$ implies that such relations exist. They can be inferred from $\widehat{D}$ *after* the approximation problem is solved. The representation-free approach to data modeling is known in the systems and control literature as the behavioral paradigm [Wil87, Wil07].

(SLRA) is a computational method for data modeling in the behavioral setting.

Related methods for system identification in the behavioral setting are developed by Roorda and Heij [RH95, Roo95]. From a system theoretic point of view, the inner minimization (14) of Hankel structured low-rank approximation problem is a Kalman smoothing problem [MD05]. This link allows alternative methods for fast cost function and derivatives evaluation.

8.3 Errors-in-variables model

In the classical regression model, an input-output partitioning of the variables is imposed and the input variables (regressors) are assumed to be noise free. This setup is inadequate for applications where all variables are obtained through measurements and should be treated on an equal footing. The relevant statistical setup in this case is the errors-in-variables model

$$d_i = \bar{d}_i + \tilde{d}_i, \qquad \text{for} \quad i = 1, \ldots, N,$$

where d_i is the vector of measured variables, $\bar{d}_i$ is its true value, and $\tilde{d}_i$ is the measurement noise. Low-rank approximation problem is a maximum likelihood estimator in the errors-in-variables setup, under the assumption that the measurement noise is zero mean, normally distributed, with covariance matrix known up to a scaling factor [Mar08]. Therefore, low-rank approximation yields a statistically optimal estimator in the errors-in-variables setting. Conversely, the errors-in-variables setting provides the relevant testbed for the methods developed in the paper.

8.4 Principal component analysis

The principal component analysis method [Pea01, Jol02, Jac03] is defined as follows: given a set of vectors

$$\mathscr{D} = \{ d_1, \ldots, d_N \} \subset \mathbb{R}^q,$$

drawn from a zero mean distribution, find a subspace $\mathscr{B} = \ker(R)$ of dimension r, which maximizes the empirical variance of the projected data on the subspace. The problem is equivalent to approximation of the data matrix

$$D = \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}.$$

by a rank r matrix.

> Principal component analysis gives a statistical interpretation of the deterministic low-rank approximation problem.

8.5 Rank minimization

In the low-rank approximation problem, the fitting error $\|p - \widehat{p}\|_w$ is minimized subject to a hard constraint on the rank of the approximating matrix $\mathscr{S}(\widehat{p})$. In the related structured rank minimization problem [Faz02]

$$\text{minimize} \quad \text{over } \widehat{p} \quad \text{rank}\left(\mathscr{S}(\widehat{p})\right) \quad \text{subject to} \quad \|p - \widehat{p}\|_w \leq e,$$

the rank of $\mathscr{S}(\widehat{p})$ is minimized subject to a hard constraint on the fitting error. Both problems are NP-hard and are equivalent in the sense that a method for solving one of them can be used for solving the other.

More generally, one can consider the bi-objective problem

$$\text{minimize} \quad \text{over } \widehat{p} \quad \begin{bmatrix} \|p - \widehat{p}\|_w \\ \text{rank}\left(\mathscr{S}(\widehat{p})\right) \end{bmatrix}.$$

Then, the rank minimization and low-rank approximation problems can be viewed as ways of scalarizing the bi-objective problem. Another scalarization of the bi-objective problem is the regularized problem

$$\text{minimize} \quad \text{over } \widehat{p} \quad \|p - \widehat{p}\|_w + \gamma \, \text{rank}\left(\mathscr{S}(\widehat{p})\right).$$

The solutions of all three problems above trace the same Pareto optimal curve when the corresponding hyper parameters—r, in the low-rank approximation problem, e in the rank minimization problem, and γ in the regularized problem—are varied.

Suboptimal solutions to these NP-hard problems can be computed by relaxing them to convex optimization problems. A popular convex relaxation of the rank constraint is the nuclear norm, defined as the sum of the singular values. An upper bound on the nuclear norm can be expressed as a linear matrix inequality, so that the relaxed problems are semi-definite programming problems. Consequently, they can be solved by readily available algorithms and software.

