Ivan Markovsky

Low-Rank Approximation

Algorithms, Implementation, Applications

January 19, 2018

Springer

Preface

Simple linear models are commonly used in engineering despite of the fact that the real world is often nonlinear. At the same time as being simple, however, the models have to be accurate. Mathematical models are obtained from first principles (natural laws, interconnection, *etc*.) and experimental data. Modeling from first principles aims at exact models. This approach is the default one in natural sciences. Modeling from data, on the other hand, allows us to tune complexity vs accuracy. This approach is common in engineering, where experimental data is available and a simple approximate model is preferred to a complicated exact one. For example, optimal control of a complex (high-order, nonlinear, time-varying) system is currently intractable, however, using simple (low-order, linear, time-invariant) approximate models and robust control methods may achieve sufficiently high performance. The topic of the book is data modeling by reduced complexity mathematical models.

A unifying theme of the book is low-rank approximation: a prototypical data modeling problem. The rank of a matrix constructed from the data corresponds to the complexity of a linear model that fits the data exactly. The data matrix being full rank implies that there is no exact low complexity linear model for that data. In this case, the aim is to find an approximate model. The approximate modeling approach considered in the book is to find small (in some specified sense) modification of the data that renders the modified data exact. The exact model for the modified data is an optimal (in the specified sense) approximate model for the original data. The corresponding computational problem is low-rank approximation of the data matrix. The rank of the approximation allows us to trade-off accuracy vs complexity.

Apart from the choice of the rank, the book covers two other user choices: the approximation criterion and the matrix structure. These choices correspond to prior knowledge about the accuracy of the data and the model class, respectively. An example of a matrix structure, called Hankel structure, corresponds to the linear time-invariant model class. The book presents local optimization, subspace and convex relaxation-based heuristic methods for a Hankel structured low-rank approximation.

Low-rank approximation is a core problem in applications. Generic examples in systems and control are model reduction and system identification. Low-rank approximation is equivalent to the principal component analysis method in machine learning. Indeed, dimensionality reduction, classification, and information retrieval problems can be posed and solved as particular low-rank approximation problems. Sylvester structured low-rank approximation has applications in computer algebra for the decoupling, factorization, and common divisor computation of polynomials.

The book covers two complementary aspects of data modeling: stochastic estimation and deterministic approximation. The former aims to find from noisy data that is generated by a low-complexity system an estimate of that data generating system. The latter aims to find from exact data that is generated by a high complexity system a low-complexity approximation of the data generating system. In applications, both the stochastic estimation and deterministic approximation aspects are present: the data is imprecise due to measurement errors and is possibly generated by a complicated phenomenon that is not exactly representable by a model in the considered model class. The development of data modeling methods in system identification and signal processing, however, has been dominated by the stochastic estimation point of view. The approximation error is represented in the mainstream data modeling literature as a random process. However, the approximation error is deterministic, so that it is not natural to treat it as a random process. Moreover, the approximation error does not satisfy stochastic regularity conditions such as stationarity, ergodicity, and Gaussianity. These aspects complicate the stochastic approach.

An exception to the stochastic paradigm in data modeling is the behavioral approach, initiated by J. C. Willems in the mid 80's. Although the behavioral approach is motivated by the deterministic approximation aspect of data modeling, it does not exclude the stochastic estimation approach. This book uses the behavioral approach as a unifying language in defining modeling problems and presenting their solutions.

The theory and methods developed in the book lead to algorithms, which are implemented in software. The algorithms clarify the ideas and vice verse the software implementation clarifies the algorithms. Indeed, the software is the ultimate unambiguous description of how the theory and methods are applied in practice. The software allows the reader to reproduce and to modify the examples in the book. The exposition reflects the sequence: theory $\mapsto$ algorithms $\mapsto$ implementation. Correspondingly, the text is interwoven with code that generates the numerical examples being discussed. The reader can try out these methods on their own problems and data. Experimenting with the methods and working on the exercises would lead you to a deeper understanding of the theory and hands-on experience with the methods.

Leuven, *Ivan Markovsky*
January 19, 2018

Contents

Chapter 1
Introduction

Most linear resistors let us treat current as a function of voltage or voltage as a function of current, since [the resistance] R is neither zero nor infinite. But in the two limiting cases—the short circuit and the open circuit—that's not true. To fit these cases neatly in a unified framework, we shouldn't think of the relation between current and voltage as defining a function.

Baez (2010)

The classical paradigm for data modeling invariably assumes that an input/output partitioning of the data is a priori given. For linear models, this paradigm leads to computational problems of solving approximately overdetermined systems of linear equations. Examples of most simple data fitting problems, however, suggest that the a priori fixed input/output partitioning of the data may be inadequate: 1) the fitting criteria often depend implicitly on the choice of the input and output variables, which may be arbitrary, and 2) the resulting computational problems are ill-conditioned in certain cases. An alternative paradigm for data modeling, sometimes refered to as the behavioral paradigm, does not assume a priori fixed input/output partitioning of the data. The corresponding computational problems involve approximation of a matrix constructed from the data by another matrix of lower rank. The chapter proceeds with review of applications in systems and control, signal processing, computer algebra, chemometrics, psychometrics, machine learning, and computer vision that lead to low-rank approximation problems. Finally, classes of generic solution methods for solving low-rank approximation problems are outlined.

1.1 Classical and behavioral paradigms for data modeling

Fitting linear models to data can be achieved, both conceptually and algorithmically, by solving a approximately an overdetermined system of linear equations $AX \approx B$, where the matrices A and B are constructed from the given data and the matrix X parametrizes the to-be-found model. In this *classical paradigm*, the main tools are the ordinary linear least squares method and its numerous variations—regularized least squares, total least squares, robust least squares, and others. The least squares method and its variations are motivated by their applications for data fitting, but they invariably consider solving approximately an overdetermined system of equations.

The underlying premise in the classical paradigm is that existence of an exact linear model for the data is equivalent to existence of solution X to a system $AX = B$. Such a model is a linear function: the variables corresponding to the A matrix are

inputs (or causes) and the variables corresponding to the B matrix are outputs (or consequences) in the sense that they are uniquely determined by the inputs and the model. Note that in the classical paradigm the input/output partition of the variables is postulated a priori. Unless the model is required to have the a priori specified input/output partition, imposing such a structure in advance is ad hoc and leads to undesirable theoretical and numerical features of the modeling methods derived.

An alternative to the classical paradigm that does not impose an a priori fixed input/output partition is the *behavioral paradigm.* In the behavioral paradigm, fitting linear models to data is equivalent to the problem of approximating a matrix D, constructed from the data, by a matrix $\widehat{D}$ of lower rank. Indeed, existence of an exact linear model for D is equivalent to D being rank deficient. Moreover, the rank of D is related to the complexity of the model. This fact is the tenet of the book and is revisited in the following chapters in the context of applications from systems and control, signal processing, computer algebra, and machine learning. Also its implication to the development of numerical algorithms for data fitting is explored.

Existence of a low-complexity exact linear model is equivalent to rank deficiency of the matrix $D = \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$, where $d_{1j}, \ldots, d_{\mathtt{q}j}$ are the observed variables in the jth outcome d_j. Assume that there are at least as many observations as variables, *i.e.*, $\mathtt{q} \leq N$. A linear model for D postulates linear relations among the variables

$$r_k^\top d_j = 0, \quad \text{for } j = 1, \ldots, N.$$

Let $\mathtt{p}$ be the number of linearly independent relations. Then D has rank less than or equal to $\mathtt{m} := \mathtt{q} - \mathtt{p}$. Equivalently, the observations d_j belong to at most $\mathtt{m}$-dimensional subspace $\mathscr{B}$ of $\mathbb{R}^{\mathtt{q}}$. The model for D, defined by the linear relations $r_1, \ldots, r_{\mathtt{p}} \in \mathbb{R}^{\mathtt{q}}$, is the set $\mathscr{B} \subset \mathbb{R}^{\mathtt{q}}$. Once a model is obtained from the data, all possible input/output partitions can be enumerated. This is an analysis problem for the identified model. Therefore, the choice of an input/output partition in the behavioral paradigm to data modeling can be incorporated, if desired, in the modeling problem and thus need not be hypothesized as necessarily done in the classical paradigm.

The classical and behavioral paradigms for data modeling are related but not equivalent. Although existence of solution of the system $AX = B$ implies that the matrix $\begin{bmatrix} A & B \end{bmatrix}$ is low rank, it is not true that $\begin{bmatrix} A & B \end{bmatrix}$ having a sufficiently low rank implies that the system $AX = B$ is solvable. This lack of equivalence causes ill-posed (or numerically ill-conditioned) data fitting problems in the classical paradigm, which have no solution (or are numerically difficult to solve). In terms of the data fitting problem, ill-conditioning of the problem $AX \approx B$ means that the a priori fixed input/output partition of the variables is not corroborated by the data. In the behavioral setting without the a priori fixed input/output partition of the variables, ill-conditioning of the data matrix D implies that the data approximately satisfies linear relations, so that nearly rank deficiency is a good feature of the data.

The classical paradigm is less general than the behavioral paradigm. Indeed, approximate solution of an overdetermined system of equations $AX \approx B$ is one possible approach to achieve low-rank approximation. Alternative approaches are approximation of the data matrix by a matrix that has at least p-dimensional null space, or

at most $\mathtt{m}$-dimensional column space. Parametrizing the null space and the column space by sets of basis vectors, the alternative approaches are:

1. *kernel representation* there is a full row rank matrix $R \in \mathbb{R}^{\mathtt{p}\times\mathtt{q}}$, such that

$$RD = 0,$$

2. *image representation* there are matrices $P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}}$ and $L \in \mathbb{R}^{\mathtt{m}\times N}$, such that

$$D = PL.$$

The approaches using kernel and image representations are equivalent to the original low-rank approximation problem. Next, the use of $AX = B$, kernel, and image representations is illustrated on the most simple data fitting problem—line fitting.

1.2 Motivating example for low-rank approximation

Given a set of points $\{d_1,\ldots,d_N\} \subset \mathbb{R}^2$ in the plane, the aim of the line fitting problem is to find a line passing through the origin that "best" matches the given points. The classical approach for line fitting is to define

$$\begin{bmatrix} a_j \\ b_j \end{bmatrix} := d_j$$

(":=" stands for "by definition", see page 269 for a list of notation) and solve approximately the overdetermined system

$$\operatorname{col}(a_1,\ldots,a_N)x = \operatorname{col}(b_1,\ldots,b_N) \qquad \text{(lse)}$$

(the notation "col" stands for "column vector") by the least squares method. Let x_{ls} be the least squares solution to (lse). Then the least squares fitting line is

$$\mathscr{B}_{\text{ls}} := \{\, d = \left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right] \in \mathbb{R}^2 \mid ax_{\text{ls}} = b \,\}.$$

Geometrically, $\mathscr{B}_{\text{ls}}$ minimizes the sum of the squared *vertical distances* from the data points to the fitting line.

The left plot in Figure 1.1 shows a particular example with $N = 10$ data points. (The data $d_1,\ldots,d_{10}$ are the circles, the fit $\mathscr{B}_{\text{ls}}$ is the solid line, and the fitting errors $e := ax_{\text{ls}} - b$ are the dashed lines.) Visually one expects the best fit to be the vertical axis, so minimizing vertical distances does not seem appropriate in this example.

Note that by solving (lse), a (the first components of the d) is treated differently from b (the second components): b is assumed to be a *function* of a. This is an arbitrary choice; the data can be fitted also by solving approximately the system

$$\operatorname{col}(a_1,\ldots,a_N) = \operatorname{col}(b_1,\ldots,b_N)x, \qquad \text{(lse}')$$

in which case a is assumed to be a function of b. Let x'_{ls} be the least squares solution to (lse′). It gives the fitting line

$$\mathscr{B}'_{\text{ls}} := \{\, d = \left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right] \in \mathbb{R}^2 \mid a = bx'_{\text{ls}} \,\},$$

which minimizes the sum of the squared *horizontal distances* (see the right plot in Figure 1.1). The line $\mathscr{B}'_{\text{ls}}$ happens to achieve the desired fit in the example.

> In the classical approach for data fitting, *i.e.*, solving approximately a system of linear equations in the least squares sense, the choice of the model representation affects the fitting criterion.

This feature of the classical approach is undesirable: it is more natural to specify a desired fitting criterion independently of how the model happens to be parametrized. In many data modeling methods, however, a model representation is a priori fixed and it implicitly corresponds to a particular fitting criterion.

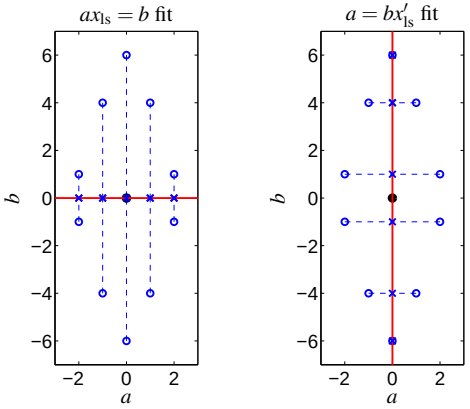


Fig. 1.1: Different partitions of the variables into inputs and outputs may lead to different approximation criteria and also different optimal approximate models. For the example shown in the figure, choosing a as an input, *i.e.*, solving $ax_{\text{ls}} = b$ approximately in the least squares sense, leads to minimization of the sum of squares of the vertical distances, which yields a horizontal line as an optimal model (left plot). On the other hand, choosing b as an input, *i.e.*, solving $a = bx'_{\text{ls}}$ in the least squares sense leads to minimization of the sum of squares of the horizontal distances, which yields a vertical line as an optimal model (right plot).

The total least squares method is an alternative to least squares method for solving approximately an overdetermined system of linear equations. In terms of data fitting, the total least squares method minimizes the sum of the squared *orthogonal*

distances from the data points to the fitting line. Using the system of equations (lse), line fitting by the total least squares method leads to the problem

$$\begin{aligned} &\text{minimize} && \text{over } x \in \mathbb{R},\ \begin{bmatrix} \widehat{a}_1 \\ \vdots \\ \widehat{a}_N \end{bmatrix} \in \mathbb{R}^N, \text{ and } \begin{bmatrix} \widehat{b}_1 \\ \vdots \\ \widehat{b}_N \end{bmatrix} \in \mathbb{R}^N && \sum_{j=1}^{N} \left\| d_j - \begin{bmatrix} \widehat{a}_j \\ \widehat{b}_j \end{bmatrix} \right\|_2^2 \\ &\text{subject to} && \widehat{a}_j x = \widehat{b}_j, \quad \text{for } j = 1, \ldots, N. \end{aligned} \tag{tls}$$

However, for the data in Figure 1.1 the total least squares problem has no solution. Informally, $x_{\text{tls}} = \infty$, which corresponds to a fit by a vertical line. Formally,

problem (tls) may have no solution and therefore may fail to give a model.

The use of (lse) in the definition of the total least squares line fitting problem restricts the fitting line to be a graph of a function $ax = b$ for some $x \in \mathbb{R}$. Thus, the vertical line is a priori excluded as a possible solution. In the example, the line minimizing the sum of the squared orthogonal distances happens to be the vertical line. For this reason, x_{tls} does not exist.

Any line $\mathscr{B}$ passing through the origin can be represented as an image and a kernel, *i.e.*, there exist matrices $P \in \mathbb{R}^{2\times 1}$ and $R \in \mathbb{R}^{1\times 2}$, such that

$$\mathscr{B} = \operatorname{image}(P) := \{ d = P\ell \in \mathbb{R}^2 \mid \ell \in \mathbb{R} \}$$

and

$$\mathscr{B} = \ker(R) := \{ d \in \mathbb{R}^2 \mid Rd = 0 \}.$$

Using the image representation of the model, the line fitting problem of minimizing the sum of the squared orthogonal distances is

$$\begin{aligned} &\text{minimize} && \text{over } P \in \mathbb{R}^{2\times 1} \text{ and } \begin{bmatrix} \ell_1 & \cdots & \ell_N \end{bmatrix} \in \mathbb{R}^{1\times N} && \sum_{j=1}^{N} \| d_j - \widehat{d}_j \|_2^2 \\ &\text{subject to} && \widehat{d}_j = P\ell_j, \quad \text{for } j = 1, \ldots, N. \end{aligned} \tag{lra$'_P$}$$

With

$$D := \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}, \quad \widehat{D} := \begin{bmatrix} \widehat{d}_1 & \cdots & \widehat{d}_N \end{bmatrix},$$

and $\| \cdot \|_{\text{F}}$ the *Frobenius norm*,

$$\|E\|_{\text{F}} := \left\| \operatorname{vec}(E) \right\|_2 = \left\| \begin{bmatrix} e_{11} & \cdots & e_{\mathtt{q}1} & \cdots & e_{1N} & \cdots & e_{\mathtt{q}N} \end{bmatrix}^\top \right\|_2, \quad \text{for all } E \in \mathbb{R}^{\mathtt{q}\times N}$$

(lra$'_P$) is more compactly written as

$$\begin{aligned} &\text{minimize} && \text{over } P \in \mathbb{R}^{2\times 1} \text{ and } L \in \mathbb{R}^{1\times N} && \|D - \widehat{D}\|_{\text{F}}^2 \\ &\text{subject to} && \widehat{D} = PL. \end{aligned} \tag{lra$_P$}$$

Similarly, using a kernel representation, the line fitting problem, minimizing the sum of squares of the orthogonal distances is

$$\begin{aligned} &\text{minimize} && \text{over } R \in \mathbb{R}^{1\times 2},\ R \neq 0, \text{ and } \widehat{D} \in \mathbb{R}^{2\times N} && \|D - \widehat{D}\|_{\text{F}}^2 \\ &\text{subject to} && R\widehat{D} = 0. \end{aligned} \tag{lra$_R$}$$

Contrary to the total least squares problem (tls), problems (lra$_P$) and (lra$_R$) always have (nonunique) solutions. In the example, solutions are, *e.g.*, $P^* = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$ and $R^* = \begin{bmatrix} 1 & 0 \end{bmatrix}$, which describe the vertical line

$$\mathscr{B}^* := \operatorname{image}(P^*) = \ker(R^*).$$

The constraints

$$\widehat{D} = PL, \text{ with } P \in \mathbb{R}^{2\times 1},\ L \in \mathbb{R}^{1\times N} \quad \text{and} \quad R\widehat{D} = 0, \text{ with } R \in \mathbb{R}^{1\times 2},\ R \neq 0$$

are equivalent to the constraint $\operatorname{rank}(\widehat{D}) \leq 1$, which shows that the points $\widehat{d}_1, \ldots, \widehat{d}_N$ being fitted exactly by a line passing through the origin is equivalent to

$$\operatorname{rank}\left(\begin{bmatrix} \widehat{d}_1 & \cdots & \widehat{d}_N \end{bmatrix} \right) \leq 1.$$

Thus, (lra$_P$) and (lra$_R$) are instances of one and the same

abstract problem: approximate the data matrix D by a low-rank matrix $\widehat{D}$.

The observations made in the line fitting example are generalized to modeling of q-dimensional data in in the following chapters. The underlying goal is:

Given a set of points in $\mathbb{R}^{\mathtt{q}}$ (the data), find a subspace of $\mathbb{R}^{\mathtt{q}}$ of bounded dimension (a model) that has the least (2-norm) distance to the data points.

Such a subspace is a (2-norm) optimal fitting *model*. The model can always be *represented* as a kernel or an image of a matrix. The classical least squares and total least squares formulations of the data modeling problem use instead the input/output representations $AX = B$ and $A = BX'$ of the model that exclude some candidate models. This leads to the theoretical and numerical issues of lack of solution and ill-conditioning. Data modeling using the kernel and image representations is free of these issues. Therefore, they are the preferred choices in data modeling.

The equations $AX = B$ and $A = BX'$ were introduced from an algorithmic point of view—by using them, the data fitting problem is turned into the standard problem of solving approximately an overdetermined system of linear equations. An interpretation of these equations in the data modeling context is that in the model represented

by the equation $AX = B$, the variable A is an input and the variable B is an output. Similarly, in the model represented by the equation $A = BX'$, A is an output and B is an input. The input/output interpretation has an intuitive appeal because it implies a causal dependence of the variables: the input is causing the output.

Representing the model by an equation $AX = B$ and $A = BX'$, as done in the classical approach, one a priori assumes that the optimal fitting model has a certain input/output structure. The consequences are:

- existence of exceptional (nongeneric) cases, which complicate the theory,
- ill-conditioning caused by "nearly" exceptional cases, which leads to lack of numerical robustness of the algorithms, and
- need of regularization, which leads to a change of the specified fitting criterion.

These aspects of the classical approach are generally considered as inherent to the data modeling problem. By choosing the alternative image and kernel model representations, the problem of solving approximately an overdetermined system of equations becomes a low-rank approximation problem, where the nongeneric cases (and the related issues of ill-conditioning and need of regularization) are avoided.

1.3 Overview of applications

The motto of the book is:

> At the core of every data modeling problem is a low-rank approximation subproblem. Finding this subproblem allows us to use existing theory, methods, and algorithms for the solution of the original data modeling problem.

This section illustrates the link between data modeling problems and low-rank approximation on examples from systems and control, signal processing, computer algebra, chemometrics, psychometrics, machine learning, and computer vision. The fact that a matrix constructed from exact data is low rank and the approximate modeling problem is low-rank approximation is sometimes well known (*e.g.*, in realization theory, model reduction, and approximate polynomial common factor). In other cases (*e.g.*, natural language processing and conic section fitting), the link to low-rank approximation is not well known and is not exploited.

1.3.1 Exact, approximate, deterministic, and stochastic modeling

As we have seen in the example of line fitting in Section 1.2, the model imposes relations on the data, which render a matrix constructed from exact data rank deficient. Although an exact data matrix is low rank, a matrix constructed from real measurements is generically full rank due to measurement noise, unaccounted effects, and assumptions about the data generating system that are not satisfied in practice. Therefore, generically, the observed data does not have an exact low-complexity model. This leads to the problem of approximate modeling.

The approximate modeling problem can be formulated as a low-rank approximation problem as follows: we aim to modify the data as little as possible, so that the matrix constructed from the modified data has a specified low rank. The modified data matrix being low rank implies that there is an exact model for the modified data. This model is by definition an approximate model for the given data. The transition from exact to approximate modeling is an important step in building a coherent theory for data modeling and is emphasized in this book.

In all applications, the exact modeling problem is discussed before the practically more relevant approximate modeling problem. This is done because 1) exact modeling is simpler than approximate modeling, so that it is the right starting place, and 2) exact modeling is a part of optimal approximate modeling and suggests ways of solving such problems suboptimally. Indeed, small modifications of exact modeling algorithms lead to effective approximate modeling algorithms. Well known examples of the transition from exact to approximate modeling in systems theory are the progressions from realization theory to model reduction and from deterministic subspace identification to approximate and stochastic subspace identification.

A consistent estimator in the stochastic estimation setting of the data modeling problem achieves an exact model. Indeed, the true data generating system is asymptotically recovered from the observed data. Estimation with finite sample size, however, necessarily involves approximation. Thus in stochastic estimation theory there is also a step of transition from exact to approximate, see Figure 1.2.

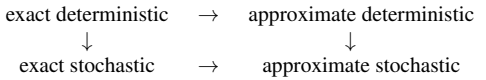


Fig. 1.2: In data modeling, as in any other area, the progression is from simpler to more complex problems, *i.e.*, from exact to approximate and from deterministic to stochastic.

The following subsections can be read in any order.

1.3.2 Applications in systems and control

A major application area of structured low-rank approximation is systems and control. Hankel structured rank-deficient matrices feature prominently in Kalman's realization theory. Also the stochastic version of the realization problem, where the given data is the output correlation function instead of the impulse response leads to a rank constraint on a Hankel matrix constructed from the given data. In both cases—deterministic and stochastic realization—the rank constraint has a physical meaning—it is the order of the system. Hankel structured rank deficient matrices appear also in system identification, where the data is a general trajectory of the unknown system. In this more general case, the rank constraint is a function of the number of inputs as well as the order of the system.

Deterministic system realization and model reduction

Realization theory addresses the problem of finding a state representation of a linear time-invariant dynamical system defined by a transfer function or impulse response representation. The key result in realization theory is that an infinite sequence

$$H = \big(H(0), H(1), \ldots, H(t), \ldots\big)$$

is an impulse response of a discrete-time linear time-invariant system of order $\mathtt{n}$ if and only if the two sided infinite Hankel matrix

$$\mathscr{H}(H) := \begin{bmatrix} H(1) & H(2) & H(3) & \cdots \\ H(2) & H(3) & \iddots & \\ H(3) & \iddots & & \\ \vdots & & & \end{bmatrix},$$

constructed from the data H has rank equal to $\mathtt{n}$, *i.e.*,

$$\operatorname{rank}\big(\mathscr{H}(H)\big) = \text{order of a minimal realization of } H.$$

Therefore, existence of a finite dimensional realization of H, which is an exact low complexity linear time-invariant model for H is equivalent to rank deficiency of a Hankel matrix constructed from the data. Moreover, a minimal state representation of the model can be obtained from a rank revealing factorization of $\mathscr{H}(H)$.

When there is no exact finite dimensional realization of the data or the exact realization is of high order, one may want to find an approximate realization of a specified low order $\mathtt{n}$. These, respectively, approximate realization and model reduction problems naturally lead to *Hankel structured low-rank approximation.*

The deterministic system realization and model reduction problems are further considered in Sections 2.2.2, 3.1, and 5.1.

Stochastic system realization

Let y be the output of an nth order linear time-invariant system, driven by white noise (a stochastic system) and let $\mathbf{E}$ be the expectation operator. The sequence

$$R_{yy} = \big(R_{yy}(0), R_{yy}(1), \ldots, R_{yy}(t), \ldots\big)$$

defined by

$$R_{yy}(\tau) := \mathbf{E}\left(y(t)y^{\top}(t-\tau)\right)$$

is called the *correlation* function of y. Stochastic realization theory is concerned with the problem of finding a state representation of a stochastic system that could have generated the observed output y, *i.e.*, a linear time-invariant system driven by white noise, whose output correlation function is equal to R_{yy}.

An important result in stochastic realization theory is that R_{yy} is the output correlation function of an $\mathtt{n}$th order linear time-invariant stochastic system if and only if the Hankel matrix $\mathscr{H}(R_{yy})$ constructed from R_{yy} has rank $\mathtt{n}$, *i.e.*,

$$\operatorname{rank}\big(\mathscr{H}(R_{yy})\big) = \text{order of a minimal stochastic realization of } R_{yy}.$$

Therefore, stochastic realization of a random process y is equivalent to deterministic realization of its correlation function R_{yy}. A finite dimensional stochastic realization can be obtained from a rank revealing factorization of the Hankel matrix $\mathscr{H}(R_{yy})$.

In practice, the correlation function is *estimated* from a finite number of output samples. With an estimate $\widehat{R}_{yy}$ of R_{yy}, $\mathscr{H}(\widehat{R}_{yy})$ is generically full rank, so that that a finite dimensional stochastic realization can not be found. In this case, the problem of finding an approximate stochastic realization occurs. This problem is equivalent to the one in the deterministic case—Hankel structured low-rank approximation.

System identification

Realization theory considers a *system representation problem*: pass from one representation of a system to another. Alternatively, it can be viewed as a special exact identification problem: find from impulse response data (a special trajectory of the system) a state space representation of the data generating system. The exact identification problem (also called deterministic identification problem) is to find from a general trajectory of a system, a representation of that system. Let

$$w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right], \quad \text{where } u = \big(u(1), \ldots, u(T)\big) \quad \text{and} \quad y = \big(y(1), \ldots, y(T)\big)$$

be an input/output trajectory of a discrete-time linear time-invariant system of order $\mathtt{n}$ with $\mathtt{m}$ inputs and $\mathtt{p}$ outputs and let $\mathtt{n}_{\max}$ be a given upper bound on the order $\mathtt{n}$. Then the Hankel matrix

$$\mathscr{H}_{\mathtt{n}_{\max}+1}(w) := \begin{bmatrix} w(1) & w(2) & \cdots & w(T-\mathtt{n}_{\max}) \\ w(2) & w(3) & \cdots & w(T-\mathtt{n}_{\max}+1) \\ \vdots & \vdots & & \vdots \\ w(\mathtt{n}_{\max}+1) & w(\mathtt{n}_{\max}+1) & \cdots & w(T) \end{bmatrix} \qquad (\mathscr{H}_i)$$

with $\mathtt{n}_{\max}+1$ block rows, constructed from the trajectory w, is rank deficient:

$$\operatorname{rank}\big(\mathscr{H}_{\mathtt{n}_{\max}+1}(w)\big) \leq \operatorname{rank}\big(\mathscr{H}_{\mathtt{n}_{\max}+1}(u)\big) + \text{order of the system.} \qquad \text{(SYSID)}$$

Conversely, if the Hankel matrix $\mathscr{H}_{\mathtt{n}_{\max}+1}(w)$ has rank $(\mathtt{n}_{\max}+1)\mathtt{m}+\mathtt{n}$ and the matrix $\mathscr{H}_{2\mathtt{n}_{\max}+1}(u)$ is full row rank (*persistency of excitation* of u), then w is a trajectory of a controllable linear time-invariant system of order $\mathtt{n}$. Under the above assumptions, the data generating system can be identified from a rank revealing factorization of the matrix $\mathscr{H}_{\mathtt{n}_{\max}+1}(w)$.

When there are measurement errors or the data generating system is not a low complexity linear time-invariant system, the data matrix $\mathscr{H}_{\mathtt{n}_{\max}+1}(w)$ is generically full rank. In such cases, the problem of finding an approximate low-complexity linear time-invariant model for w is Hankel structured low-rank approximation.

System identification is a main topic of the book and appears frequently in the following chapters. Similarly, to the analogy between deterministic and stochastic system realization, there is an analogy between deterministic and stochastic system identification. In a stochastic setting, we consider both errors-in-variables and Auto-Regressive Moving-Average eXogenous (ARMAX) identification problems.

1.3.3 Application for polynomial common factor computation

The greatest common divisor of two polynomials

$$p(z) = p_0 + p_1 z + \cdots + p_n z^n \quad \text{and} \quad q(z) = q_0 + q_1 z + \cdots + q_m z^m$$

is a polynomial c of maximal degree that divides p and q, *i.e.*, there are polynomials r and s, such that $p = rc$ and $q = sc$. Define the Sylvester matrix of p and q

$$\mathscr{R}(p,q) := \begin{bmatrix} p_0 & & & q_0 & & & \\ p_1 & p_0 & & q_1 & q_0 & & \\ \vdots & p_1 & \ddots & \vdots & q_1 & \ddots & \\ p_n & \vdots & \ddots & p_0 & q_m & \vdots & \ddots & q_0 \\ & p_n & & p_1 & & q_m & & q_1 \\ & & \ddots & \vdots & & & \ddots & \vdots \\ & & & p_n & & & & q_m \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}. \qquad (\mathscr{R})$$

(By convention, in this book, all missing entries in a matrix are assumed to be zeros.) A well known fact in algebra is that the degree of the greatest common divisor of p and q is equal to the rank deficiency (co-rank) of $\mathscr{R}(p,q)$, *i.e.*,

$$\operatorname{degree}(c) = n + m - \operatorname{rank}\big(\mathscr{R}(p,q)\big). \qquad \text{(GCD)}$$

Suppose that p and q have a greatest common divisor of known degree $\mathtt{d} > 0$, but the coefficients of the polynomials p and q are imprecise. The perturbed polynomials $p_{\mathtt{d}}$ and $q_{\mathtt{d}}$ are generically co-prime, *i.e.*, they have no common factor, or equivalently, the matrix $\mathscr{R}(p_{\mathtt{d}}, q_{\mathtt{d}})$ is full rank.

The problem of finding an *approximate common factor* of $p_{\mathtt{d}}$ and $q_{\mathtt{d}}$ with degree $\mathtt{d}$, can be formulated as follows. Modify the coefficients of $p_{\mathtt{d}}$ and $q_{\mathtt{d}}$, as little as possible, so that the resulting polynomials, say, $\widehat{p}$ and $\widehat{q}$ have a greatest common divisor of degree $\mathtt{d}$. This problem is a Sylvester structured low-rank approximation problem. The approximate common factor $\widehat{c}$ for the perturbed polynomials $p_{\mathtt{d}}$ and $q_{\mathtt{d}}$ is the exact greatest common divisor of $\widehat{p}$ and $\widehat{q}$.

The problem of computing approximate common factor of two polynomials is considered in Section 5.3.

1.3.4 Applications for antenna array signal processing

An array of antennas or sensors is used for direction of arrival estimation and adaptive beamforming. Consider $\mathtt{q}$ antennas in a fixed configuration and a wave propagating from distant sources, see Figure 1.3.

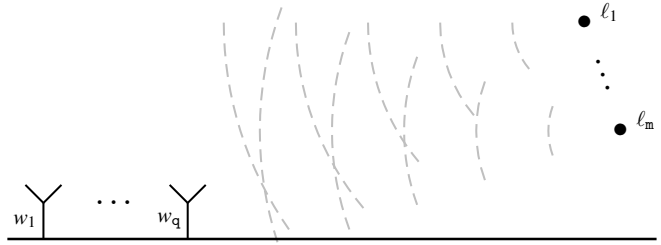


Fig. 1.3: In array signal processing, $\mathtt{q}$ antenna receive the superposition of $\mathtt{m}$ waves emitted from distant sources.

Consider, first, the case of a single source. The source intensity ℓ_1 (the signal) is a function of time. Let $w(t) \in \mathbb{R}^{\mathtt{q}}$ be the response of the array at time t (w_i being the response of the ith antenna). Assuming that the source is far from the array (relative to the array's length), the array's response is proportional to the source intensity

$$w(t) = p_1 \ell_1(t - \tau_1),$$

where τ_1 is the time needed for the wave to travel from the source to the array and $p_1 \in \mathbb{R}^{\mathtt{q}}$ is the array's response to the source emitting at a unit intensity. The vector p_1 depends only on the array geometry and the source location, so that it is constant in time. Measurements of the antenna at time $t = 1, \ldots, T$ give a data matrix

$$D := \begin{bmatrix} w(1) & \cdots & w(T) \end{bmatrix} = p_1 \underbrace{\begin{bmatrix} \ell_1(1-\tau) & \cdots & \ell_1(T-\tau) \end{bmatrix}}_{\ell_1} = p_1 \ell_1,$$

which has rank equals to one.

Consider now $\mathtt{m} < \mathtt{q}$ distant sources emitting with intensities $\ell_1, \ldots, \ell_{\mathtt{m}}$. Let p_k be the response of the array to the kth source emitting alone with unit intensity. Assuming that the array responds linearly to a mixture of sources,

$$D = \begin{bmatrix} w(1) & \cdots & w(T) \end{bmatrix} = \sum_{k=1}^{\mathtt{m}} p_k \underbrace{\begin{bmatrix} \ell_k(1-\tau_k) & \cdots & \ell_k(T-\tau_k) \end{bmatrix}}_{\ell_k} = PL,$$

where $P := \begin{bmatrix} p_1 & \cdots & p_{\mathtt{m}} \end{bmatrix}$, $L := \operatorname{col}(\ell_1, \ldots, \ell_{\mathtt{m}})$, and τ_k is the delay of the wave coming from the kth source. This shows that the rank of D is less than or equal to the number of sources $\mathtt{m}$. If there are enough antennas and observed samples ($q > \mathtt{m}$ and $T > \mathtt{m}$) and, in addition, the source signals are linearly independent,

$$\operatorname{rank}(D) = \text{the number of sources transmitting to the array.}$$

Moreover, the factors P and L in a rank revealing factorization PL of D carry information about the source locations.

With noisy observations, the matrix D is generically a full rank matrix. Then, assuming that the array's geometry is known, low-rank approximation can be used to estimate the number of sources and their locations.

1.3.5 Applications in chemometrics for multivariate calibration

A basic problem in chemometrics, called multivariate calibration, is identification of the number and type of chemical components from spectral measurements of mixtures of these components. Let $p_k \in \mathbb{R}^{\mathtt{q}}$ be the spectrum of the kth component at $\mathtt{q}$ predefined frequencies. Under a linearity assumption, the spectrum of a mixture of $\mathtt{m}$ components with concentrations $\ell_1, \ldots, \ell_{\mathtt{m}}$ is $d = P\ell$, where $P := \begin{bmatrix} p_1 & \cdots & p_{\mathtt{m}} \end{bmatrix}$ and $\ell = \operatorname{col}(\ell_1, \ldots, \ell_{\mathtt{m}})$. Given N mixtures of the components with vectors of concentrations $\ell^{(1)}, \ldots, \ell^{(N)}$, the matrix of the corresponding spectra $d_1, \ldots, d_N$ is

$$D := \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix} = \sum_{k=1}^{\mathtt{m}} p_k \underbrace{\begin{bmatrix} \ell_k^{(1)} & \cdots & \ell_k^{(N)} \end{bmatrix}}_{\ell_k} = PL. \qquad \text{(RRF)}$$

Therefore, the rank of D is at most equal to the number of components $\mathtt{m}$. Assuming that $\mathtt{q} > \mathtt{m}$, $N > \mathtt{m}$, the spectral responses $p_1, \ldots, p_{\mathtt{m}}$ of the components are linearly independent, and the concentration vectors $\ell_1, \ldots, \ell_{\mathtt{m}}$ are linearly independent,

$$\operatorname{rank}(D) = \text{the number of chemical components in the mixtures.}$$

The factors P and L in a rank revealing factorization PL of D carry information about the components' spectra and the concentrations of the components in the mixtures. With noisy observations D is full rank, so that low-rank approximation can be used to estimate the number, concentrations, and spectra of the chemical components.

1.3.6 Applications in psychometrics for factor analysis

The psychometric data is test scores and biometrics of a group of people. The test scores can be organized in a data matrix D, whose rows correspond to the scores and the columns correspond to the group members. Factor analysis is a popular method that explains the data as a linear combination of a small number of abilities of the group members. These abilities are called *factors* and the weights by which they are combined in order to reproduce the data are called *loadings*. Factor analysis is based on the assumption that the exact data matrix is low rank with rank being equal to the number of factors. Indeed, the factor model can be written as $D = PL$, where the columns of P correspond to the factors and the rows of L correspond to the loadings. In practice, the data matrix is full rank because the factor model is an idealization of the way in which the test data is generated. Despite of the fact that the factor model is a simplification of the reality, it can be used as an approximation of the way humans perform on tests. Low-rank approximation then is a method for deriving optimal in a specified sense approximate psychometric factor models.

The factor model, explained above, is used to assess candidates at universities. An important element of the acceptance decision in US universities for undergraduate study is the Scholastic Aptitude Test, and for postgraduate study, the Graduate Record Examination. These tests report three independent scores: writing, mathematics, and critical reading for the Scholastic Aptitude Test; and verbal, quantitative, and analytical for the Graduate Record Examination. The three scores assess what are believed to be the three major factors for, respectively, undergraduate and postgraduate academic performance. In other words, the premise on which the tests are based is that the ability of a prospective student to do undergraduate and postgraduate study is predicted well by a combination of the three factors. Of course, in different areas of study, the weights by which the factors are taken into consideration are different. Even in "pure subjects", however, all scores are important.

> Many graduate-school advisors have noted that an applicant for a mathematics fellowship with a high score on the verbal part of the Graduate Record Examination is a better bet as a Ph.D. candidate than one who did well on the quantitative part but badly on the verbal.
>
> Halmos (1985, page 5)

1.3.7 Applications in machine learning

Low-rank approximation is a core problem in machine learning. Applications considered next are natural language processing, recommender systems, multidimensional scaling, and microarray data analysis. Each of these applications imposes specific constraints on the data besides the low-rank structure. In natural language processing, the data consists of frequencies of words in a document, so that the data matrix is element-wise nonnegative. In recommender systems, the data is ordinal and the goal is to estimate missing values. In multidimensional scaling, the data matrix is a nonlinear (bilinear) transformation of the original measurements. In microarray data analysis, data matrix D is approximated by a low-rank matrix PL, where P has known zero elements and the rows of L are nonnegative, smooth, and periodic as a function of the index.

Natural language processing

Latent semantic analysis is a method in natural language processing for document classification, search by keywords, synonymy and polysemy detection, *etc*. Latent semantic analysis is based on low-rank approximation and fits into the pattern of the other applications reviewed here:

1. An exact data matrix is rank deficient with rank related to the complexity of the data generating model.
2. A noisy data matrix is full rank and, for the purpose of approximate modeling, it is approximated by a low-rank matrix.

Consider N documents, involving $\mathtt{q}$ terms and $\mathtt{m}$ concepts. If a document belongs to the kth concept only, it contains the ith term with frequency p_{ik}, resulting in the term frequencies vector $p_k := \mathrm{col}(p_{1k},\ldots,p_{\mathtt{q}k})$, related to the kth concept. The latent semantic analysis model assumes that if a document involves a mixture of the concepts with weights $\ell_1,\ldots,\ell_{\mathtt{m}}$ (ℓ_k indicates the relevance of the kth concept to the document), then the vector of term frequencies for that document is

$$d = P\ell, \quad \text{where } P := \begin{bmatrix} p_1 & \cdots & p_{\mathtt{m}} \end{bmatrix} \quad \text{and} \quad \ell = \mathrm{col}(\ell_1,\ldots,\ell_{\mathtt{m}}).$$

Let d_j be the vector of term frequencies, related to the jth document and let $\ell_k^{(j)}$ be the relevance of the kth concept to the jth document. The latent semantic analysis model is the low-rank representation (RRF) of the term–document frequencies matrix D. The rank of the data matrix is less than or equal to the number of concepts $\mathtt{m}$. Assuming that $\mathtt{m}$ is smaller than the number of terms $\mathtt{q}$, $\mathtt{m}$ is smaller than the number of documents N, the term frequencies $p_1,\ldots,p_{\mathtt{m}}$ are linearly independent, and the relevance of concepts $\ell_1,\ldots,\ell_{\mathtt{m}}$ are linearly independent,

$$\mathrm{rank}(D) = \text{the number of concepts related to the documents.}$$

The factors P and L of a rank revealing factorization PL of D carry information about the relevance of the concepts to the documents and the concepts' term frequencies.

The latent semantic analysis model is not satisfied exactly in practice because the notion of (small number of) concepts related to (many) documents is an idealization. Also the linearity assumption is not likely to hold in practice. In reality the term–document frequencies matrix D is full rank indicating that the number of concepts is equal to either the number of terms or the number of documents. Low-rank approximation, however, can be used to find a small number of concepts that explain approximately the term–documents frequencies via the model (RRF). Subsequently, similarity of documents can be evaluated in the concepts space, which is a low dimensional vector space. For example, the j_1th and j_2th documents are related if they have close relevance $\ell_k^{(j_1)}$ and $\ell_k^{(j_2)}$ to all concepts $k = 1,\ldots,\mathtt{m}$. This gives a way to classify the documents. Similarly, terms can be clustered in the concepts space by looking at the rows of the P matrix. Nearby rows of P correspond to terms that are related to the same concepts. (Such terms are likely to be synonymous.) Finally, a search for documents by keywords can be done by first translating the keywords to a vector in the concepts space and then finding a nearby cluster of documents to this vector. For example, if there is a single keyword, which is the ith term, then the ith row of the P matrix shows the relevant combination of concepts for this search.