9 Conclusions

As diverse applications as linear time-invariant system identification, algebraic curve fitting, approximate common divisor computation, and recommender systems can be posed and solved as a single core computational problem: low-rank matrix approximation. This fact allows reuse of ideas, algorithms, and software from one application domain into another. In general, the low-rank approximation problem is nonconvex. We presented a method based on the variable projection principle, which is applicable to general linearly structured matrices and weighted 2-norm approximation criteria. In addition, the method can take into account fixed and missing data values. In the case of mosaic-Hankel-like matrices, the developed algorithm for low-rank approximation has linear computational complexity in the number of data points. This makes it efficient for data modeling by low-complexity linear time-invariant dynamical models.

Acknowledgements

Sections 5, 6.6, and 7 are based on joint work with K. Usevich. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Grant agreement number 258581 "Structured low-rank approximation: Theory, algorithms, and applications".

References

[AMH91] T. Abatzoglou, J. Mendel, and G. Harada. The constrained total least squares technique and its application to harmonic superresolution. *IEEE Trans. Signal Proc.*, 39:1070–1087, 1991.

[AMS08] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, 2008.

[AMSD02] P.-A. Absil, R. Mahony, R. Sepulchre, and P. Van Dooren. A Grassmann–Rayleigh quotient iteration for computing invariant subspaces. *SIAM Review*, 44(1):57–73, 2002.

[ASÅ04] M. Agarwal, P. Stoica, and T. P Åhgren. Common factor estimation and two applications in signal processing. *Signal Processing*, 84:421–429, 2004.

[DM93] B. De Moor. Structured total least squares and L_2 approximation problems. *Linear Algebra Appl.*, 188–189:163–207, 1993.

[Faz02] M. Fazel. *Matrix rank minimization with applications*. PhD thesis, Elec. Eng. Dept., Stanford University, 2002.

[GGS94] W. Gander, G. Golub, and R. Strebel. Fitting of circles and ellipses: Least squares solution. *BIT*, 34:558–578, 1994.

[Gol73] G. Golub. Some modified matrix eigenvalue problems. *SIAM Review*, 15:318–344, 1973.

[GP03] G. Golub and V. Pereyra. Separable nonlinear least squares: the variable projection method and its applications. *Institute of Physics, Inverse Problems*, 19:1–26, 2003.

[GV80] G. Golub and C. Van Loan. An analysis of the total least squares problem. *SIAM J. Numer. Anal.*, 17:883–893, 1980.

[Hei95] G. Heinig. Generalized inverses of Hankel and Toeplitz mosaic matrices. *Linear Algebra Appl.*, 216(0):43–59, February 1995.

[Hot33] H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. of Educational Psychology*, 24(7):498–520, 1933.

[IWVD06] Markovsky I, J. C. Willems, S. Van Huffel, and B. De Moor. *Exact and Approximate Modeling of Linear Systems: A Behavioral Approach*. SIAM, March 2006.

[Jac03] J. Jackson. *A User's Guide to Principal Components*. Wiley, 2003.

[JdK06] D. Jibetean and E. de Klerk. Global optimization of rational functions: a semidefinite programming approach. *Mathematical Programming*, 106:93–109, 2006.

[Jol02] I. Jolliffe. *Principal component analysis*. Springer-Verlag, 2002.

[KL98] N. Karmarkar and Y. Lakshman. On approximate GCDs of univariate polynomials. In S. Watt and H. Stetter, editors, *J. Symbolic Comput.*, volume 26, pages 653–666, 1998. Special issue on Symbolic Numeric Algebra for Polynomials.

[Knu92] D. Knuth. *Literate programming*. Cambridge University Press, 1992.

[KS95] T. Kailath and A. Sayed. Displacement structure: theory and applications. *SIAM Review*, 37(3):297–386, 1995.

[KT82] R. Kumaresan and D. Tufts. Estimating the parameters of exponentially damped sinusoids and pole-zero modeling in noise. *IEEE Trans. Acust., Speech, Signal Proc.*, 30(6):833–840, 1982.

[Kun78] S. Kung. A new identification method and model reduction algorithm via singular value decomposition. In *Proc. 12th Asilomar Conf. Circuits, Systems, Computers*, pages 705–714, Pacific Grove, 1978.

[KYZ06] E. Kaltofen, Z. Yang, and L. Zhi. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. In *Proc. Int. Symp. Symbolic Algebraic Computation*, pages 169–176, 2006.

[Lju99] L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, Upper Saddle River, NJ, 1999.

[LMV00] P. Lemmerling, N. Mastronardi, and S. Van Huffel. Fast algorithm for solving the Hankel/Toeplitz structured total least squares problem. *Numerical Algorithms*, 23:371–392, 2000.