Recommender system

The main issue underlying the abstract low-rank approximation problem and the applications reviewed up to now is data approximation. In the recommender system problem, the main issue is the one of *missing data*: Given ratings of some items by some users, infer the missing ratings. Unique recovery of the missing data is impossible without additional assumptions. The underlying assumption in many recommender system problems is that the complete matrix of the ratings is of low rank.

Consider $\mathtt{q}$ items and N users and let d_{ij} be the rating of the ith item by the jth user. As in the psychometrics example, it is assumed that there is a "small" number $\mathtt{m}$ of "typical" (or characteristic, or factor) users, such that all user ratings can be obtained as linear combinations of the ratings of the typical users. This implies that the complete matrix $D = \begin{bmatrix} d_{ij} \end{bmatrix}$ of the ratings has rank $\mathtt{m}$, *i.e.*,

$$\mathrm{rank}(D) = \text{number of "typical" users.}$$

Exploiting the prior knowledge that the number of "typical" users is small, the missing data recovery problem can be posed as the *matrix completion problem*

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{D} \quad \mathrm{rank}(\widehat{D}) \\ &\text{subject to} \quad \widehat{D}_{ij} = D_{ij} \quad \text{for all } (i,j), \text{ where } D_{ij} \text{ is given.} \end{aligned} \tag{MC}$$

This gives a procedure for solving the exact modelling problem (the given elements of D are assumed to be exact). The corresponding solution method can be viewed as

the equivalent of the rank revealing factorization problem in exact modeling problems, for the case of complete data.

Of course, the rank minimization problem (MC) is much harder to solve than the rank revealing factorization problem. Moreover, theoretical justification and additional assumptions (about the number and distribution of the given elements of D) are needed for a solution $\widehat{D}$ of (MC) to be unique and to coincide with the complete true matrix D. It turns out, however, that under certain specified assumptions exact recovery is possible by solving the convex optimization problem obtained by replacing $\operatorname{rank}(\widehat{D})$ in (MC) with the nuclear norm

$$\|\widehat{D}\|_* := \text{sum of the singular values of } \widehat{D}.$$

The importance of the result is that under the specified assumptions the hard problem (MC) can be solved efficiently and reliably by convex optimization methods.

In real-life application of recommender systems, however, the additional problem of data approximation occurs. In this case the constraint $\widehat{D}_{ij} = D_{ij}$ of (MC) has to be relaxed, *e.g.*, replacing it by

$$\widehat{D}_{ij} = D_{ij} + \Delta D_{ij},$$

where ΔD_{ij} are corrections, accounting for the data uncertainty. The corrections are additional optimization variables. Taking into account the prior knowledge that the corrections are small, a term $\lambda\|\Delta D\|_{\mathrm{F}}$ is added in the cost function. The resulting matrix approximation problem is

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{D} \text{ and } \Delta D \quad \operatorname{rank}(\widehat{D}) + \lambda\|\Delta D\|_{\mathrm{F}} \\ \text{subject to} & \widehat{D}_{ij} = D_{ij} + \Delta D_{ij} \quad \text{for all } (i,j), \text{ where } D_{ij} \text{ is given.} \end{array} \tag{AMC}$$

In a stochastic setting the term $\lambda\|\Delta D\|_{\mathrm{F}}$ corresponds to the assumption that the true data is perturbed with zero mean, independent, equal variance, Gaussian noise.

Again the problem can be relaxed to a convex optimization problem by replacing rank with nuclear norm. The choice of the λ parameter reflects the trade-off between complexity (number of identified "typical" users) and accuracy (size of the correction ΔD) and depends in the stochastic setting on the noise variance.

Nuclear norm and low-rank approximation methods for estimation of missing values are developed in Sections 3.4 and 4.4.

Multidimensional scaling

Consider a set of N points in the plane

$$\mathscr{X} := \{x_1, \ldots, x_N\} \subset \mathbb{R}^2$$

and let d_{ij} be the squared distance from x_i to x_j, *i.e.*,

$$d_{ij} := \|x_i - x_j\|_2^2.$$

The $N \times N$ matrix $D = \begin{bmatrix} d_{ij} \end{bmatrix}$ of the pair-wise distances, called in what follows the distance matrix (for the set of points $\mathscr{X}$), has rank at most 4. Indeed,

$$d_{ij} = (x_i - x_j)^\top (x_i - x_j) = x_i^\top x_i - 2x_i^\top x_j + x_j^\top x_j,$$

so that

$$D = \underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} x_1^\top x_1 & \cdots & x_N^\top x_N \end{bmatrix}}_{\text{rank} \le 1} - 2 \underbrace{\begin{bmatrix} x_1^\top \\ \vdots \\ x_N^\top \end{bmatrix} \begin{bmatrix} x_1 & \cdots & x_N \end{bmatrix}}_{\text{rank} \le 2} + \underbrace{\begin{bmatrix} x_1^\top x_1 \\ \vdots \\ x_N^\top x_N \end{bmatrix} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}}_{\text{rank} \le 1}. \tag{$*$}$$

The localization problem from pair-wise distances is: Given the distance matrix D, find the locations $\{x_1, \ldots, x_N\}$ of the points up to a rigid transformation, *i.e.*, up to translation, rotation, and reflection of the points. Note that rigid transformations preserve the pair-wise distances, so that the distance matrix D alone is not sufficient to locate the points uniquely.

With exact data, the problem can be posed and solved as a rank revealing factorization problem ($*$). With noisy measurements, the matrix D is generically full rank. In this case, the relative (up to rigid transformation) point locations can be *estimated* by approximating D by a rank-4 matrix $\widehat{D}$. In order to be a valid distance matrix, however, $\widehat{D}$ must have the structure

$$\widehat{D} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \begin{bmatrix} \widehat{x}_1^\top \widehat{x}_1 & \cdots & \widehat{x}_N^\top \widehat{x}_N \end{bmatrix} - 2\widehat{X}^\top \widehat{X} + \begin{bmatrix} \widehat{x}_1^\top \widehat{x}_1 \\ \vdots \\ \widehat{x}_N^\top \widehat{x}_N \end{bmatrix} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}. \tag{MDS}$$

Therefore, the problem is a *bilinearly structured low-rank approximation.*

Microarray data analysis

The measurements of a microarray are collected in a $\mathrm{q} \times N$ matrix D—rows correspond to genes and columns correspond to time instances. The element d_{ij} is the *expression level* of the ith gene at the jth moment of time. The rank of D is equal to the number of *transcription factors* that regulate the gene expression levels

$$\operatorname{rank}(D) = \text{number of transcription factors.}$$

In a rank revealing factorization $D = PL$, the jth column of L is a vector of intensities of the transcription factors at time j, and the ith row of P is a vector of sensitivities of the ith gene to the transcription factors. For example, p_{ij} equal to zero means that the jth transcription factor does not regulate the ith gene.

An important problem in bioinformatics is to discover what transcription factors regulate a particular gene and what their time evaluations are. This problem amounts to computing an (approximate) factorization PL of the matrix D. The need of approximation comes from:

1. inability to account for all relevant transcription factors (therefore accounting only for a few dominant ones), and
2. measurement errors occurring in the collection of the data.

Often it is known a priori that certain transcription factors do not regulate certain genes. This implies that certain elements of the sensitivity matrix P are known to be zeros. In addition, the transcription factor activities are modeled to be nonnegative, smooth, and periodic functions of time. Where transcription factors down regulate a gene, the elements of P have to be negative to account for this. In Section 8.2, this prior knowledge is formalized as constraints in a low-rank matrix approximation problem and optimization methods for solving the problem are developed.

1.3.8 Applications in computer vision

The data in computer vision problems has a hidden low-rank structure. The data matrix, however, depends nonlinearly on the data. This makes the resulting structured low-rank approximation problems challenging to solve. In this section, we show examples of conic section fitting and fundamental matrix estimation, where the structure of the data matrix is quadratic and bilinear.

Conic section fitting

In the applications reviewed so far, the low-rank approximation was applied to *linear* data modeling problem. Nonlinear data modeling, however, can also be solved by low-rank approximation methods. The key step—"linearizing" the problem—involves preprocessing the data by a nonlinear function defining the model structure. In the machine learning literature, where nonlinear data modeling is a common practice, the nonlinear function is called the *feature map* and the resulting modeling methods are referred to as *kernel methods*.

As a specific example, consider the problem of fitting data by a conic section, *i.e.*, given a set of points in the plane

$$\{d_1,\dots,d_N\} \subset \mathbb{R}^2, \quad \text{where } d_j = \begin{bmatrix} x_j \\ y_j \end{bmatrix},$$

find a conic section

$$\mathscr{B}(A,b,c) := \{ d \in \mathbb{R}^2 \mid d^\top A d + b^\top d + c = 0 \}. \tag{$\mathscr{B}(A,b,c)$}$$

that fits them. Here A is a 2×2 symmetric matrix, b is a 2×1 vector, and c is a scalar. A, b, and c are parameters defining the conic section. In order to avoid a trivial case $\mathscr{B} = \mathbb{R}^2$, it is assumed that at least one of the parameters A, b, or c is nonzero. The representation ($\mathscr{B}(A,b,c)$) is called implicit representation, because it imposes a relation (implicit function) on the elements x and y of d.

Defining the parameter vector

$$\theta := \begin{bmatrix} a_{11} & 2a_{12} & b_1 & a_{22} & b_2 & c \end{bmatrix},$$

and the extended data vector

$$d_{\text{ext}} := \begin{bmatrix} x^2 & xy & x & y^2 & y & 1 \end{bmatrix}^\top, \tag{d_{ext}}$$

then

$$d \in \mathscr{B}(\theta) = \mathscr{B}(A,b,c) \quad \iff \quad \theta d_{\text{ext}} = 0.$$

(The map $d \mapsto d_{\text{ext}}$, defined by (d_{ext}), is called the *feature map*.) Consequently, all data points $d_1,\dots,d_N$ are fitted by the model if

$$\theta \underbrace{\begin{bmatrix} d_{\text{ext},1} & \cdots & d_{\text{ext},N} \end{bmatrix}}_{D_{\text{ext}}} = 0 \quad \iff \quad \operatorname{rank}(D_{\text{ext}}) \leq 5. \tag{CSF}$$

Therefore, for $N > 5$ data points, exact fitting is equivalent to rank deficiency of the extended data matrix D_{ext}. For $N < 5$ data points, there is nonunique exact fit independently of the data. For $N = 5$ different points, the exact fitting conic section is unique, see Figure 1.4.

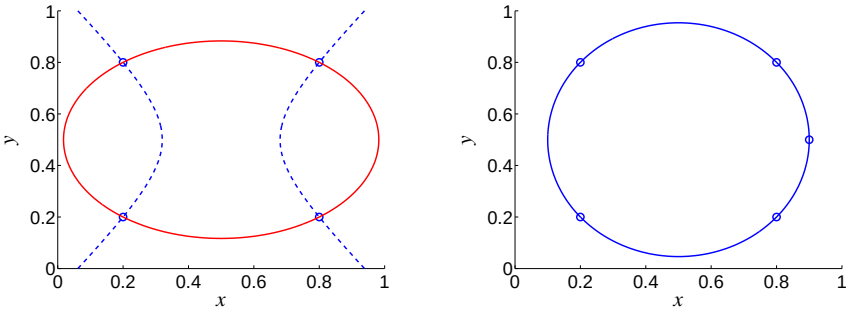


Fig. 1.4: Fitting a conic section to $N = 4$ points (left plot) leads to nonunique solution. Two exact fitting conic sections are shown in the figure with solid and dashed lines, respectively. With $N = 5$ different points (right plot) the solution is unique.

With $N > 5$ noisy data points, the extended data matrix D_{ext} is generically full rank, so that an exact fit does not exists. A problem called geometric fitting is to

minimize the sum of squared distances from the data points to the conic section. The problem is equivalent to *quadratically structured low-rank approximation*.

Generalization of the conic section fitting problem to algebraic curve fitting and solution methods for the latter are presented in Chapter 7.

Fundamental matrix estimation

A scene is captured by two cameras (stereo vision) and N matching pairs of points

$$\{u_1,\ldots,u_N\} \subset \mathbb{R}^2 \quad \text{and} \quad \{v_1,\ldots,v_N\} \subset \mathbb{R}^2$$

are located in the resulting images. Corresponding points u and v of the two images satisfy what is called an *epipolar constraint*

$$\begin{bmatrix} v^\top & 1 \end{bmatrix} F \begin{bmatrix} u \\ 1 \end{bmatrix} = 0, \quad \text{for some } F \in \mathbb{R}^{3\times 3}, \text{ with } \operatorname{rank}(F) = 2. \tag{EPI}$$

The matrix F, called *fundamental matrix*, characterizes the relative position and orientation of the cameras and does not depend on the pairs of points (u_i, v_i). Estimation of F from data is a calibration step in computer vision methods.

The epipolar constraint (EPI) is linear in F. Indeed, defining

$$d_{\text{ext}} := \begin{bmatrix} u_x v_x & u_x v_y & u_x & u_y v_x & u_y v_y & u_y & v_x & v_y & 1 \end{bmatrix}^\top \in \mathbb{R}^9, \tag{d'_{ext}}$$

where $u = \left[\begin{smallmatrix} u_x \\ u_y \end{smallmatrix}\right]$ and $v = \left[\begin{smallmatrix} v_x \\ v_y \end{smallmatrix}\right]$, (EPI) can be written as

$$\operatorname{vec}^\top(F) d_{\text{ext}} = 0.$$

Note that, as in the application for conic section fitting, the original data (u, v) is mapped to an extended data vector d_{ext} via a nonlinear function (a feature map). In this case, however, the function is *bilinear*.

The epipolar constraints for all data points, lead to the matrix equation

$$\operatorname{vec}^\top(F) D_{\text{ext}} = 0, \quad \text{where } D_{\text{ext}} := \begin{bmatrix} d_{\text{ext},1} & \cdots & d_{\text{ext},N} \end{bmatrix}. \tag{FME}$$

The rank constraint imposed on F implies that F is a nonzero matrix. Therefore, by (FME), for $N \geq 8$ data points, the extended data matrix D_{ext} is rank deficient. Moreover, the fundamental matrix F can be reconstructed up to a scaling factor from a vector in the left kernel of D_{ext}.

Noisy data with $N \geq 8$ data points generically gives rise to a full rank extended data matrix D_{ext}. The estimation problem is a *bilinearly structured low-rank approximation* problem with an additional constraint that $\operatorname{rank}(F) = 2$.

Summary of applications

The applications reviewed in Section 1.3 lead to a structured low-rank approximation problem. In this sense, they are related. The differences are in the type of the given data, the structure of the data matrix that is constructed from the given data, and the rank constraint, which has different interpretation in the different applications. Table 1.1 summarized this information, allowing an easy comparison. More applications are cited in the notes and references section on page 28.

Table 1.1: The rank of the data matrix has physical meaning in the applications.

application	data	data matrix	structure	rank =	ref.
approximate realization	impulse response H	$\mathscr{H}(H)$	Hankel	system's order	Sec. 2.2.2 Sec. 3.1 Sec. 5.1
stochastic realization	correlation function R	$\mathscr{H}(R)$	Hankel	system's order	Sec. 2.3
system identification	trajectory w of the system	$\mathscr{H}_{\mathtt{n}_{\max}+1}(w)$	Hankel	(SYSID)	Sec. 5.2
approximate GCD	polynomials p_{d} and q_{d}	$\mathscr{R}(p_{\mathrm{d}}, q_{\mathrm{d}})$	Sylvester	(GCD)	Sec. 5.3
array processing	array response $\big(w(1),\ldots,w(T)\big)$	$\begin{bmatrix} w(1) & \cdots & w(T) \end{bmatrix}$	unstructured	# of signal sources	—
multivariate calibration	spectral responses $\{d_1,\ldots,d_N\} \subset \mathbb{R}^{\mathtt{q}}$	$\begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$	unstructured	# of chemical components	—
factor analysis	test scores d_{ij}	$\begin{bmatrix} d_{ij} \end{bmatrix}$	unstructured	# of factors	—
language processing	term–document frequencies d_{ij}	$\begin{bmatrix} d_{ij} \end{bmatrix}$	unstructured	# of concepts	—
recommender system	some ratings d_{ij}	$\begin{bmatrix} d_{ij} \end{bmatrix}$	unstructured missing data	# of tastes	Sec. 4.1.4 Sec. 4.4
multidim. scaling	pair-wise distances d_{ij}	$\begin{bmatrix} d_{ij} \end{bmatrix}$	(MDS)	$\dim(x)+2$	—
microarray data analysis	gene expression levels d_{ij}	$\begin{bmatrix} d_{ij} \end{bmatrix}$	unstructured	# of transcript. factors	Sec. 8.2
conic section fitting	points $\{d_1,\ldots,d_N\} \subset \mathbb{R}^2$	(d_{ext}), (CSF)	quadratic	5	Ch. 7
fundamental matrix estimation	points $u_j, v_j \in \mathbb{R}^2$	(d'_{ext}), (FME)	bilinear	6	—

1.4 Overview of algorithms

The rank constraint in the low-rank approximation problem corresponds to the constraint in the data modeling problem that the data is fitted exactly by a linear model of bounded complexity. Therefore, the question of representing the rank constraint in the low-rank approximation problem corresponds to the question of choosing the model representation in the data fitting problem. Different representations lead to

- *optimization problems*, the relation among which may not be obvious;
- *algorithms*, which may have different convergence properties and efficiency; and
- *numerical software*, which may have different numerical robustness.

The virtues of the abstract, representation free, low-rank approximation problem formulation, are both *conceptual:* it clarifies the equivalence among different parameter optimization problems, and *practical:* it shows various ways of formulating one and the same high level data modeling problem as parameter optimization problems. On the conceptual level, the low-rank approximation problem formulation shows what one aims to achieve without a reference to implementation details. In particular, the model representation is such a detail, which is not needed for a high level formulation of data modeling problems. The representations, however, are unavoidable when one solves the problem analytically or numerically.

On the practical level, the low-rank problem formulation allows one to translate the abstract data modeling problem to different concrete parametrized problems by choosing the model representation. Different representations naturally lend themselves to different analytical and numerical methods. For example, a controllable linear time-invariant system can be represented by a transfer function, state space, convolution, *etc*. representations. The analysis tools related to these representations are rather different and, consequently, the obtained solutions differ despite of the fact that they solve the same abstract problem. Moreover, the parameter optimization problems, resulting from different model representations, lead to algorithms and numerical implementations, whose robustness properties and computational efficiency differ. Although, often in practice, there is no universally "best" algorithm or software implementation, having a wider set of available options is an advantage.

Independent of the choice of the rank representation only a few special low-rank approximation problems have analytic solutions. So far, the most important special case with an analytic solution is the unstructured low-rank approximation in the Frobenius norm. The solution in this case can be obtained from the singular value decomposition of the data matrix (Eckart–Young–Mirsky theorem). Extensions of this basic solution are problems known as generalized and restricted low-rank approximation, where some columns or, more generally submatrices of the approximation, are constrained to be equal to given matrices. The solutions to these problems are given by, respectively, the generalized and restricted singular value decompositions. Another example of low-rank approximation problem with analytic solution is the circulant structured low-rank approximation, where the solution is expressed in terms of the discrete Fourier transform of the data.

In general, low-rank approximation problems are NP-hard. There are three fundamentally different solution approaches for low-rank approximation:

- heuristic methods based on convex relaxations,
- local optimization methods, and
- global optimization methods.

From the class of heuristic methods the most popular ones are the subspace methods. The approach used in the subspace type methods is to relax the difficult low-rank approximation problem to a problem with an analytic solution in terms of the singular value decomposition, *e.g.*, ignore the structure constraint of a structured low-rank approximation problem. The subspace methods are found to be very effective in model reduction, system identification, and signal processing. The class of the subspace system identification methods is based on the unstructured low-rank approximation in the Frobenius norm (*i.e.*, singular value decomposition) while the original problems are Hankel structured low-rank approximation.

The methods based on local optimization split into two main categories:

- *alternating projections* and
- *variable projection*

type algorithms. Both alternating projections and variable projection exploit the bilinear structure of the low-rank approximation problems.

In order to explain the ideas underlining the alternating projections and variable projection methods, consider the optimization problem

$$\text{minimize} \quad \text{over } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \text{ and } L \in \mathbb{R}^{\mathtt{m}\times N} \quad \|D-PL\|_{\mathrm{F}}^2 \qquad (\mathrm{LRA}_P)$$

corresponding to low-rank approximation with an image representation of the rank constraint. The term PL is bilinear in the optimization variables P and L, so that for a fixed P, (LRA_P) becomes a linear least squares problem in L and vice verse, for a fixed L, (LRA_P) becomes a linear least squares problem in P. This suggests an iterative algorithm starting from an initial guess for P and L and alternatively solves the problem with one of the variables fixed. Since each step is a projection operation the method has the name alternating projections. It is globally convergent to a locally optimal solution of (LRA_P) with a linear convergence rate.

The analytic solution of (LRA_P) with respect to L for a fixed P gives us an equivalent cost function with P as an optimization variable. The original problem can then be solved by minimizing the equivalent cost function over P. The latter problem is unconstrained nonlinear optimization, for which standard optimization methods can be used. The elimination of L from the problem has the advantage of reducing the number of optimization variables. Evaluation of the cost function for a given P is a projection operation. In the course of the nonlinear minimization over P this variable changes, thus the name of the method—variable projection.

In the statistical literature, the alternating projections algorithm is given the interpretation of *expectation maximization*. The problem of computing the optimal approximation $\widehat{D} = PL$, given P is the expectation step and the problem of computing P, given L is the maximization step of the expectation maximization procedure.

Literate programming

> At first, I thought programming was primarily analogous to musical composition—to the creation of intricate patterns, which are meant to be performed. But lately I have come to realize that a far better analogy is available: Programming is best regarded as the process of creating *works of literature*, which are meant to be read.
>
> Knuth (1992, page ix)

The ideas presented in the book are best expressed as algorithms for solving data modeling problems. The algorithms, in turn, are practically useful when implemented in ready-to-use software. The gap between the theoretical discussion of data modeling methods and the practical implementation of these methods is bridged by using a literate programming style. The software implementation (MATLAB code) is interwoven in the text, so that the full implementation details are available in a human readable format and they come in the appropriate context of the presentation.

A literate program is composed of interleaved code segments, called chunks, and text. The program can be split into chunks in any way and the chunks can be presented in any order, deemed helpful for the understanding of the program. This allows us to focus on the logical structure of the program rather than the way a computer executes it. The actual computer executable code is *tangled* from a *web* of the code chunks by skipping the text and putting the chunks in the right order. In addition, literate programming allows us to use a powerful typesetting system such as LaTeX (rather than plain text) for the documentation of the code.

The `noweb` system for literate programming is used. Its main advantage over alternative systems is independence of the programming language being used.

Typographic conventions

The code is typeset in `small true type` font and consists of a number of code chunks. The code chunks begin with tags enclosed in angle brackets (*e.g.*, ⟨*code tag*⟩) and are sequentially numbered by the page number and a letter identifying them on the page. Thus the chunk's identification number (given to the left of the tag) is also used to locate the chunk in the text. For example, in order to make the results reproducible, in the simulation examples involving randomness, we first initialize the random number generator:

25 ⟨*initialize the random number generator* 25⟩≡ (123b 139c 143b 146a 147b 186a 216a)

```
randn('seed', 0); rand('seed', 0);
```

has identification number 25, locating the code as being on page 25.

If a chunk is included in, is a continuation of, or is continued by other chunk(s), its definition has references to the related chunk(s). The syntax convention for doing this is best explained by an example.

Example: Hankel matrix constructor

Consider the implementation of the (block) Hankel matrix constructor

$$\mathscr{H}_{i,j}(w) := \begin{bmatrix} w(1) & w(2) & \cdots & w(j) \\ w(2) & w(3) & \cdots & w(j+1) \\ \vdots & \vdots & & \vdots \\ w(i) & w(i+1) & \cdots & w(j+i-1) \end{bmatrix}, \qquad (\mathscr{H}_{i,j})$$

where $w = \big(w(1), \ldots, w(T)\big)$, with $w(t) \in \mathbb{R}^{\mathtt{q} \times N}$. The definition of the function, showing its input and output arguments, is

26a ⟨*Hankel matrix constructor* 26a⟩≡ 26c▷

```
function H = blkhank(w, i, j)
```

Defines:
`blkhank`, used in chunks 73c, 77, 118c, 138b, 143a, 145b, 147a, 149, 239a, 245b, and 250a.

(The reference to the right of the identification tag shows that the definition is continued in chunk number 26c.) The third input argument of `blkhank`—the number of block columns `j` is optional. Its default value is maximal number of block columns

$$j = T - i + 1.$$

26b ⟨*optional number of (block) columns* 26b⟩≡ (26d 27b)

```
if nargin < 3 | isempty(j), j = T - i + 1; end
if j <= 0, error('Not enough data.'), end
```

(References on the right of the identification tag now shows that this chunk is included in other chunks—the ones identified by the reference.)

Two cases are distinguished, depending on whether w is a vector ($N = 1$) or matrix ($N > 1$) valued trajectory.

26c ⟨*Hankel matrix constructor* 26a⟩+≡ ◁26a

```
if length(size(w)) == 3
  ⟨matrix valued trajectory w 26d⟩
else
  ⟨vector valued trajectory w 27b⟩
end
```

(The reference to the right of the identification tag shows that this chunk is a continuation of chunk 26a and is not followed by other chunks.)

- If w is a matrix valued trajectory, the input argument `w` should be a 3 dimensional tensor, constructed as follows $\mathtt{w}(:, :, \mathtt{t}) = w(t)$.

26d ⟨*matrix valued trajectory w* 26d⟩≡ (26c) 26e▷

```
[q, N, T] = size(w);
⟨optional number of (block) columns 26b⟩
```

(This chunk is both included and followed by other chunks.) In this case, the construction of the Hankel matrix $H_{i,j}(w)$ is done explicitly by a double loop:

26e ⟨*matrix valued trajectory w* 26d⟩+≡ (26c) ◁26d

```
H = zeros(i * q, j * N);
for ii = 1:i
```

```
    for jj = 1:j
      H(((ii - 1) * q + 1):(ii * q), ...
        ((jj - 1) * N + 1):(jj * N)) = w(: ,:, ii + jj - 1);
    end
  end
```

- If w is a vector valued trajectory , the input argument `w` should be a matrix formed as follows $\mathtt{w}(:, \mathtt{t}) = w(t)$, however, since T must be greater than or equal to the number of variables $\mathtt{q} := \dim\big(w(t)\big)$, when `w` has more rows than columns, the input is treated as $\mathtt{w}(\mathtt{t}, :) = w^{\top}(t)$.

27a ⟨*reshape w and define* q, *T* 27a⟩≡ (27b 78 117b 145a 146c 148 245b)

```
[q, T] = size(w); if T < q, w = w'; [q, T] = size(w); end
```

27b ⟨*vector valued trajectory w* 27b⟩≡ (26c) 27c▷

⟨*reshape w and define* q, *T* 27a⟩
⟨*optional number of (block) columns* 26b⟩

The reason to consider the case of a vector valued w separately is that in this case the construction of the Hankel matrix $\mathscr{H}_{i,j}(w)$ can be done with a single loop.

27c ⟨*vector valued trajectory w* 27b⟩+≡ (26c) ◁27b

```
H = zeros(i * q, j);
for ii = 1:i
  H(((ii - 1) * q + 1):(ii * q), :) = w(:, ii:(ii + j - 1));
end
```

We use `blkhank` is system identification problems, where $i \ll j$. MATLAB, being an interpreted language, executes `for` loops slowly, so that the reduction to a single `for` loop on the block rows of the matrix leads to decrease of the execution time compared to an implementation with two nested `for` loops.

1.5 Notes and references

Classical and behavioral paradigms for data modeling

Methods for solving overdetermined systems of linear equations (*i.e.*, data modeling methods using the classical input/output paradigm) are reviewed in Appendix A. The behavioral paradigm for data modeling was developed by J. C. Willems in the early 80's from "a need to put a clear and rational foundation under the problem of obtaining models from time series" (Willems, 1986, page 561). It became firmly established with the publication of the three-part paper (Willems, 1986, 1987). Other landmark publications on the behavioral paradigm are (Willems, 1989, 1991, 2007).

The numerical linear algebra problem of low-rank approximation is a computational tool for data modeling, which fits the behavioral paradigm as "a hand fits a glove". Historically the low-rank approximation problem is closely related to the singular value decomposition, which is a computational method for low-rank approximations and is used in algorithms for data modeling. A historical account of the development of the singular value decomposition is given in (Stewart, 1993). The Eckart–Young–Mirsky theorem is proven in (Eckart and Young, 1936).

Applications

Realization and system identification problems are presented in Sections 2.2.2, 3.1, and 5.2. Direction of arrival and adaptive beamforming problems are discussed in (Krim and Viberg, 1996; Kumaresan and Tufts, 1983). Low-rank approximation methods for estimation of concentrations in chemometrics are presented in (Wentzell, 2014; Wentzell et al, 1997). An early reference on the polynomial approximate common factor problem is (Karmarkar and Lakshman, 1998). Other computer algebra problems that reduce to structured low-rank approximation are discussed in (Botting, 2004; Usevich, 2014).

Many problems for information retrieval in machine learning, see, *e.g.*, (Bishop, 2006; Fierro and Jiang, 2005; Shawe-Taylor and Cristianini, 2004), are low-rank approximation problems and corresponding solution techniques developed by the machine learning community are methods for low-rank approximation. For example, clustering problems have been related to low-rank approximation problems in (Cohen et al, 2015; Ding and He, 2004; Kiers, 2002; Vichia and Saporta, 2009).

Machine learning problems, however, are often posed in a stochastic estimation setting which obscures their deterministic approximation interpretation. For example, principal component analysis (Jackson, 2003; Jolliffe, 2002; Vidal et al, 2016) and unstructured low-rank approximation with Frobenius norm are equivalent optimization problems, however, the former is defined in a statistical setting while the latter is defined as an optimization problem. From the numerical linear algebra perspective, principal component analysis provides a statistical interpretation of the low-rank approximation problem. From a statistical estimation perspective low-rank approximation provides computational methods for principal component analysis.

The conic section fitting problem has extensive literature, see Chapter 7 and the tutorial paper (Zhang, 1997). The kernel principal component analysis method is developed in the machine learning and statistics literature (Schölkopf et al, 1999). Despite of the close relation between kernel principal component analysis and conic section fitting, the corresponding literature are disjoint.

Closely related to the estimation of the fundamental matrix problem in two-view computer vision is the shape from motion problem (Ma et al, 2004; Tomasi and Kanade, 1993). Other problems in computer vision are related or use low-rank approximation methods, see (Vidal et al, 2005) and the edited book (Fu, 2014).

Matrix factorization techniques are used in microarray data analysis in (Alter and Golub, 2006; Kim and Park, 2007). Alter and Golub (2006) propose a principal component projection to visualize high dimensional gene expression data and show that known biological aspects of the data are visible in the two dimensional subspace defined by the first two principal components. Since 2012, low-rank approximation and matrix completion methods became the state-of-the-art approach in medical imaging (Bydder et al, 2017; Dong et al, 2014; Haldar, 2014; Nguyen et al, 2013).

Distance problems

The low-rank approximation problem aims at finding the "smallest" correction of a given matrix that makes the corrected matrix rank deficient. This is a special case of a distance problems: find the "nearest" matrix with a specified property to a given matrix. For an overview of distance problems, see (Higham, 1989). In (Byers, 1988), an algorithm for computing the distance of a stable matrix (Hurwitz matrix in the case of continuous-time and Schur matrix in the case of discrete-time linear time-invariant dynamical system) to the set of unstable matrices is presented. Stability radius for structured perturbations and its relation to the algebraic Riccati equation is presented in (Hinrichsen and Pritchard, 1986).

Structured linear algebra

Related to the topic of distance problems is the grand idea that the whole linear algebra (solution of systems of equations, matrix factorization, *etc*.) can be generalized to uncertain data. The uncertainty is described as structured perturbation on the data and a solution of the problem is obtained by correcting the data with a correction of the smallest size that renders the problem solvable for the corrected data. Some early references on the topic of structured linear algebra are (Calafiore and Ghaoui, 2014; Chandrasekaran et al, 1998; El Ghaoui and Lebret, 1997).

Structured pseudospectra

Let $\lambda(A)$ be the set of eigenvalues of $A \in \mathbb{C}^{n\times n}$ and $\mathscr{M}$ be a set of structured matrices

$$\mathscr{M} := \{\, \mathscr{S}(p) \mid p \in \mathbb{R}^{n_p} \,\},$$

with a given structure specification $\mathscr{S}$. The ε-structured pseudospectrum (Graillat, 2006; Trefethen and Embree, 1999) of A is defined as the set

$$\lambda_\varepsilon(A) := \{\, z \in \mathbb{C} \mid z \in \lambda(\widehat{A}),\ \widehat{A} \in \mathscr{M},\ \text{and } \|A - \widehat{A}\|_2 \le \varepsilon \,\}.$$

Using the structured pseudospectra, one can determine the structured distance of A to singularity as the minimum of the following optimization problem:

$$\text{minimize} \quad \text{over } \widehat{A} \quad \|A - \widehat{A}\|_2 \quad \text{subject to} \quad \widehat{A} \text{ is singular and } \widehat{A} \in \mathscr{M}.$$

This is a special structured low-rank approximation problem for squared data matrix and rank reduction by one. Related to structured pseudospectra is the structured condition number problem for a system of linear equations, see (Rump, 2003).

Statistical properties

Related to low-rank approximation are the *orthogonal regression* (Gander et al, 1994), *errors-in-variables* (Gleser, 1981), and *measurement errors* methods in the statistical literature (Carroll et al, 1995; Cheng and Van Ness, 1999). Classic papers on the univariate errors-in-variables problem are (Adcock, 1877, 1878; Koopmans, 1937; Madansky, 1959; Pearson, 1901; York, 1966). Closely related to the errors-in-variables framework for low-rank approximation is the probabilistic principal component analysis framework of (Tipping and Bishop, 1999). For a detailed presentation of the probabilistic principal component analysis and its connection to low-rank approximation, see (Vidal et al, 2016).

Reproducible research

> An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures. Buckheit and Donoho (1995)

The reproducible research concept is at the core of all sciences. In applied fields such as data modeling, however, algorithms' implementation, availability of data, and reproducibility of the results obtained by the algorithms on data are often neglected. This leads to a situation, described in (Buckheit and Donoho, 1995) as a scandal. See also (Kovacevic, 2007).

A quick and easy way of making computational results obtained in MATLAB reproducible is to use the function `publish`. Better still, the code and the obtained results can be presented in a literate programming style.

Literate programming

The creation of the literate programming is a byproduct of the TEX project, see (Knuth, 1984, 1992). The original system, called `web` is used for documentation of the TEX program (Knuth, 1986) and is for the Pascal language. Later a version `cweb` for the C language was developed. The `web` and `cweb` systems are followed by many other systems for literate programming that target specific languages. Unfortunately this leads to numerous literate programming dialects.

The `noweb` system for literate programming created by N. Ramsey is not restricted to a specific programming language and text processing system. A tutorial introduction is given in (Ramsey, 1994). The `noweb` syntax is also adopted in the org-mode of Emacs (Dominik, 2010)—a package for keeping structured notes that includes support for organization and automatic evaluation of computer code.

Exercises

> Actively doing problems takes so much more time than passively reading the text, that it is always tempting to assume that exercises tucked away at the end of the chapter are a kind of optional appendix. ... Nothing could be further from the truth.
>
> (Gardiner, 1982, Page viii)

1.1 (Geometric interpretation of rank-1 approximation). Show that the rank-1 approximation problems (lra_P) and (lra_R) on page 5 minimize the sum of the squared orthogonal distances from the data points $d_1, \ldots, d_N$ to the fitting line $\mathscr{B} = \ker(R) = \text{image}(P)$ over all lines passing through the origin.

1.2 (Quadratically constrained problem, equivalent to rank-1 approximation). Show that (lra_P) is equivalent to the quadratically constrained optimization problem

$$\text{minimize } M(P) \quad \text{subject to} \quad P^\top P = 1, \tag{lra'_P}$$

where

$$M(P) = \text{trace}\left(D^\top (I_2 - PP^\top) D\right).$$

1.3 (Line fitting by rank-1 approximation). Plot the cost function $M(P)$ for

$$d_1 = \begin{bmatrix} -2 \\ 1 \end{bmatrix}, \quad d_2 = \begin{bmatrix} -1 \\ 4 \end{bmatrix}, \quad d_3 = \begin{bmatrix} 0 \\ 6 \end{bmatrix}, \quad d_4 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \quad d_5 = \begin{bmatrix} 2 \\ 1 \end{bmatrix},$$
$$d_6 = \begin{bmatrix} 2 \\ -1 \end{bmatrix}, \quad d_7 = \begin{bmatrix} 1 \\ -4 \end{bmatrix}, \quad d_8 = \begin{bmatrix} 0 \\ -6 \end{bmatrix}, \quad d_9 = \begin{bmatrix} -1 \\ -4 \end{bmatrix}, \quad d_{10} = \begin{bmatrix} -2 \\ -1 \end{bmatrix}.$$

over all P on the unit circle $P^\top P = 1$. Find the minimum points of M.

1.4 (Analytic solution of a rank-1 approximation problem). Show that for the data in Problem 1.3,

$$M(P) = P^\top \begin{bmatrix} 140 & 0 \\ 0 & 20 \end{bmatrix} P.$$

Argue that the minimum of M for a P on the unit circle is 20 and is achieved for $P^{*,1} = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$ and $P^{*,2} = \left[\begin{smallmatrix} 0 \\ -1 \end{smallmatrix}\right]$. Compare with the approach of Problem 1.3.

1.5 (Litterate program for Sylvester matrix construction).

1. Download and install `noweb` from `www.cs.tufts.edu/~nr/noweb/`.
2. Using `noweb`, write a literate program in MATLAB for construction of the Sylvester matrix $\mathscr{R}(p,q)$, defined in ($\mathscr{R}$) on page 11. Test the program by finding the degree `d` of the greatest common divisor of the polynomials

$$p(z) = 1 + 3z + 5z^2 + 3z^3 \quad \text{and} \quad q(z) = 3 + 5z + 3z^2 + z^3.$$

References

Adcock R (1877) Note on the method of least squares. The Analyst 4:183–184

Adcock R (1878) A problem in least squares. The Analyst 5:53–54

Alter O, Golub GH (2006) Singular value decomposition of genome-scale mRNA lengths distribution reveals asymmetry in RNA gel electrophoresis band broadening. Proc Nat Academy of Sci 103:11,828–11,833

Baez J (2010) This week's finds in mathematical physics. `http://math.ucr.edu/home/baez/week294.html`

Bishop C (2006) Pattern recognition and machine learning. Springer

Botting B (2004) Structured total least squares for approximate polynomial operations. Master's thesis, School of Computer Science, University of Waterloo

Buckheit J, Donoho D (1995) Wavelets and statistics, Springer-Verlag, chap Wavelab and reproducible research

Bydder M, Rapacchi S, Girard O, Guye M, Ranjeva JP (2017) Trimmed autocalibrating K-space estimation based on structured matrix completion. Magnetic Resonance Imaging 43:88–94

Byers R (1988) A bisection method for measuring the distance of a stable matrix to the unstable matrices. SIAM J Sci Stat Comput 9(5):875–881

Calafiore G, Ghaoui LE (2014) Optimization models. Cambridge University Press

Carroll R, Ruppert D, Stefanski L (1995) Measurement Error in Nonlinear Models. Chapman & Hall/CRC, London

Chandrasekaran S, Golub G, Gu M, Sayed A (1998) Parameter estimation in the presence of bounded data uncertainties. SIAM J Matrix Anal Appl 19:235–252

Cheng C, Van Ness JW (1999) Statistical regression with measurement error. London: Arnold

Cohen M, Elder S, Musco C, Musco C, Persu M (2015) Dimensionality reduction for k-means clustering and low rank approximation. In: Proc. 47th Annual ACM Symposium on Theory of Computing, ACM, pp 163–172

Ding C, He X (2004) K-means clustering via principal component analysis. In: Proc. Int. Conf. Machine Learning, pp 225–232

Dominik C (2010) The org mode 7 reference manual. Network theory ltd, URL `http://orgmode.org/`

Dong W, Shi G, Li X, Ma Y, Huang F (2014) Compressive sensing via nonlocal low-rank regularization. IEEE Trans on Image Processing 23(8):3618–3632

Eckart G, Young G (1936) The approximation of one matrix by another of lower rank. Psychometrika 1:211–218

El Ghaoui L, Lebret H (1997) Robust solutions to least-squares problems with uncertain data. SIAM J Matrix Anal Appl 18:1035–1064

Fierro R, Jiang E (2005) Lanczos and the Riemannian SVD in information retrieval applications. Numer Linear Algebra Appl 12:355–372

Fu Y (ed) (2014) Low-Rank and Sparse Modeling for Visual Analysis. Springer

Gander W, Golub G, Strebel R (1994) Fitting of circles and ellipses: Least squares solution. BIT 34:558–578

Gardiner A (1982) Infinite Processes: Background to Analysis. Springer-Verlag

Gleser L (1981) Estimation in a multivariate errors-in-variables regression model: Large sample results. The Annals of Statistics 9(1):24–44
Graillat S (2006) A note on structured pseudospectra. J Comput Appl Math 191:68–76
Haldar J (2014) Low-rank modeling of local *k*-space neighborhoods for constrained MRI. IEEE Trans on Medical Imaging 33(3):668–681
Halmos P (1985) I want to be a mathematician: An automathography. Springer
Higham N (1989) Matrix nearness problems and applications. In: Gover M, Barnett S (eds) Applications of Matrix Theory, Oxford University Press, pp 1–27
Hinrichsen D, Pritchard AJ (1986) Stability radius for structured perturbations and the algebraic Riccati equation. Control Lett 8:105–113
Jackson J (2003) A User's Guide to Principal Components. Wiley
Jolliffe I (2002) Principal component analysis. Springer-Verlag
Karmarkar N, Lakshman Y (1998) On approximate GCDs of univariate polynomials. In: Watt S, Stetter H (eds) J. Symbolic Comput., vol 26, pp 653–666
Kiers H (2002) Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems. Comput Stat Data Anal 41:157–170
Kim H, Park H (2007) Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. Bioinformatics 23:1495–1502
Knuth D (1984) Literate programming. Comput J 27(2):97–111
Knuth D (1986) Computers & Typesetting, Volume B: TeX: The Program. Addison-Wesley
Knuth D (1992) Literate programming. Cambridge University Press
Koopmans T (1937) Linear regression analysis of economic time series, vol 20. DeErven F. Bohn
Kovacevic J (2007) How to encourage and publish reproducible research. In: Proc. IEEE Int. Conf. Acoustics, Speech Signal Proc., pp 1273–1276
Krim H, Viberg M (1996) Two decades of array signal processing research. IEEE Signal Proc Magazine 13:67–94
Kumaresan R, Tufts D (1983) Estimating the angles of arrival of multiple plane waves. IEEE Trans Aerospace Electronic Systems 19(1):134–139
Ma Y, Soatto S, Kosecká J, Sastry S (2004) An Invitation to 3-D Vision, Interdisciplinary Applied Mathematics, vol 26. Springer
Madansky A (1959) The fitting of straight lines when both variables are subject to error. J Amer Statist Assoc 54:173–205
Markovsky I (2008) Structured low-rank approximation and its applications. Automatica 44(4):891–909
Markovsky I (2014) Recent progress on variable projection methods for structured low-rank approximation. Signal Processing 96PB:406–419
Markovsky I, Van Huffel S (2007) Overview of total least squares methods. Signal Processing 87:2283–2302
Markovsky I, Willems JC, Van Huffel S, De Moor B (2006) Exact and Approximate Modeling of Linear Systems: A Behavioral Approach. SIAM