[Mar08] I. Markovsky. Structured low-rank approximation and its applications. *Automatica*, 44(4):891–909, 2008.

[Mar12] I. Markovsky. *Low Rank Approximation: Algorithms, Implementation, Applications*. Communications and Control Engineering. Springer, 2012.

[Mat] MathWorks. *Control System Toolbox: User's guide*. The MathWorks.

[MD05] I. Markovsky and B. De Moor. Linear dynamic filtering with noisy input and output. *Automatica*, 41(1):167–171, 2005.

[MLV00] N. Mastronardi, P. Lemmerling, and S. Van Huffel. Fast structured total least squares algorithm for solving the basic deconvolution problem. *SIAM J. Matrix Anal. Appl.*, 22:533–553, 2000.

[MU12a] I. Markovsky and K. Usevich. Software for weighted structured low-rank approximation. Technical Report 339974, Univ. of Southampton, `http://eprints.soton.ac.uk/339974`, 2012.

[MU12b] I. Markovsky and K. Usevich. Structured low-rank approximation with missing values. *SIAM J. Matrix Anal. Appl.*, 2012.

[MV07] I. Markovsky and S. Van Huffel. Overview of total least squares methods. *Signal Proc.*, 87:2283–2302, 2007.

[MVP05] I. Markovsky, S. Van Huffel, and R. Pintelon. Block-Toeplitz/Hankel structured total least squares. *SIAM J. Matrix Anal. Appl.*, 26(4):1083–1099, 2005.

[Nie01] Y. Nievergelt. Hyperspheres and hyperplanes fitting seamlessly by algebraic constrained total least-squares. *Linear Algebra Appl.*, 331:43–59, 2001.

[Pai79] C. C. Paige. Computer solution and perturbation analysis of generalized linear least squares problems. *Mathematics of Computation*, 33(145):171–183, 1979.

[Pea01] K. Pearson. On lines and planes of closest fit to points in space. *Philos. Mag.*, 2:559–572, 1901.

[Ram94] N. Ramsey. Literate programming simplified. *IEEE Software*, 11:97–105, 1994.

[RH95] B. Roorda and C. Heij. Global total least squares modeling of multivariate time series. *IEEE Trans. Automat. Control*, 40(1):50–63, 1995.

[Roo95] B. Roorda. Algorithms for global total least squares modelling of finite multivariable time series. *Automatica*, 31(3):391–404, 1995.

[RPG96] J. Rosen, H. Park, and J. Glick. Total least norm formulation and solution of structured problems. *SIAM J. Matrix Anal. Appl.*, 17:110–126, 1996.

[SM05] P. Stoica and R. Moses. *Spectral Analysis of Signals*. Prentice Hall, 2005.

[Söd07] T. Söderström. Errors-in-variables methods in system identification. *Automatica*, 43:939–958, 2007.

[SS89] T. Söderström and P. Stoica. *System Identification*. Prentice Hall, 1989.

[Ste04] H. Stetter. *Numerical Polynomial Algebra*. SIAM, 2004.

[UM12] K. Usevich and I. Markovsky. Structured low-rank approximation as a rational function minimization. In *Proc. of the 16th IFAC Symposium on System Identification*, Brussels, 2012.

[UM13] K. Usevich and I. Markovsky. Variable projection for affinely structured low-rank approximation in weighted 2-norm. *J. Comput. Appl. Math.*, 2013. Available from `http://arxiv.org/abs/1211.3938`.

[VD96] P. Van Overschee and B. De Moor. *Subspace identification for linear systems: Theory, implementation, applications*. Kluwer, Boston, 1996.

[WAH+97] P. Wentzell, D. Andrews, D. Hamilton, K. Faber, and B. Kowalski. Maximum likelihood principal component analysis. *J. Chemometrics*, 11:339–366, 1997.

[Wil87] J. C. Willems. From time series to linear system—Part I. Finite dimensional linear time invariant systems, Part II. Exact modelling, Part III. Approximate modelling. *Automatica*, 22, 23:561–580, 675–694, 87–115, 1986, 1987.

[Wil07] J. C. Willems. The behavioral approach to open and interconnected systems: Modeling by tearing, zooming, and linking. *Control Systems Magazine*, 27:46–99, 2007. Available online `http://homes.esat.kuleuven.be/~jwillems/Articles/JournalArticles/2007.1.pdf`.