Nguyen H, Peng X, Do M, Liang Z (2013) Denoising MR spectroscopic imaging data with low-rank approximations. IEEE Trans Biomed Eng 60(1):78–89
Pearson K (1901) On lines and planes of closest fit to points in space. Philos Mag 2:559–572
Ramsey N (1994) Literate programming simplified. IEEE Software 11:97–105
Rump S (2003) Structured perturbations part I: Normwise distances. SIAM J Matrix Anal Appl 25:1–30
Schölkopf B, Smola A, Müller K (1999) Kernel principal component analysis., MIT Press, Cambridge, MA, pp 327–352
Shawe-Taylor J, Cristianini N (2004) Kernel Methods for Pattern Analysis. Cambridge University Press
Stewart GW (1993) On the early history of the singular value decomposition. SIAM Review 35(4):551–566
Tipping M, Bishop C (1999) Probabilistic principal component analysis. J R Stat Soc B 61(3):611–622
Tomasi C, Kanade T (1993) Shape and motion from image streames: A factorization method. Proc Natl Adadem Sci USA 90:9795–9802
Trefethen LN, Embree M (1999) Spectra and pseudospectra: The behavior of non-normal matrices and operators. Princeton University Press
Usevich K (2014) Decomposing multivariate polynomials with structured low-rank matrix completion. In: Proceedings of the 21th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2014)
Vichia M, Saporta G (2009) Clustering and disjoint principal component analysis. Comput Stat Data Anal 53:3194–3208
Vidal R, Ma Y, Sastry S (2005) Generalized principal component analysis (gpca). IEEE Trans Pattern Anal Machine Intelligence 27(12):1945–1959
Vidal R, Ma Y, Sastry S (2016) Generalized Principal Component Analysis. Springer
Wentzell P (2014) Measurement errors in multivariate chemical data. Journal of the Brazilian Chemical Society 25(2):183–196
Wentzell P, Andrews D, Hamilton D, Faber K, Kowalski B (1997) Maximum likelihood principal component analysis. J Chemometrics 11:339–366
Willems JC (1986) From time series to linear system—Part I. Finite dimensional linear time invariant systems. Automatica 22(5):561–580
Willems JC (1986, 1987) 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
Willems JC (1989) Models for dynamics. Dynamics reported 2:171–269
Willems JC (1991) Paradigms and puzzles in the theory of dynamical systems. IEEE Trans Automat Control 36(3):259–294
Willems JC (2007) The behavioral approach to open and interconnected systems: Modeling by tearing, zooming, and linking. Control Systems Magazine 27:46–99
York D (1966) Least squares fitting of a straight line. Can J Physics 44:1079–1086
Zhang Z (1997) Parameter estimation techniques: A tutorial with application to conic fitting. Image Vision Comp J 15(1):59–76

Part I
Linear modeling problems

Chapter 2
From data to models

> *. . . whenever we have two different representations of the same thing we can learn a great deal by comparing representations and translating descriptions from one representation into the other. Shifting descriptions back and forth between representations can often lead to insights that are not inherent in either of the representations alone.*
>
> *Abelson and diSessa (1986, page 105)*

Section 1.2 presented as a motivating example an equivalence between line fitting and rank-one approximation. Section 1.3 listed examples from different applications, where the problems reduce to low-rank approximation. This lead us to claim that "Behind every data modeling problem there is a (hidden) low-rank approximation problem." In the rest of the book, we pose and solve multivariable linear static, dynamic linear time-invariant, and nonlinear data modeling problems in the low-rank setting. The advantage of this approach is that, one algorithm (and software implementation) can be used to solve a variety of practical data modeling problems.

In order to state the general data modeling problem considered in the book, we need the notions of a model, model complexity, and lack of fit between the data and a model. Therefore, first, we define rigorously the basic notion of a mathematical model. Important questions considered are: "How to represent a model by equations?" and "How to convert one representation into another one?". When two models fit the data equally well, we prefer the simpler model. For this purpose, we define the notion of model complexity. Two principles—misfit and latency—used to quantify the lack of fit between the data and a model are introduced. In a stochastic setting, the misfit principle leads to the errors-in-variables modeling problems and the latency principle leads to the static regression and dynamic ARMAX problems.

In defining the notions of model, representation, and model complexity, first are considered linear static models, in which case the representation is an algebraic equation. Then, the results are extended to linear time-invariant models, in which case the model representation is a differential or difference equation. Finally, the ideas are applied to stochastic systems, *i.e.*, systems driven by stochastic processes.

2.1 Static model representations

First, we define the notions of a linear static model as well as kernel, image, and input/output representations of a model. Then, we show the transitions among the three representations. The computational details of the various transitions from one representation to another are presented in a literate programming style, using the MATLAB language. Finally, we define the notion of a model complexity.

2.1.1 Kernel, image, and input/output representations

A *linear static model* with $\mathtt{q}$ variables is a subspace of the $\mathtt{q}$-dimensional space $\mathbb{R}^{\mathtt{q}}$. We denote the set of linear static models with $\mathtt{q}$ variables by $\mathscr{L}_0^{\mathtt{q}}$. Three basic representations of a linear static model $\mathscr{B} \subseteq \mathbb{R}^{\mathtt{q}}$ are the kernel, image, and input/output:

- *kernel representation* with parameter $R \in \mathbb{R}^{\mathtt{g}\times\mathtt{q}}$

$$\mathscr{B} = \ker(R) := \{\, d \in \mathbb{R}^{\mathtt{q}} \mid Rd = 0 \,\}, \qquad (\mathrm{KER}_0)$$

- *image representation* with parameter $P \in \mathbb{R}^{\mathtt{q}\times\mathtt{s}}$

$$\mathscr{B} = \mathrm{image}(P) := \{\, d = P\ell \in \mathbb{R}^{\mathtt{q}} \mid \ell \in \mathbb{R}^{\mathtt{s}} \,\}, \qquad (\mathrm{IMAGE}_0)$$

- *input/output representation* with parameters $X \in \mathbb{R}^{\mathtt{m}\times\mathtt{p}}$ and a permutation Π

$$\mathscr{B}_{\mathrm{i/o}}(X,\Pi) := \{\, d = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \in \mathbb{R}^{\mathtt{q}} \mid u \in \mathbb{R}^{\mathtt{m}},\ y = X^\top u \,\}. \qquad (\mathrm{I/O}_0)$$

If the parameter Π in an input/output representation is not specified, then by default it is assumed to be the identity matrix $\Pi = I_{\mathtt{q}}$, *i.e.*, the first $\mathtt{m}$ variables are assumed to be inputs and the other $\mathtt{p} := \mathtt{q} - \mathtt{m}$ variables assumed to be are outputs.

In the representation (IMAGE_0), the columns of P are *generators* of the model $\mathscr{B}$, *i.e.*, they span the model. In the representation (KER_0), the rows of R are *annihilators* of $\mathscr{B}$, *i.e.*, they are orthogonal to $\mathscr{B}$. The parameters R and P are not unique.

1. Linearly dependent generators and annihilators can be added, respectively, to an existing set of annihilators and generators of $\mathscr{B}$, keeping the model the same.
2. A change of basis transformation can be applied to the generators or annihilators

$$\ker(R) = \ker(UR), \quad \text{for all } U \in \mathbb{R}^{\mathtt{g}\times\mathtt{g}}, \text{ such that } \det(U) \neq 0,$$
$$\mathrm{image}(P) = \mathrm{image}(PV), \quad \text{for all } V \in \mathbb{R}^{\mathtt{s}\times\mathtt{s}}, \text{ such that } \det(V) \neq 0.$$

The smallest possible number of generators, *i.e.*, $\mathrm{col\,dim}(P)$, such that (IMAGE_0) holds is invariant of the representation and is equal to $\mathtt{m} := \dim(\mathscr{B})$—the dimension of $\mathscr{B}$. Integers, such as $\mathtt{m}$ and $\mathtt{p} := \mathtt{q} - \mathtt{m}$ that are invariant of the representation, characterize properties of the model and are called *model invariants*. The integers $\mathtt{m}$ and $\mathtt{p}$ have data modeling interpretation as number of inputs and number of outputs,

respectively. Indeed, m variables are free (unassigned by the model) and the other p variables are determined by the model and the inputs. The number of inputs and outputs of the model $\mathscr{B}$ are denoted by $\mathtt{m}(\mathscr{B})$ and $\mathtt{p}(\mathscr{B})$, respectively.

The model class of linear static models with q variables, at most m of which are inputs is denoted by $\mathscr{L}^{\mathtt{q}}_{\mathtt{m},0}$. With $\operatorname{col\,dim}(P) = \mathtt{m}$, the columns of P form a basis for $\mathscr{B}$. The smallest possible $\operatorname{row\,dim}(R)$, such that $\ker(R) = \mathscr{B}$ is invariant of the representation and is equal to the number of outputs of $\mathscr{B}$. With $\operatorname{row\,dim}(R) = \mathtt{p}$, the rows of R form a basis for the orthogonal complement $\mathscr{B}^{\perp}$ of $\mathscr{B}$. Therefore, without loss of generality we can assume that $P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}}$ and $R \in \mathbb{R}^{\mathtt{p}\times\mathtt{q}}$.

In general, many input/output partitions of the variables d are possible. Let $\ker(R)$ and $\operatorname{image}(P)$ be minimal representations of model $\mathscr{B} \in \mathscr{L}^{\mathtt{p}+\mathtt{m}}_{\mathtt{m},0}$. Choosing an input/output partition amounts to choosing a full rank $\mathtt{p}\times\mathtt{p}$ submatrix of R or a full rank $\mathtt{m}\times\mathtt{m}$ submatrix of P. In some data modeling problems, there is no a priori reason to prefer one partition of the variables over another. For such problems, the classical setting posing the problem as an overdetermined system of linear equations $AX \approx B$ is not a natural starting point.

2.1.2 *Transition among input/output, kernel, and image representations*

The transition from one model representation to another gives insight into the properties of the model. These *analysis problems* need to be solved before the more complicated modeling problems are considered. The latter can be viewed as *synthesis problems* since the model is created or synthesised from data and prior knowledge.

If the parameters R, P, and (X,Π) describe the same system $\mathscr{B}$, then they are related. We show the relations that the parameters must satisfy as well as code that does the transition from one representation to another. Before we describe the transitions among the parameters, however, we need an efficient way to store and multiply by permutation matrices and a tolerance for computing rank numerically.

Input/output partition of the variables

In the input/output representation $\mathscr{B}_{\mathrm{i/o}}(X,\Pi)$, the partitioning of the variables $d \in \mathbb{R}^{\mathtt{q}}$ into inputs $u \in \mathbb{R}^{\mathtt{m}}$ and outputs $y \in \mathbb{R}^{\mathtt{p}}$ is specified by a permutation matrix Π,

$$d \underset{\Pi}{\overset{\Pi^{-1}=\Pi^{\top}}{\rightleftarrows}} \begin{bmatrix} u \\ y \end{bmatrix}, \quad d = \Pi \begin{bmatrix} u \\ y \end{bmatrix}, \quad \begin{bmatrix} u \\ y \end{bmatrix} = \Pi^{\top} d.$$

In the software implementation, however, it is more convenient (as well as more memory and computation efficient) to specify the partitioning by a vector

$$\pi := \Pi \operatorname{col}(1,\ldots,\mathtt{q}) \in \{1,\ldots,\mathtt{q}\}^{\mathtt{q}}.$$

The vector π contains the same information as the matrix Π. We can reconstruct Π from π by permutation of the rows of the identity matrix. (In the code io is the variable corresponding to the vector π and Pi is the variable corresponding to Π.)

40a ⟨$\pi \mapsto \Pi$ 40a⟩≡ (40c)

```
Pi = eye(length(io)); Pi = Pi(io,:);
```

Permuting the elements of a vector is done more efficiently by direct reordering of the elements, instead of a matrix-vector multiplication. If d is a variable corresponding to a vector d, then d(io) corresponds to the vector Πd.

The default value for Π is the identity matrix I, corresponding to first m variables of d being inputs and the remaining p variables outputs, *i.e.*, $d = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$.

40b ⟨*default input/output partition* 40b⟩≡ (42 43b)

```
if ~exist('io') || isempty(io), io = 1:q; end
```

In case the inverse permutation $\Pi^{-1} = \Pi^{\top}$ is needed, it can be found from

$$\pi' = \Pi^{\top} \operatorname{col}(1,\ldots,\mathtt{q}) \in \{1,\ldots,\mathtt{q}\}^{\mathtt{q}}.$$

In the code inv_io is the variable corresponding to the vector π' and the transition from the original variables d to the partitioned variables uy = [u; y] via io and inv_io is done by the following indexing operations:

$$\mathtt{d} \underset{\mathtt{io}}{\overset{\mathtt{io\_inv}}{\rightleftarrows}} \mathtt{uy}\ , \quad \mathtt{d} = \mathtt{uy(io)}, \quad \mathtt{uy} = \mathtt{d(io\_inv)}.$$

40c ⟨*inverse permutation* 40c⟩≡ (44a)

```
⟨π ↦ Π 40a⟩, inv_io = (1:length(io)) * Pi;
```

Tolerance for rank computation

In the computation of an input/output representation from a given kernel or image representation of a model (as well as in the computation of the models' complexity), we need to compute rank of a matrix. Numerically this is an ill-posed problem. Arbitrary small perturbations of the matrix's elements may (generically will) change the rank. A solution to this problem is to replace rank with "numerical rank" defined as

$$\operatorname{num\,rank}(A,\varepsilon) := \text{number of singular values of } A \text{ greater than } \varepsilon, \qquad (\text{num rank})$$

where $\varepsilon \in \mathbb{R}_+$ is a user defined tolerance. Note that

$$\operatorname{num\,rank}(A,0) = \operatorname{rank}(A),$$

i.e., by taking the tolerance to be equal to zero, the numerical rank reduces to the theoretical rank. A nonzero tolerance ε makes the numerical rank robust to perturbations of size (measured in the induced 2-norm) less than ε. Therefore, ε reflects the size of the expected errors in the matrix. The default tolerance is set to a small

value, which corresponds to numerical errors due to a double precision arithmetic. (In the code `tol` is the variable corresponding to ε.)

41 ⟨*default tolerance* `tol` 41⟩≡ (43 44c 74a)
```
if ~exist('tol') || isempty(tol), tol = 1e-12; end
```

Note 2.1. The numerical rank definition (num rank) is the solution of an unstructured rank minimization problem: find a matrix $\widehat{A}$ of minimal rank, such that $\|A-\widehat{A}\|_2 < \varepsilon$.

From input/output representation to kernel or image representations

Consider a linear static model $\mathscr{B} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},0}$, defined by an input/output representation $\mathscr{B}_{\mathrm{i/o}}(X,\Pi)$. From $y = X^\top u$, we have

$$\begin{bmatrix} X^\top & -I \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} = 0$$

or since $d = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$,

$$\underbrace{\begin{bmatrix} X^\top & -I \end{bmatrix} \Pi^\top}_{R} \underbrace{\Pi \begin{bmatrix} u \\ y \end{bmatrix}}_{d} = 0.$$

Therefore, the matrix

$$R = \begin{bmatrix} X^\top & -I \end{bmatrix} \Pi^\top \qquad ((X,\Pi) \mapsto R)$$

is a parameter of a kernel representations of $\mathscr{B}$, *i.e.*,

$$\mathscr{B}_{\mathrm{i/o}}(X,\Pi) = \ker\left(\begin{bmatrix} X^\top & -I \end{bmatrix} \Pi^\top \right) = \mathscr{B}.$$

Moreover, the representation is minimal because R has full row rank.

Similarly, a minimal image representation is derived from the input/output representation as follows. From $y = X^\top u$,

$$\begin{bmatrix} u \\ y \end{bmatrix} = \begin{bmatrix} I \\ X^\top \end{bmatrix} u,$$

so that, using $d = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$,

$$d = \Pi \begin{bmatrix} u \\ y \end{bmatrix} = \Pi \begin{bmatrix} I_{\mathtt{m}} \\ X^\top \end{bmatrix} u =: Pu.$$

Therefore, the matrix

$$P = \Pi \begin{bmatrix} I \\ X^\top \end{bmatrix} \qquad ((X,\Pi) \mapsto P)$$

is a parameter of an image representations of $\mathscr{B}$, *i.e.*,

$$\mathscr{B}_{\mathrm{i/o}}(X,\Pi) = \mathrm{image}\left(\Pi \begin{bmatrix} I_{\mathtt{m}} \\ X^\top \end{bmatrix} \right) = \mathscr{B}.$$

The representation is minimal because P has full column rank.

Formulae ($(X,\Pi) \mapsto R$) and ($(X,\Pi) \mapsto P$) give us a way to transform a given input/output representation to minimal kernel and minimal image representations.

42a ⟨$(X,\Pi) \mapsto R$ 42a⟩≡
```
function r = xio2r(x, io)
r = [x', -eye(size(x, 2))]; q = size(r, 2);
⟨default input/output partition 40b⟩, r = r(:, io);
```

42b ⟨$(X,\Pi) \mapsto P$ 42b⟩≡
```
function p = xio2p(x, io)
p = [eye(size(x, 1)); x']; q = size(p, 1);
⟨default input/output partition 40b⟩, p = p(io, :);
```

From image to kernel and from kernel to image representation

The relation

$$\ker(R) = \mathrm{image}(P) = \mathscr{B} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},0} \quad \Longrightarrow \quad RP = 0 \qquad (R \leftrightarrow P)$$

gives us a link between the parameters P and R. In particular, a minimal image representation $\mathrm{image}(P) = \mathscr{B}$ can be obtained from a given kernel representation $\ker(R) = \mathscr{B}$ by computing a basis for the null space of R. Conversely, a minimal kernel representation $\ker(R) = \mathscr{B}$ can be obtained from a given image representation $\mathrm{image}(P) = \mathscr{B}$ by computing a basis for the left null space of P.

42c ⟨$R \mapsto P$ 42c⟩≡
```
function p = r2p(r), p = null(r);
```

42d ⟨$P \mapsto R$ 42d⟩≡
```
function r = p2r(p), r = null(p')';
```

Converting an image or kernel representation to a minimal one

The kernel and image representations obtained from `xio2r`, `xio2p`, `p2r`, and `r2p` are minimal. In general, a given kernel or image representations is not minimal, *i.e.*, R has redundant rows and P has redundant columns. The kernel representation, defined by R, is minimal if and only if R has full row rank. Similarly, the image representation, defined by P, is minimal if and only if P has full column rank.

The problems of converting kernel and image representations to minimal ones are equivalent to the problem of finding a full rank matrix that has the same kernel or image as a given matrix. A numerically reliable way to solve this problem is to use the singular value decomposition.

Consider a model $\mathscr{B} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},0}$ with parameters $R \in \mathbb{R}^{\mathtt{g}\times\mathtt{q}}$ and $P \in \mathbb{R}^{\mathtt{q}\times\mathtt{s}}$ of respectively kernel and image representations. Let $R = U\Sigma V^\top$ be the singular value decomposition of R and let $\mathtt{p}$ be the rank of R. With the partitioning,

$$V =: \begin{bmatrix} V_1 & V_2 \end{bmatrix}, \quad \text{where } V_1 \in \mathbb{R}^{\mathtt{q}\times\mathtt{p}},$$

we have that

$$\operatorname{image}(R^\top) = \operatorname{image}(V_1).$$

Therefore,

$$\ker(R) = \ker(V_1^\top) \quad \text{and } V_1 \text{ has full rank},$$

so that $V_1^\top$ is a parameter of a minimal kernel representation of $\mathscr{B}$.

Let $P = U\Sigma V^\top$ be the singular value decomposition of P. With the partitioning,

$$U =: \begin{bmatrix} U_1 & U_2 \end{bmatrix}, \quad \text{where } U_1 \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}},$$

we have that

$$\operatorname{image}(P) = \operatorname{image}(U_1).$$

Since U_1 has full rank, it is a parameter of a minimal image representation of $\mathscr{B}$.

In the numerical implementation, the rank is replaced by the numerical rank with respect to a user defined tolerance `tol`.

43a ⟨$R \mapsto$ *minimal* R 43a⟩≡

```
function r = minr(r, tol)
[p, q] = size(r); ⟨default tolerance tol 41⟩
[u, s, v] = svd(r, 'econ'); pmin = sum(diag(s) > tol);
if pmin < p, r = v(:, 1:pmin)'; end
```

Defines:
`minr`, used in chunks 44a and 45b.

From kernel or image to input/output representation

The transformations from kernel to input/output and from image to input/output representations are closely related. They have as a sub-problem partitioning of the variables into inputs and outputs. Assume first that the input/output partition is given. This amounts to knowing the permutation matrix $\Pi \in \mathbb{R}^{\mathtt{q}\times\mathtt{q}}$ in (I/O$_0$).

43b ⟨$(R,\Pi) \mapsto X$ 43b⟩≡ 44a▷

```
function x = rio2x(r, io, tol)
q = size(r, 2); ⟨default input/output partition 40b⟩, ⟨default tolerance tol 41⟩
```

Defines:
`rio2x`, used in chunks 45b and 148b.

Consider a given parameter $R \in \mathbb{R}^{\mathtt{p}\times\mathtt{q}}$ of a *minimal* kernel representation and define

$$R\Pi =: \begin{bmatrix} R_u & R_y \end{bmatrix}, \quad \text{where } R_y \in \mathbb{R}^{\mathtt{p}\times\mathtt{p}}.$$

44a ⟨$(R,\Pi) \mapsto X$ 43b⟩+≡ ◁43b 44b▷

```
r = minr(r, tol); p = size(r, 1); m = q - p;
⟨inverse permutation 40c⟩, rpi = r(:, inv_io);
ru  = rpi(:, 1:m); ry = rpi(:, (m + 1):end);
```

Uses `minr` 43a.

Similarly, for a parameter $P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}}$ of *minimal* image representation, define

$$\Pi^\top P =: \begin{bmatrix} P_u \\ P_y \end{bmatrix}, \quad \text{where } P_u \in \mathbb{R}^{\mathtt{m}\times\mathtt{m}}.$$

If R_y and P_u are non-singular, it follows from ($(X,\Pi) \mapsto R$) and ($(X,\Pi) \mapsto P$) that

$$X = -(R_y^{-1}R_u)^\top \qquad ((R,\Pi) \mapsto X)$$

and

$$X = (P_y P_u^{-1})^\top \qquad ((P,\Pi) \mapsto X)$$

is the parameter of the input/output representation $\mathscr{B}_{\mathrm{i/o}}(X,\Pi)$, *i.e.*,

$$\ker\Big(\underbrace{\begin{bmatrix} \overset{\mathtt{m}}{R_u} & \overset{\mathtt{p}}{R_y} \end{bmatrix}\Pi^\top}_{R}\Big) = \mathscr{B}_{\mathrm{i/o}}\big(-(R_y^{-1}R_u)^\top, \Pi\big)$$

and

$$\operatorname{image}\left(\underbrace{\Pi\begin{bmatrix} P_u \\ P_y \end{bmatrix}}_{P}\begin{matrix} \mathtt{m} \\ \mathtt{p} \end{matrix}\right) = \mathscr{B}_{\mathrm{i/o}}\big((P_y P_u^{-1})^\top, \Pi\big).$$

44b ⟨$(R,\Pi) \mapsto X$ 43b⟩+≡ ◁44a

```
[u, s, v] = svd(ry); s = diag(s);
if s(end) < tol
    warning('Computation of X is ill conditioned.'); x = NaN;
else
    x = -( v * diag(1 ./ s) * u' * ru )';
end
```

Singularity of the blocks R_y and P_u implies that the input/output representation with a permutation matrix Π is not possible. In such cases, the function `rio2x` issues a warning message and returns `NaN` value for X.

The function `r2io` uses `rio2x` in order to find all possible input/output partitions for a model specified by a kernel representation.

44c ⟨$R \mapsto \Pi$ 44c⟩≡ 45a▷

```
function IO = r2io(r, tol)
q = size(r, 2); ⟨default tolerance tol 41⟩
```

The search is exhaustive over all input/output partitionings of the variables (*i.e.*, all choices of `m` elements of the set of variable indexes $\{1,\ldots,q\}$), so that the computation is feasible only for a small number of variables (say, less than 6).

45a ⟨$R \mapsto \Pi$ 44c⟩+≡ ◁44c 45b▷

```
IO = perms(1:q); nio = size(IO, 1);
```

The parameter X for each candidate partition is computed. If the computation of X is ill-conditioned, the corresponding partition is discarded.

45b ⟨$R \mapsto \Pi$ 44c⟩+≡ ◁45a

```
not_possible = []; warning_state = warning('off');
r = minr(r, tol);
for i = 1:nio
    x = rio2x(r, IO(i, :), tol);
    if isnan(x), not_possible = [not_possible, i]; end
end
warning(warning_state); IO(not_possible, :) = [];
```

Uses `minr` 43a and `rio2x` 43b.

Summary of transitions among representations

Figure 2.1 summarizes the links among the parameters R, P, and (X,Π) of a linear static model $\mathscr{B}$ and the functions that implement the transitions.

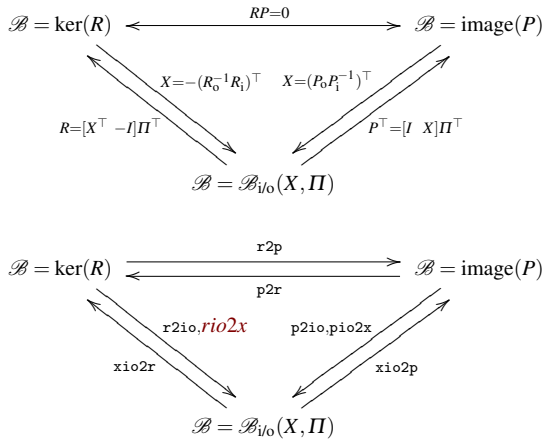


Fig. 2.1: Going from one model representation to another means converting the parameters of a given representation into the parameters of another representation of the same model. The upper diagram summarizes the formulas for the transitions among the kernel, image, and input/output representations. The lower diagram shows the corresponding MATLAB functions that implement these transitions.

2.1.3 Linear static model complexity

A linear static model is a finite dimensional subspace. The dimension `m` of the subspace is equal to the number of inputs and is invariant of the model representation. The integer constant `m` quantifies the *model complexity*: the model is more complex when it has more inputs. The rationale for this definition of model complexity is that inputs are "unexplained" variables by the model, so the more inputs the model has, the less it "explains" the modeled phenomenon. In data modeling the aim is to obtain low-complexity models, a principle generally refered to as *Occam's razor*.

Note 2.2 (Rank estimation problems in computation of model complexity). Finding the model's complexity from given exact data, nonminimal kernel, or nonminimal image representation is a rank estimation problem.

2.2 Dynamic model representations

The concepts of model, representation, and complexity, presented in Section 2.1 for linear static systems are generalized next to linear time-invariant systems. The linear time-invariant model class is a proper generalization of the linear static model class. There is a richer variety of possible representations and they involve differential, in continuous-time, and difference, in discrete-time, equations. This leads to more transitions among representations and increased complexity in comparison with Section 2.1, where linear algebraic equations were used.

First, we define formally a dynamical model as a set of functions—the trajectories of the model. Properties of the model, such as linearity and time-invariance are also defined on the level of the model trajectories. Then, a plethora of possible representations—kernel, input/output, state space, image, convolution, and transfer function—are introduced. The links among them are summarized in Figure 2.3. The transition, called realization problem, from impulse response to a state space representation is described in details because of its importance for subspace system identification. Finally, the notion of model complexity that we have introduced for the static linear model class is generalized to the linear time-invariant model class.

2.2.1 Kernel, input/output, state space, image, convolution, and transfer function representations

An observation d_j of a static model is a vector of variables. In the dynamic case, the observations depend on time, so that apart from the multivariable aspect, there is also a time evaluation aspect. In the dynamic case, an observation is refered to as a *trajectory*, *i.e.*, it is a vector valued *function* of a scalar argument. The time variable

takes its values in the set of integers $\mathbb{Z}$ (*discrete-time* model) or in the set of real numbers $\mathbb{R}$ (*continuous-time* model). We denote the time axis by the symbol $\mathscr{T}$.

A dynamic model $\mathscr{B}$ with $\mathtt{q}$ variables is a subset of the trajectory space $(\mathbb{R}^{\mathtt{q}})^{\mathscr{T}}$ — the set of all functions from the time axis $\mathscr{T}$ to the variable space $\mathbb{R}^{\mathtt{q}}$.

Next, we consider the class of finite dimensional linear time-invariant dynamical models. By definition, a model $\mathscr{B}$ is *linear* if it is a subspace of the data space $(\mathbb{R}^{\mathtt{q}})^{\mathscr{T}}$. In order to define the *time-invariance* property, we introduce the *shift operator* σ^τ. Acting on a signal w, σ^τ produces a signal $\sigma^\tau w$, which is the backwards shifted version of w by τ time units, *i.e.*,

$$(\sigma^\tau w)(t) := w(t+\tau), \quad \text{for all } t \in \mathscr{T}.$$

Acting on a set of trajectories, σ^τ shifts all trajectories in the set, *i.e.*,

$$\sigma^\tau \mathscr{B} := \{\, \sigma^\tau w \mid w \in \mathscr{B} \,\}.$$

A model $\mathscr{B}$ is shift-invariant if it is invariant under any shift in time, *i.e.*,

$$\sigma^\tau \mathscr{B} = \mathscr{B}, \quad \text{for all } \tau \in \mathscr{T}.$$

The model $\mathscr{B}$ is *finite dimensional* if it is a closed subset (in the topology of pointwise convergence). Finite dimensionality is equivalent to the property that at any time t the future behavior of the model is deterministically independent of the past behavior, given a finite dimensional vector, called a *state* of the model. Intuitively, the state is the information (or memory) of the past that is needed in order to predict the future. The smallest state dimension is an invariant of the system, called the *order*. We denote the set of finite dimensional linear time-invariant models with $\mathtt{q}$ variables and order at most $\mathtt{n}$ by $\mathscr{L}^{\mathtt{q},\mathtt{n}}$ and the *order* of $\mathscr{B}$ by $\mathtt{n}(\mathscr{B})$.

A finite dimensional linear time-invariant model $\mathscr{B} \in \mathscr{L}^{\mathtt{q},\mathtt{n}}$ admits a representation by a difference or differential equation

$$\begin{aligned} R_0 w + R_1 \lambda w + \cdots + R_\ell \lambda^\ell w &= (\underbrace{R_0 + R_1 \lambda + \cdots + R_\ell \lambda^\ell}_{R(\lambda)}) w \\ &= R(\lambda) w = 0, \end{aligned} \tag{DE}$$

where λ is the unit shift operator σ in the discrete-time case and the differential operator $\frac{\mathrm{d}}{\mathrm{d}t}$ in the continuous-time case. Therefore, the model $\mathscr{B}$ is the kernel

$$\mathscr{B} := \ker\big(R(\lambda)\big) = \{\, w \mid \text{(DE) holds} \,\}, \tag{KER}$$

of the operator $R(\lambda)$. The smallest degree ℓ of a polynomial matrix

$$R(z) := R_0 + R_1 z + \cdots + R_\ell z^\ell \in \mathbb{R}^{\mathtt{g}\times\mathtt{q}}[z],$$

that defines a kernel representation (KER) of the system $\mathscr{B}$ is invariant of the representation and is called the *lag* $\ell(\mathscr{B})$ of the system $\mathscr{B}$.

The order of the system is the total degree (sum of the row degrees) of the polynomial matrix R in a kernel representation of the system. Therefore, we have the following link between the order and the lag of a linear time-invariant model:

$$\mathtt{n}(\mathscr{B}) \leq \mathtt{p}(\mathscr{B})\ell(\mathscr{B}).$$

As in the static case, the smallest possible number of rows $\mathtt{g}$ of the polynomial matrix R in a kernel representation (KER) of a finite dimensional linear time-invariant system $\mathscr{B}$ is the number of outputs $\mathtt{p}(\mathscr{B})$ of $\mathscr{B}$. Finding an input/output partitioning for a model specified by a kernel representation amounts to selection of a nonsingular $\mathtt{p}\times\mathtt{p}$ submatrix of R. The resulting input/output representation is:

$$\mathscr{B} = \mathscr{B}_{\mathrm{i/o}}(P,Q,\Pi) := \{\, w = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid Q(\sigma)u = P(\sigma)y \,\} \tag{I/O}$$

with parameters the polynomial matrices $Q \in \mathbb{R}^{\mathtt{p}\times\mathtt{m}}[z]$ and $P \in \mathbb{R}^{\mathtt{p}\times\mathtt{p}}[z]$, such that $\det(P) \neq 0$, and the permutation matrix Π.

In general, the representation (I/O) involves higher order shifts or derivatives. A first order representation

$$\begin{aligned} \mathscr{B} = \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D,\Pi) := \{\, w = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid \text{there is } x, \text{ such that} \\ \lambda x = Ax + Bu \text{ and } y = Cx + Du \,\}, \end{aligned} \tag{I/S/O}$$

with an auxiliary variable x, however, is always possible. The representation (I/S/O) displays not only the input/output structure of the model but also its state structure and is refered to as an input/state/output representation of the model. The parameters of an input/state/output representation are the matrices $A \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$, $B \in \mathbb{R}^{\mathtt{n}\times\mathtt{m}}$, $C \in \mathbb{R}^{\mathtt{p}\times\mathtt{n}}$, $D \in \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$, and a permutation matrix Π. The parameters are non unique due to

- nonuniqueness in the choice of the input/output partition,
- existence of redundant states (nonminimality of the representation), and
- change of state space basis

$$\begin{aligned} \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D) = \mathscr{B}_{\mathrm{i/s/o}}(T^{-1}AT, T^{-1}B, CT, D), \\ \text{for any nonsingular matrix } T \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}. \end{aligned} \tag{CB}$$

An input/state/output representation $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$ is called *minimal* when the state dimension $\mathtt{n}$ is as small as possible. The minimal state dimension is an invariant of the system and is equal to the order $\mathtt{n}(\mathscr{B})$ of the system.

A system $\mathscr{B}$ is *autonomous* if the "past" $\big(\ldots, w(-2), w(-1)\big)$ of a trajectory $w \in \mathscr{B}$ completely determines the "future" $\big(w(0), w(1), \ldots\big)$. It can be shown that a system $\mathscr{B}$ is autonomous if and only if it has no inputs. An autonomous finite dimensional linear time-invariant system is parametrized by the pair of matrices A and C via the state space representation

$$\mathscr{B}(A,C) := \{\, w = y \mid \text{there is } x, \text{ such that } \sigma x = Ax \text{ and } y = Cx \,\}.$$

The dimension $\dim(\mathscr{B})$ of a linear autonomous $\mathscr{B}$ is equal to its order $\mathtt{n}(\mathscr{B})$.

In a certain sense the opposite of an autonomous model is a controllable system. The model $\mathscr{B}$ is *controllable* if for any $w_{\mathrm{p}}, w_{\mathrm{f}} \in \mathscr{B}$, there is $\tau > 0$ and $w \in \mathscr{B}$, such that $w(t) = w_{\mathrm{p}}(t)$, for all $t < 0$, and $w(t) = w_{\mathrm{f}}(t)$, for all $t \geq \tau$. In words, it is possible to steer the "past" trajectory w_{p} to the "future" trajectory w_{f} over a "control" interval $[0, \tau]$, see Figure 2.2. The subset of controllable systems of the set of linear time-invariant systems $\mathscr{L}^{\mathtt{q}}$ is denoted by $\mathscr{L}^{\mathtt{q}}_{\mathrm{ctrb}}$.

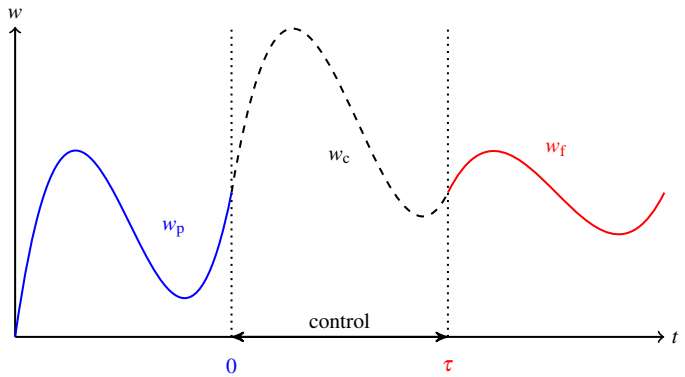


Fig. 2.2: A system is controllable if any "past" trajectory w_{p} can be concatenated with any "future" trajectory w_{f} via a suitable control w_{c} over an interval $[0, \tau]$.

A summary of properties of a dynamical system is given in Table 2.1.

Table 2.1: Summary of commonly used model properties.

property	definition
linearity	— $w, v \in \mathscr{B} \implies \alpha w + \beta v \in \mathscr{B}$, for all $\alpha, \beta \in \mathbb{R}$
time-invariance	— $\sigma^{\tau}\mathscr{B} = \mathscr{B}$, for all $\tau \in \mathscr{T}$
finite dimensionality	— $\mathscr{B}$ is a closed set; equivalently $\mathtt{n}(\mathscr{B}) < \infty$
autonomy	— the past of any trajectory completely determines its future; equivalently $\mathtt{m}(\mathscr{B}) = 0$
controllability	— the past of any trajectory can be concatenated to the future of any other trajectory by a third trajectory if a transition period is allowed

Apart from the kernel, input/output, and input/state/output representations, a controllable finite dimensional linear time-invariant model admits an image, convolution, and transfer function representations.

- *Image representation* with parameter the polynomial matrix $P(z) \in \mathbb{R}^{\mathtt{q}\times g}[z]$ is

$$\mathscr{B} = \operatorname{image}\big(P(\lambda)\big) := \{\, w \mid w = P(\lambda)\ell, \text{ for some } \ell \,\}. \tag{IMAGE}$$

- *Convolution representation* with parameters the signal $H : \mathscr{T} \to \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$ and a permutation matrix Π is

$$\mathscr{B} = \mathscr{B}_{\mathrm{i/o}}\big(H, \Pi\big) := \{\, w = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid y = H \star u \,\}, \tag{CONV}$$

  where $\star$ is the convolution operator

$$y(t) = (H \star u)(t) := \sum_{\tau=0}^{\infty} H(\tau)u(t-\tau), \quad \text{in discrete-time, or}$$
$$y(t) = (H \star u)(t) := \int_0^{\infty} H(\tau)u(t-\tau)\mathrm{d}\tau, \quad \text{in continuous-time.}$$

- *Transfer function representation* with parameters the rational matrix $H \in \mathbb{R}^{\mathtt{p}\times\mathtt{m}}(z)$ and a permutation matrix Π is

$$\mathscr{B} = \mathscr{B}_{\mathrm{i/o}}\big(H, \Pi\big) := \{\, w = \Pi \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid \mathscr{F}(y) = H(z)\mathscr{F}(u) \,\}, \tag{TF}$$

  where $\mathscr{F}$ is the Z-transform in discrete-time and the Laplace transform in continuous-time.

Transitions among the parameters H, $H(z)$, and (A,B,C,D) are classical problems, see Figure 2.3. Next, we review the transition from impulse response H to parameters (A,B,C,D) of an input/state/output representation, which plays an important role in deriving methods for Hankel structured low-rank approximation.

2.2.2 The realization problem

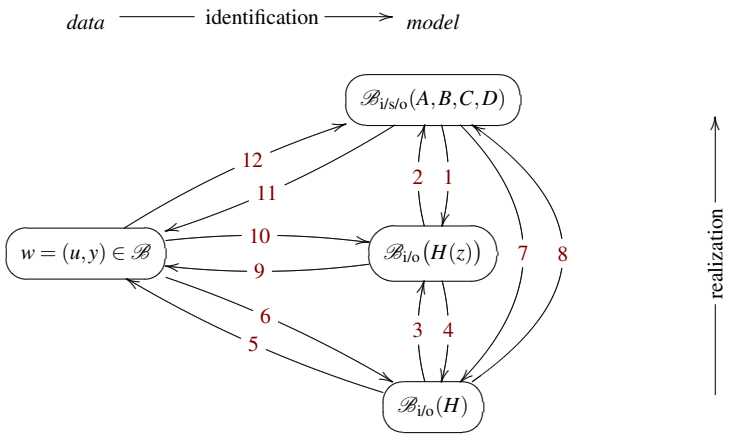
The problem of passing from a convolution representation to an input/state/output representation is called impulse response realization.

Definition 2.3 (Impulse response realization). A linear time-invariant system $\mathscr{B}$ with $\mathtt{m}$ inputs and $\mathtt{p}$ outputs and an input/output partition, specified by a permutation matrix Π, is a *realization* of (or realizes) an impulse response $H : \mathscr{T} \to \mathbb{R}^{\mathtt{p}\times\mathtt{m}}$ if $\mathscr{B}$ has a convolution representation $\mathscr{B} = \mathscr{B}_{\mathrm{i/o}}(H, \Pi)$. A realization $\mathscr{B}$ of H is *minimal* if its order $\mathtt{n}(\mathscr{B})$ is the smallest over all realizations of H.

In what follows, we fix the input/output partition Π to the default one $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$.

The impulse response realization problem is a special system identification problem. To see this, define

$$H := \begin{bmatrix} h_1 & \cdots & h_{\mathtt{m}} \end{bmatrix} \quad I_{\mathtt{m}} := \begin{bmatrix} e_1 & \cdots & e_{\mathtt{m}} \end{bmatrix},$$



1. $H(z)=C(Iz-A)^{-1}B+D$
2. realization of a transfer function
3. Z or Laplace transform of $H(t)$
4. inverse transform of $H(z)$
5. convolution $y_{\mathrm{d}}=H\star u_{\mathrm{d}}$
6. exact identification
7. $H(0)=D$, $H(t)=CA^{t-1}B$ (discrete-time), $H(t)=Ce^{At}B$ (continuous-time), for $t>0$
8. realization of an impulse response
9. simulation with input u_{d} and $x(0)=0$
10. exact identification
11. simulation with input u_{d} and $x(0)=x_{\mathrm{ini}}$
12. exact identification

Fig. 2.3: Similar to the static case, shown in Figure 2.1, the parameters of one representation of a linear time-invariant dynamic system can be converted to parameters of a different representation of the same system. In addition to the input/state/output, transfer function, and impulse response representations, exact trajectory w of the system, may completely specify the system. The transitions upwards from impulse response and transfer function to state space are known as the realization problem. The transitions from data to a representation are system identification problems.

let δ be the delta function and let $\wedge$ be the concatenation map (at time 0)

$$w=w_{\mathrm{p}}\wedge w_{\mathrm{f}},\quad w(t):=\begin{cases}w_{\mathrm{p}}(t), & \text{if } t<0\\ w_{\mathrm{f}}(t), & \text{if } t\geq 0.\end{cases}$$

The system $\mathscr{B}$ realizes H if and only if

$$w_i:=\begin{bmatrix}e_i\delta\\ 0\wedge h_i\end{bmatrix}\in\mathscr{B},\quad \text{for } i=1,\ldots,\mathtt{m},$$

i.e., $\mathscr{B}$ is an exact model for the signals $w_1,\ldots,w_N$. Finding $\mathscr{B}$ from $w_1,\ldots,w_N$ is an exact system identification problem. Note that the data consists of multiple

experiments, however, these experiments are carried out under special conditions: pulse inputs along one of the input channels of the system and zero initial conditions.

Note 2.4 (Discrete-time vs continuous-time system realization). There are some differences between the discrete and continuous-time realization theory. Next, we consider the discrete-time case. It turns out, however, that the discrete-time algorithms can be used for realization of continuous-time systems by applying them on the sequence of the Markov parameter $\big(H(0),\frac{\mathrm{d}}{\mathrm{d}t}H(0),\ldots\big)$ of the system.

The sequence

$$H=\big(H(0),H(1),H(2),\ldots,H(t),\ldots\big),\quad \text{where } H(t)\in\mathbb{R}^{\mathtt{p}\times\mathtt{m}}$$

is a one sided infinite matrix-values time series. Acting on H, the shift operator σ, removes the first sample, *i.e.*,

$$\sigma H=\big(H(1),H(2),\ldots,H(t),\ldots\big).$$

A sequence H might not be realizable by a *finite dimensional* linear time-invariant system, but if it is realizable, a minimal realization is unique.

Theorem 2.5 (Test for realizability). *The sequence $H:\mathbb{Z}_+\to\mathbb{R}^{\mathtt{p}\times\mathtt{m}}$ is realizable by a finite dimensional linear time-invariant system if and only if the (infinite) Hankel matrix $\mathscr{H}(\sigma H)$ has a finite rank. Moreover, the order of a minimal realization is equal to* $\operatorname{rank}\big(\mathscr{H}(\sigma H)\big)$ *and there is a unique system $\mathscr{B}$ in $\mathscr{L}_{\mathtt{m}}^{\mathtt{q},\mathtt{n}}$ that realizes H.*

Proof. ($\Longrightarrow$) Let H be realizable by a system $\mathscr{B}\in\mathscr{L}_{\mathtt{m}}^{\mathtt{q},\mathtt{n}}$ with a minimal input/state/output representation $\mathscr{B}=\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$. Then

$$H(0)=D\quad\text{and}\quad H(t)=CA^{t-1}B,\quad \text{for } t>0.$$

The (i,j)th block element of the Hankel matrix $\mathscr{H}(\sigma H)$ is

$$H(i+j-1)=CA^{i+j-2}B=CA^{i-1}A^{j-1}B.$$

Let

$$\mathscr{O}_t(A,C):=\operatorname{col}(C,CA,\ldots,CA^{t-1}) \qquad (\mathscr{O})$$

be the extended observability matrix of the pair (A,C) and

$$\mathscr{C}_t(A,B):=\begin{bmatrix}B & AB & \cdots & A^{t-1}B\end{bmatrix} \qquad (\mathscr{C})$$

be the extended controllability matrix of the pair (A,B). With $\mathscr{O}(A,C)$ and $\mathscr{C}(A,B)$ being the infinite observability and controllability matrices, we have

$$\mathscr{H}(\sigma H)=\mathscr{O}(A,C)\mathscr{C}(A,B) \qquad (\mathscr{O}\mathscr{C})$$

Since the representation $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$ is assumed to be minimal, $\mathscr{C}(A,B)$ has full row rank and $\mathscr{O}(A,C)$ has full column rank. Therefore, ($\mathscr{O}\mathscr{C}$) is a rank revealing factorization of $\mathscr{H}(\sigma H)$ and

$$\operatorname{rank}\big(\mathscr{H}(\sigma H)\big) = \mathtt{n}(\mathscr{B}).$$

($\Longleftarrow$) In this direction, the proof is constructive and results in an *algorithm* for computation of the minimal realization of H in $\mathscr{L}_{\mathtt{m}}^{\mathtt{q},\mathtt{n}}$, where $\mathtt{n} = \mathit{rank}\big(\mathscr{H}(\sigma H)\big)$. A realization algorithm is presented in Section 3.1.

Theorem 2.5 shows that

$$\operatorname{rank}\big(\mathscr{H}_{i,j}(\sigma H)\big) = \mathtt{n}(\mathscr{B}), \quad \text{for all } i \text{ and } j, \text{ such that } \mathtt{p}i \geq \mathtt{n}(\mathscr{B}) \text{ and } \mathtt{m}j \geq \mathtt{n}(\mathscr{B}).$$

This suggests a method to find the order $\mathtt{n}(\mathscr{B})$ of the minimal realization of H: check the rank deficiency of $\mathscr{H}_{i,j}(\sigma H)$ for growing values of i and j. Alternatively, if an an upper bound $\mathtt{n}_{\max}$ of the order is a priori known, compute the rank of the *finite* Hankel matrix $\mathscr{H}_{i,j}(\sigma H)$ ensuring that both dimensions are bigger than $\mathtt{n}_{\max}$ ($i \geq \lceil \mathtt{n}_{\max}/\mathtt{p} \rceil$ and $j \geq \lceil \mathtt{n}_{\max}/\mathtt{m} \rceil$). Algorithms for computing the order and parameters of the minimal realization are presented in Section 3.1.

2.2.3 Linear time-invariant model complexity

Associate with a linear time-invariant dynamical system $\mathscr{B}$, we have defined the following system invariants:

$\mathtt{m}(\mathscr{B})$	— number of inputs,	$\mathtt{n}(\mathscr{B})$	— order,
$\mathtt{p}(\mathscr{B})$	— number of outputs,	$\ell(\mathscr{B})$	— lag.

The complexity of a linear static model $\mathscr{B}$ is the number of inputs $\mathtt{m}(\mathscr{B})$ of $\mathscr{B}$ or, equivalently, the dimension $\dim(\mathscr{B})$ of $\mathscr{B}$. Except for the class of autonomous systems, however, the dimension of a dynamical model is infinite. We define the restriction $\mathscr{B}|_T$ of $\mathscr{B}$ to the interval $[1,T]$,

$$\mathscr{B}|_T := \{\, w \in \mathbb{R}^T \mid \text{there exist } w_{\mathrm{p}} \text{ and } w_{\mathrm{f}}, \text{ such that } (w_{\mathrm{p}}, w, w_{\mathrm{f}}) \in \mathscr{B} \,\}. \qquad (\mathscr{B}|_T)$$

For a linear time-invariant model $\mathscr{B}$ and for $T > \mathtt{n}(\mathscr{B})$,

$$\dim(\mathscr{B}|_T) = \mathtt{m}(\mathscr{B})T + \mathtt{n}(\mathscr{B}) \leq \mathtt{m}(\mathscr{B})T + \ell(\mathscr{B})\mathtt{p}(\mathscr{B}), \qquad (\dim \mathscr{B})$$

which shows that the pairs of natural numbers $\big(\mathtt{m}(\mathscr{B}), \mathtt{n}(\mathscr{B})\big)$ and $\big(\mathtt{m}(\mathscr{B}), \ell(\mathscr{B})\big)$ characterize the model's complexity. The elements of the model class $\mathscr{L}_{\mathtt{m},\ell}^{\mathtt{q}}$ are linear time-invariant systems of complexity bounded by the pair $(\mathtt{m}, \ell)$ and, similarly, the elements of the model class $\mathscr{L}_{\mathtt{m}}^{\mathtt{q},\mathtt{n}}$ are linear time-invariant systems of complexity bounded by the pair $(\mathtt{m}, \mathtt{n})$. A static model is a special case of a dynamic model when the lag (or the order) is zero. Note that the linear static model class $\mathscr{L}_{\mathtt{m},0}$ corresponds to the linear time-invariant model class $\mathscr{L}_{\mathtt{m},\ell}$ with $\ell = 0$.

In the autonomous case, *i.e.*, with $\mathtt{m}(\mathscr{B}) = 0$, $\mathscr{B}$ is finite dimensional, $\dim(\mathscr{B}) = \mathtt{n}$. The dimension of the system corresponds to the number of degrees of freedom in selecting a trajectory $w \in \mathscr{B}$. In the case of an autonomous system, the trajectory

depends only on the initial condition (an $\mathtt{n}(\mathscr{B})$ dimensional vector). In the presence of inputs, the number of degrees of freedom is increased on each time step by $\mathtt{m}$— the number of inputs. Asymptotically as $T \to \infty$, the term $\mathtt{m}T$ in ($\dim \mathscr{B}$) dominates the term $\mathtt{n}$. Therefore, in comparing the complexity of linear time-invariant systems, by convention, a system with more inputs is more complex than a system with less inputs, irrespective of their state dimensions.

2.3 Stochastic model representation

A deterministic dynamical system is a collection of trajectories. In analogy, a stochastic dynamical system is a collection of stochastic processes. This section reviews, first, the basic notions of correlation function, spectral density, and white process. Then, the class of stochastic systems, called ARMA system, is defined as a deterministic linear time-invariant system with an input that is a white noise process. An ARMA system is the stochastic equivalent of a deterministic autonomous system. The stochastic equivalent of an input/output system is an ARMAX system, which has a deterministic input in addition to the stochastic input. The main result of the section is a theorem that gives five representations of an ARMA system.

Fig. 2.4: An ARMA system (left) is a deterministic linear time-invariant system with white stochastic process e as an input. An ARMAX system (right) has, in addition, a deterministic input u. The channel $e \mapsto y$ is the "stochastic part" and $e \mapsto y$ is the "deterministic part" of the ARMAX system.

Stochastic processes and correlation functions

We consider real valued, jointly Gaussian, zero mean, stationary, ergodic processes. Let w be a stochastic process on $\mathbb{Z}$ with $\mathtt{q}$-variables. The *correlation function* of w

$$\operatorname{corr}(w) = R_{ww} := \big(\ldots, R_{ww}(0), \ldots, R_{ww}(t), \ldots\big)$$

is defined as

$$R_{ww}(t) := \mathbf{E}\left(w(t)w^{\top}(0)\right) \in \mathbb{R}^{\mathtt{q}\times\mathtt{q}},$$

where $\mathbf{E}$ is the expectation operator.

An observed realization of w is denoted by w_{d} ("d" stands for "data"). Due to the ergodicity assumption, R_{ww} can be expressed in terms of an infinite realization w_{d} as

$$R_{ww}(t) = \lim_{T\to\infty} \frac{1}{T} \sum_{\tau=1}^{T} w_{\mathrm{d}}(t+\tau) w_{\mathrm{d}}^{\top}(\tau).$$

If the realization w_{d} is over a finite interval

$$w_{\mathrm{d}} = \big(w_{\mathrm{d}}(1), \ldots, w_{\mathrm{d}}(T)\big),$$

only a finite sample estimate can be computed, *e.g.*, the *standard biased estimator*

$$\widehat{R}_{ww}(t) := \frac{1}{T} \sum_{\tau=1}^{T-t} w_{\mathrm{d}}(t+\tau) w_{\mathrm{d}}^{\top}(\tau) \quad \text{and} \quad \widehat{R}_{ww}(-t) = \widehat{R}_{ww}^{\top}(t),$$
$$\text{for } t = 0, \ldots, T-1. \quad (\widehat{R}_{ww})$$

Alternative methods for computing $\widehat{R}_{ww}$ from w_{d} are described in (Stoica and Moses, 2005, Chapter 2) in the context of the nonparameteric spectral estimation.

Due to the symmetry property $R_{ww}(t) = R_{ww}^{\top}(-t)$ of the correlation function, it is sufficient to specify a correlation function $R_{ww} \in (\mathbb{R}^{\mathtt{q}\times\mathtt{q}})^{\mathbb{Z}}$ on the half line $\mathbb{Z}_+$. The covariance matrix of the vector $\operatorname{col}\big(w(1), \ldots, w(t)\big)$ has Toeplitz structure

$$\begin{bmatrix} R_{ww}(0) & R_{ww}(-1) & R_{ww}(-2) & \cdots & R_{ww}(-t+1) \\ R_{ww}(1) & R_{ww}(0) & R_{ww}(-1) & \ddots & \vdots \\ R_{ww}(2) & R_{ww}(1) & R_{ww}(0) & \ddots & R_{ww}(-2) \\ \vdots & \ddots & \ddots & \ddots & R_{ww}(-1) \\ R_{ww}(t-1) & \cdots & R_{ww}(2) & R_{ww}(1) & R_{ww}(0) \end{bmatrix}.$$

The *spectral density* of w is defined as the Z-transform of the correlation function

$$S_{ww} := \mathscr{Z}(R_{ww}).$$

Independence of two random vectors or processes is denoted by $\perp$. A process ε is called *white* if $\varepsilon(t_1) \perp \varepsilon(t_2)$ for all $t_1 \neq t_2$, and *normalized* if $\mathbf{E}\big(\varepsilon(0)\varepsilon^{\top}(0)\big) = I$.

Auto Regressive Moving Average (ARMA) systems

The set $\mathscr{B} \subseteq (\mathbb{R}^{\mathtt{p}})^{\mathbb{Z}}$ is a behavior of an ARMA system if

$$\mathscr{B} = \{\, y \in (\mathbb{R}^{\mathtt{p}})^{\mathbb{Z}} \mid \text{there is } e,\ \operatorname{corr}(e) = \delta I_{\mathtt{e}},$$
$$\text{such that } A(\sigma)w = M(\sigma)e \,\}. \quad \text{(ARMA)}$$

We refer to the behavior $\mathscr{B}$ of an ARMA system as the system itself. The polynomial matrices $A \in \mathbb{R}^{\mathtt{g}\times\mathtt{p}}[z]$ and $M \in \mathbb{R}^{\mathtt{g}\times\mathtt{e}}[z]$ in (ARMA) are the parameters of what is called *ARMA representation* of the system $\mathscr{B}$. Note that, contrary to the deterministic case, we define a stochastic system via a particular representation. A behavioral

characterization of static stochastic systems is given in (Willems, 2013). A behavioral characterization of a dynamic stochastic system is still an open problem.

In addition to polynomials (A, M), an ARMA system can be represented via a correlation function, spectral density, and state space representation.

Theorem 2.6 (Representations of ARMA systems). *The following are equivalent:*

1. ***(ARMA representation)*** *$\mathscr{B}$ is an ARMA system,* i.e., *there are $A \in \mathbb{R}^{\mathtt{g}\times\mathtt{p}}[z]$ and $M \in \mathbb{R}^{\mathtt{g}\times\mathtt{e}}[z]$, such that (ARMA) holds,*
2. ***(correlation function representation)*** *there is a sequence $R_{yy} \in (\mathbb{R}^{\mathtt{p}\times\mathtt{p}})^{\mathbb{Z}}$, the correlation function of $\mathscr{B}$, such that* $\operatorname{rank}\big(\mathscr{H}(R_{yy})\big) < \infty$ *and*

$$\mathscr{B} = \{\, y \in (\mathbb{R}^{\mathtt{p}})^{\mathbb{Z}} \mid \mathit{corr}(y) = R_{yy} \,\};$$

3. ***(spectral density representation)*** *there is a sequence $S \in (\mathbb{C}^{\mathtt{p}\times\mathtt{p}})^{\mathbb{Z}}$, the spectral density function of $\mathscr{B}$, that is rational and*

$$\mathscr{B} = \{\, y \in (\mathbb{R}^{\mathtt{p}})^{\mathbb{Z}} \mid \mathscr{Z}\big(\mathit{corr}(y)\big) = S \,\};$$

4. ***(state space representation)*** *there are $A \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$, $B \in \mathbb{R}^{\mathtt{n}\times\mathtt{e}}$, $C \in \mathbb{R}^{\mathtt{p}\times\mathtt{n}}$, and $D \in \mathbb{R}^{\mathtt{p}\times\mathtt{e}}$, such that*

$$\mathscr{B} = \{\, y \in (\mathbb{R}^{\mathtt{p}})^{\mathbb{Z}} \mid \text{there are } e \text{ and } x,\ \text{with } \mathit{corr}(e) = \delta I_{\mathtt{e}},$$
$$\text{such that } \sigma x = Ax + Be,\ y = Cx + De \,\};$$

5. ***(innovation representation)*** *there are $A \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$, $B \in \mathbb{R}^{\mathtt{n}\times\mathtt{p}}$, $K \in \mathbb{R}^{\mathtt{p}\times\mathtt{n}}$, and $V_e \in \mathbb{R}^{\mathtt{p}\times\mathtt{p}}$, such that*

$$\mathscr{B} = \{\, y \in (\mathbb{R}^{\mathtt{p}})^{\mathbb{Z}} \mid \text{there are } e \text{ and } x,\ \text{with } \mathit{corr}(e) = \delta V_e,$$
$$\text{such that } \sigma x = Ax + Be,\ y = Kx + e \,\}.$$

As in the case of deterministic linear time-invariant systems, the transitions from one representation of the stochastic system to another representation of the same system are the pillars of the stochastic system theory. Next, we outline transitions among ARMA representations and mention the corresponding computational problems.

The transition from a state space representation to an innovation representation is the *Kalman filter* problem. The Kalman filter problem involves solution of a Riccati difference equation, see (Anderson and Moore, 1979; Kailath et al, 2000).

The transition from a state space representation to the correlation function

$$R_{yy}(0) = CV_xC^{\top} + DD^{\top}, \quad R_{yy}(t) = CA^{t-1}\big(AV_xC^{\top} + BD^{\top}\big), \quad \text{for all } t > 0,$$
$$((A, B, C, D) \mapsto R_{yy})$$

requires solution of the Lyapunov equation for the state covariance

$$V_x := \mathbf{E}\big(x(0)x^{\top}(0)\big), \quad V_x = AV_xA^{\top} + BB^{\top}.$$

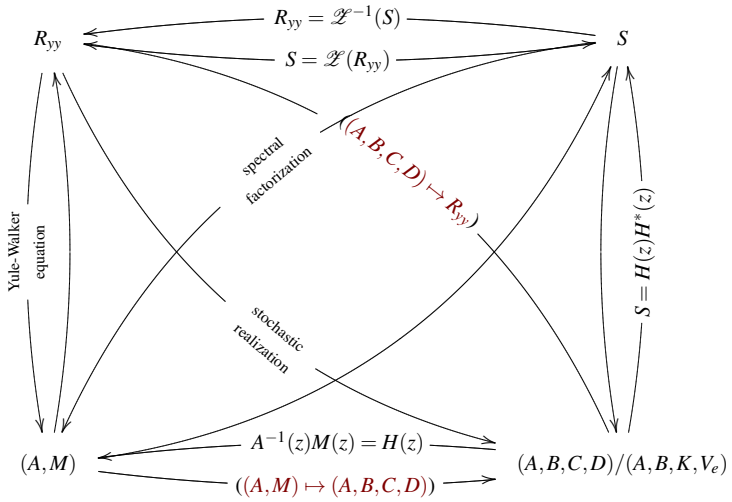


Fig. 2.5: The computational problems related to the transitions from one representation of an ARMA system to another one are the pillars of the stochastic system's theory. Among these pillars are the stochastic realization, spectral factorization, Yule-Walker equation, and Kalman filter.

The inverse transition $R_{yy} \mapsto (A,B,C,D)$ is the *stochastic realization* problem (Picci, 1991). Similarly to the deterministic realization problem (see Sections 2.2.2 and 3.1), it can be solved by rank revealing factorization of a Hankel matrix and solution of a system of linear equations, see Algorithm 1.

Algorithm 1 Stochastic realization

Input: Correlation function R_{yy}, such that $\mathscr{H}(R_{yy})$ has finite rank.

1: Factor $\mathscr{H}(\sigma R_{yy}) = \mathscr{O}\mathscr{C}$ into full column rank $\mathscr{O}$ and full row rank $\mathscr{C}$.

2: Let G be the first block entry of $\mathscr{C}$, C the first block entry of $\mathscr{O}$, and $A := (\sigma^{-1}\mathscr{O})^{+}(\sigma\mathscr{O})$, where $\sigma^{-1}\mathscr{O}$ removes the first block-element and $\sigma\mathscr{O}$ removes the last block-element of $\mathscr{O}$.

3: Solve for V_x,

$$V_x = AV_xA^\top + (G - AV_xC^\top)\big(R_{yy}(0) - CV_xC^\top\big)(G - AV_xC^\top)^\top.$$

4: Define B and D so that

$$\begin{bmatrix} V_x & G \\ G^\top & R_{yy}(0) \end{bmatrix} - \begin{bmatrix} A \\ C \end{bmatrix} V_x \begin{bmatrix} A^\top & C^\top \end{bmatrix} = \begin{bmatrix} B \\ D \end{bmatrix} \begin{bmatrix} B^\top & D^\top \end{bmatrix}.$$

Output: State space realization's parameters (A,B,C,D) of the system defined by R_{yy}.

The transition from the spectral density S to the ARMA representation (A,M) is the *spectral factorization* problem (Kucera, 1991):

$$S(z) = H(z)H^*(z), \quad \text{with } H(z) = A^{-1}(z)M(z) \text{ stable.}$$

The transformation from (A,M) to a state space representation is

$$A = \begin{bmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & I \\ -A_0 & -A_1 & \cdots & \cdots & -A_{\ell-1} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ I \end{bmatrix}, \qquad ((A,M) \mapsto (A,B,C,D))$$

$$C = \begin{bmatrix} C_0 & C_1 & \cdots & C_{\ell-1} \end{bmatrix}, \qquad D = M_\ell,$$

where $C_0, C_1, \ldots, C_{n-1}$ are the coefficients of $M(z) - M_\ell A(z)$.

Auto Regressive Moving Average eXogenous (ARMAX) systems

The set $\mathscr{B} \subseteq (\mathbb{R}^{\mathfrak{q}})^{\mathbb{Z}}$ is a behavior of an ARMAX system if there are polynomial matrices Y, U, and E, with Y square and nonsingular, such that $\mathscr{B}$ is the set of processes $w := \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ satisfying the difference equation

$$Y(\sigma)y + U(\sigma)u = E(\sigma)\varepsilon, \qquad \text{(ARMAX)}$$

with ε white, normalized, independent of u process. Under generic conditions, u is an input and y is an output, *i.e.*, for all u there are y and $\varepsilon \perp u$ satisfying (ARMAX).

The polynomial matrix R is *left prime* if any factorization $R = FR'$ with F square implies that F is unimodular, *i.e.*, $\det(F)$ is a nonzero constant. Equivalently, R is left prime if $R(\lambda)$ has full row rank for all $\lambda \in \mathbb{C}$. A full row rank polynomial matrix R, can be factored as $R = FR'$, with F square and R' left prime. Using this factorization we can refine the representation (ARMAX) as follows:

$$A(\sigma)\big(P(\sigma)y + Q(\sigma)u\big) = M(\sigma)\varepsilon, \qquad \text{(ARMAX')}$$

where A and P are square and nonsingular, A is Schur, and $R := \begin{bmatrix} P & Q \end{bmatrix}$ is left prime. The polynomial matrix

- A represents the *AR (autoregressive)-part*,
- M the *MA (moving average)-part*, and
- $R = \begin{bmatrix} P & Q \end{bmatrix}$ the *X (exogenous)-part* of the ARMAX system.

The rational function $G := P^{-1}Q$ is the transfer function of the deterministic X-part. We do not require G to be proper, *i.e.*, we do not impose causality.

2.4 Exact and approximate data modeling

This section defines formally the problem considered in the book. In order to treat static as well as dynamic, linear as well as nonlinear, and deterministic as well as stochastic modeling problems using unified terminology and notation, we need an abstract setting that is general enough to accommodate all envisaged applications. First, we define such a setting using the notions of a model as a set and model complexity as a "size" of the model. Then, we discuss the simplest data modeling situation, when the model is required to fit the data exactly. In this case the problem, called exact modeling, is to minimize the model complexity subject to the constraint that the data is fitted exactly. The solution of this problem is called the *most powerful unfalsified model*. In the more general case of approximate modeling, the problem is biobjective: minimization of the fitting error as well as the model complexity. We define two fundamentally different ways of quantifying the fitting error, called *misfit and latency*. The misfit between a given model and the data is the size of the minimal perturbation on the data that renders the data exact. The latency is the size of the minimal latent input that makes the data consistent with the model. In a stochastic estimation setting, misfit and latency correspond to, respectively, errors-in-variables and ARMAX modeling.

2.4.1 General setting for data modeling

The data $\mathscr{D}$ and a model $\mathscr{B}$ for the data are subsets of a *universal set* $\mathscr{U}$ of possible observations. In static modeling problems, $\mathscr{U}$ is a real $\mathtt{q}$-dimensional vector space $\mathbb{R}^{\mathtt{q}}$, *i.e.*, the observations are real valued vectors. In dynamic modeling problems, $\mathscr{U}$ is a function space $(\mathbb{R}^{\mathtt{q}})^{\mathscr{T}}$, with $\mathscr{T}$ being $\mathbb{Z}$ in the discrete-time case and $\mathbb{R}$ in the continuous-time case.

Note 2.7 (Categorical data and finite automata). In modeling problems with categorical data and finite automata, the universal set $\mathscr{U}$ is discrete and may be finite.

We consider data sets $\mathscr{D}$ consisting of a finite number of observations

$$\mathscr{D} = \{ w_{\mathrm{d},1}, \ldots, w_{\mathrm{d},N} \} \subset \mathscr{U} .$$

In dynamic modeling problems, the $w_{\mathrm{d},j}$'s are trajectories, *i.e.*,

$$w_{\mathrm{d},j} = \big(w_{\mathrm{d},j}(1), \ldots, w_{\mathrm{d},j}(T_j) \big), \quad \text{with } w_{\mathrm{d},j}(t) \in \mathbb{R}^{\mathtt{q}} \text{ for all } t .$$

In static modeling problems, an observation $w_{\mathrm{d},j}$ is a vector $w_{\mathrm{d},j} \in \mathbb{R}^{\mathtt{q}}$ and the alternative notation $d_j = w_{\mathrm{d},j}$ is used in order to emphasise the fact that the observations do not depend on time. In dynamic problems, the data $\mathscr{D}$ often consists of a single trajectory $w_{\mathrm{d},1}$, in which case the index 1 is skipped and $\mathscr{D}$ is identified with w_{d}.

Note 2.8 (Given data vs general trajectory). In order to distinguish a general trajectory w of the system from the given data w_{d} (a specific trajectory) we use the subscript "d" in the notation of the given data.

A *model class* $\mathscr{M}$ is a set of sets of $\mathscr{U}$, *i.e.*, $\mathscr{M}$ is an element of the power set $2^{\mathscr{U}}$ of $\mathscr{U}$ (*i.e.*, the set of all sets). We consider the generic model classes of

- linear static models $\mathscr{L}_0$,
- linear time-invariant models $\mathscr{L}$, and
- polynomial static models $\mathscr{P}$ (see Chapter 7).

In some cases, however, subclasses of the generic classes above are of interest. For examples, the controllable and finite impulse response model subclasses of the class of linear time-invariant models, and the ellipsoids subclass of the class of second order polynomial models (conic sections).

The complexity $\mathtt{c}$ of a model $\mathscr{B}$ is a vector of positive integers

$$\mathtt{c}(\mathscr{B}) := \begin{cases} \mathtt{m}(\mathscr{B}) = \dim(\mathscr{B}), & \text{if } \mathscr{B} \in \mathscr{L}_0, \\ \big(\mathtt{m}(\mathscr{B}), \ell(\mathscr{B})\big) \text{ or } \big(\mathtt{m}(\mathscr{B}), \mathtt{n}(\mathscr{B})\big), & \text{if } \mathscr{B} \in \mathscr{L}, \\ \big(\mathtt{m}(\mathscr{B}), \deg(R)\big), \text{ where } \mathscr{B} = \ker(R), & \text{if } \mathscr{B} \in \mathscr{P}. \end{cases} \qquad (\mathtt{c}(\mathscr{B}))$$

Complexities are compared in this book by the lexicographic ordering, *i.e.*, two complexities are compared by comparing their corresponding elements in the increasing order of the indexes. The first time an index is larger, the corresponding complexity is declared larger. For linear time-invariant dynamic models, this convention and the ordering of the elements in $\mathtt{c}(\mathscr{B})$ imply that a model with more inputs is always more complex than a model with less inputs irrespective of their orders.

The complexity $\mathtt{c}$ of a model class $\mathscr{M}$ is the largest complexity of a model in the model class. Of interest is the restriction of the generic model classes $\mathscr{M}$ to subclasses $\mathscr{M}_{\mathtt{c}_{\max}}$ of models with bounded complexity, *e.g.*, $\mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell_{\max}}$, with $\mathtt{m} < \mathtt{q}$.

2.4.2 Exact data modeling

A model $\mathscr{B}$ is an *exact model* for the data $\mathscr{D}$ if the data is constrained in the model, $\mathscr{D} \subset \mathscr{B}$. Otherwise, it is an *approximate model*. An exact model for the data may not exist in a model class $\mathscr{M}_{\mathtt{c}_{\max}}$ of bounded complexity. This is generically the case when the data is noisy and the data set $\mathscr{D}$ is large enough (relative to the model complexity). A practical data modeling problem must involve approximation. Our starting point, however, is the simpler problem of exact data modeling.

Problem 2.9 (Exact data modeling). Given data $\mathscr{D} \subset \mathscr{U}$ and a model class $\mathscr{M}_{\mathtt{c}_{\max}} \in 2^{\mathscr{U}}$, find a model $\widehat{\mathscr{B}}$ in $\mathscr{M}_{\mathtt{c}_{\max}}$ that contains the data and has minimal (in the lexicographic ordering) complexity or assert that such a model does not exist, *i.e.*,

$$\text{minimize} \quad \text{over } \mathscr{B} \in \mathscr{M}_{\mathtt{c}_{\max}} \quad \mathtt{c}(\mathscr{B}) \quad \text{subject to} \quad \mathscr{D} \subset \mathscr{B} \qquad \text{(EM)}$$

The following question about existence of exact model occurs.

(Representability) Under what conditions on the data $\mathscr{D}$ and the model class $\mathscr{M}_{\mathtt{c}_{\max}}$ does a solution to problem (EM) exist?

If a solution exists, it is unique. This unique solution is called the *most powerful unfalsified model* for the data $\mathscr{D}$ in the model class $\mathscr{M}_{\mathtt{c}_{\max}}$ and is denoted by $\mathscr{B}_{\mathrm{mpum}}(\mathscr{D})$. (The model class $\mathscr{M}_{\mathtt{c}_{\max}}$ is not a part of the notation $\mathscr{B}_{\mathrm{mpum}}(\mathscr{D})$ and is understood from the context.)

Suppose that the data $\mathscr{D}$ is generated by a model $\bar{\mathscr{B}}$ in the model class $\mathscr{M}_{\mathtt{c}_{\max}}$, *i.e.*,

$$\mathscr{D} \subset \bar{\mathscr{B}} \in \mathscr{M}_{\mathtt{c}_{\max}}.$$

Then, the exact modeling problem has a solution in the model class $\mathscr{M}_{\mathtt{c}_{\max}}$, however, the solution $\mathscr{B}_{\mathrm{mpum}}(\mathscr{D})$ may not be equal to the data generating model $\bar{\mathscr{B}}$.

(Identifiability) Under what conditions on the data $\mathscr{D}$, the data generating model $\bar{\mathscr{B}}$, and the model class $\mathscr{M}_{\mathtt{c}_{\max}}$, the most powerful unfalsified model $\mathscr{B}_{\mathrm{mpum}}(\mathscr{D})$ in $\mathscr{M}_{\mathtt{c}_{\max}}$ coincides with the data generating model $\bar{\mathscr{B}}$?

Example 2.10 (Representability in $\mathscr{L}_{0,\mathtt{m}}$). Existence of a linear static model $\widehat{\mathscr{B}}$ of bounded complexity $\mathtt{m}$ for the data $\mathscr{D}$ is equivalent to rank deficiency of the matrix

$$\Phi(\mathscr{D}) := \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix} \in \mathbb{R}^{\mathtt{q}\times N},$$

composed of the data. Moreover, the rank of the matrix $\Phi(\mathscr{D})$ is equal to the *minimal* dimension of an exact model for $\mathscr{D}$

$$\text{existence of } \widehat{\mathscr{B}} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},0}, \text{ such that } \mathscr{D} \subset \widehat{\mathscr{B}} \quad \Longleftrightarrow \quad \operatorname{rank}\big(\Phi(\mathscr{D})\big) \leq \mathtt{m}. \qquad (*)$$

The exact model

$$\widehat{\mathscr{B}} = \operatorname{image}\big(\Phi(\mathscr{D})\big) \qquad (**)$$

of minimal dimension

$$\mathtt{c}(\mathscr{B}) = \operatorname{rank}\big(\Phi(\mathscr{D})\big)$$

always exists and is unique.

The equivalence $(*)$ between data modeling and the concept of rank is the basis for application of linear algebra and matrix computations to linear data modeling. Indeed, $(**)$ provides an *algorithm* for exact linear static data modeling. As shown next, exact data modeling has also direct relevance to approximate data modeling.

2.4.3 Approximate data modeling

When an exact model does not exist in the considered model class, an approximate model that is "close" to the data is aimed at instead. Closeness is measured by a suitably defined criterion. This leads to the following data modeling problem.

Problem 2.11 (Approximate data modeling). Given data $\mathscr{D} \subset \mathscr{U}$, a model class $\mathscr{M}_{\mathtt{c}_{\max}} \in 2^{\mathscr{U}}$, and a measure $\operatorname{error}(\mathscr{D},\mathscr{B})$ for the lack of fit of the data $\mathscr{D}$ by a model $\mathscr{B}$, find a model $\widehat{\mathscr{B}}$ in $\mathscr{M}_{\mathtt{c}_{\max}}$ that minimizes the lack of fit, *i.e.*,

$$\text{minimize} \quad \text{over } \mathscr{B} \in \mathscr{M}_{\mathtt{c}_{\max}} \quad \operatorname{error}(\mathscr{D},\mathscr{B}). \qquad \text{(AM)}$$

Since an observation w is a point in and the model $\mathscr{B}$ is a subset of the data space $\mathscr{U}$, a natural measure for the lack of fit $\operatorname{error}(w,\mathscr{B})$ between w and $\mathscr{B}$ is the *geometric distance* defined by the orthogonal projection of w on $\mathscr{B}$

$$\operatorname{dist}(w,\mathscr{B}) := \min_{\widehat{w}\in\mathscr{B}} \big\|w - \widehat{w}\big\|_2. \qquad (\operatorname{dist}(w,\mathscr{B}))$$

The auxiliary variable $\widehat{w}$ is the best approximation of w in $\mathscr{B}$.

For the set of observations $\mathscr{D}$, we define the distance from $\mathscr{D}$ to $\mathscr{B}$ as

$$\operatorname{dist}(\mathscr{D},\mathscr{B}) := \min_{\widehat{w}_1,\ldots,\widehat{w}_N\in\mathscr{B}} \sqrt{\textstyle\sum_{j=1}^N \big\|w_{\mathrm{d},j} - \widehat{w}_j\big\|_2^2}. \qquad \text{(dist)}$$

The set of points $\widehat{\mathscr{D}} = \{\widehat{w}_1,\ldots,\widehat{w}_N\}$ in the definition of (dist) is an approximation of the data $\mathscr{D}$ in the model $\mathscr{B}$. Note that problem (dist) is separable, *i.e.*, it decouples into N independent problems ($\operatorname{dist}(w,\mathscr{B})$).

Algorithms for computing the geometric distance are discussed in Section 4.2.3, in the case of linear models, and in Chapter 7, in the case of polynomial models.

Example 2.12 (Geometric distance for linear and quadratic models). The two plots in Figure 2.6 illustrate the geometric distance (dist) from a set of eight data points $d_1 = \left[\begin{smallmatrix} x_1 \\ y_1 \end{smallmatrix}\right], \ldots, d_8 = \left[\begin{smallmatrix} x_8 \\ y_8 \end{smallmatrix}\right]$ in the plane to, respectively, linear $\mathscr{B}_1$ and quadratic $\mathscr{B}_2$ models. In order to compute the geometric distance, we project the data points on the models. This is a simple task (linear least squares problem) for linear models but a nontrivial task (nonconvex optimization problem) for nonlinear models. In contrast, the algebraic "distance" (not visualised in the figure) has no simple geometrical interpretation but is easy to compute for linear and nonlinear models alike.

An alternative distance measure, called *algebraic distance*, is based on a kernel representation $\mathscr{B} = \ker(R)$ of the model $\mathscr{B}$. Since R is a mapping from $\mathscr{U}$ to $\mathbb{R}^{\mathtt{g}}$, such that $w \in \mathscr{B}$ if and only if $R(w) = 0$, we have that $\|R(w)\|_{\mathrm{F}} > 0$ if and only if $w \notin \mathscr{B}$. The algebraic "distance" measures the lack of fit between w and $\mathscr{B}$ by the "size" $\|R(w)\|_{\mathrm{F}}$ of the residual $R(w)$. For a data set $\mathscr{D}$, we define

$$\operatorname{dist}'(\mathscr{D},\mathscr{B}) := \sqrt{\textstyle\sum_{j=1}^N \big\|R\big(w_{\mathrm{d},j}\big)\big\|_{\mathrm{F}}^2}, \qquad (\operatorname{dist}')$$

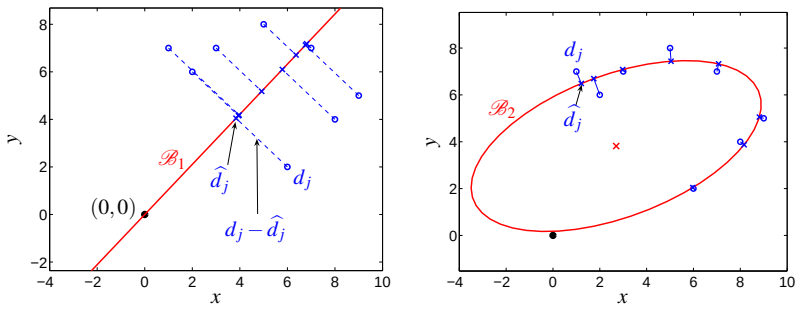


Fig. 2.6: The geometric distance $\operatorname{dist}(\mathscr{D}, \mathscr{B})$ from the data $\mathscr{D}$ to a model $\mathscr{B}$ is computed by projecting all data points $d_1, \ldots, d_N$ on the model $\mathscr{B}$. In the linear case (left plot), the projection is a linear operation (multiplication by a projection matrix). In the nonlinear case, the projection is defined by a nonconvex optimization problem, which, in general, has no closed form solution.

The algebraic distance depends on the choice of the parameter R in a kernel representation of the model, while the geometric distance is representation invariant. In addition, the algebraic distance is not invariant to a rigid transformation. However, a modification of the algebraic distance that is invariant to a rigid transformation is presented in Section 7.3.

Note 2.13 (Approximate modeling in the case of exact data). If an exact model $\mathscr{B}$ exists in the model class $\mathscr{M}_{\mathsf{c}_{\max}}$, then $\mathscr{B}$ is a global optimum point of the approximate modeling problem (AM) (irrespective of the approximation criterion f being used). Indeed,

$$\mathscr{D} \subset \mathscr{B} \quad \Longleftrightarrow \quad \operatorname{dist}(\mathscr{D}, \mathscr{B}) = \operatorname{dist}'(\mathscr{D}, \mathscr{B}) = 0.$$

An optimal approximate model, *i.e.*, a solution of (AM), however, need not be unique. In contrast, the most powerful unfalsified model is unique. This is due to the fact that (AM) imposes an upper bound but does not minimize the model complexity, while (EM) minimizes the model complexity. As a result, when $\mathsf{c}\big(\mathscr{B}_{\text{mpum}}(\mathscr{D})\big) < \mathsf{c}_{\max}$, (AM) has a nonunique solution. In the next section, we present a more general approximate data modeling problem formulation that minimizes simultaneously the complexity as well as the fitting error.

The terms "geometric" and "algebraic" distance comes from the computer vision application of the methods for fitting curves and surfaces to data. In the system identification community, the geometric fitting method is related to the *misfit* approach and the algebraic fitting criterion is related to the *latency* approach. Misfit and latency computation are data smoothing operations. For linear time-invariant systems, the misfit and latency can be computed efficiently by Riccati type recursions.

2.4.4 Stochastic data modeling

In the statistics literature, the geometric fitting is related to *errors-in-variable (EIV) estimation* and the algebraic fitting is related to the *classical regression* and the *ARMAX modeling* (Ljung, 1999; Söderström and Stoica, 1989), see Table 2.2.

Table 2.2: The computer vision, system identification, and statistics communities use different terms for the misfit and latency approximate fitting principles. From a mathematical point of view, the essential difference of the fitting principles is in using a relation or a function.

	computer vision	system identification	statistics	mathematics
misfit	geometric fitting	dynamic EIV	static EIV	relation
latency	algebraic fitting	ARMAX	regression	function

Example 2.14 (Geometric fit and errors-in-variables modeling). From a statistical point of view, the approximate data modeling problem (AM) with the geometric fitting criterion (dist) yields a maximum likelihood estimator for the true model $\bar{\mathscr{B}}$ in the errors-in-variables setup

$$w_{\mathrm{d},j} = \bar{w}_j + \widetilde{w}_j, \tag{EIV}$$

where

$$\overline{\mathscr{D}} := \{\, \bar{w}_1, \ldots, \bar{w}_N \,\} \subset \bar{\mathscr{B}}$$

is the true data and $\widetilde{\mathscr{D}} := \{\, \widetilde{w}_1, \ldots, \widetilde{w}_N \,\}$ is the measurement noise, which is assumed to be zero mean, independent, Gaussian, with covariance matrix $\sigma^2 I$.

Example 2.15 (Algebraic fit by a linear static model and regression). A linear model class, defined by the input/output representation $\mathscr{B}_{\text{i/o}}(X)$ and algebraic fitting criterion (dist′), where

$$w := \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \quad \text{and} \quad R(w) := X^\top u - y$$

lead to the ordinary linear least squares problem

$$\text{minimize} \quad \text{over } X \in \mathbb{R}^{\mathtt{m}\times\mathtt{p}} \quad \left\| X^\top \begin{bmatrix} u_{\mathrm{d},1} & \cdots & u_{\mathrm{d},N} \end{bmatrix} - \begin{bmatrix} y_{\mathrm{d},1} & \cdots & y_{\mathrm{d},1} \end{bmatrix} \right\|_{\mathrm{F}}. \tag{LS}$$

The statistical setting for the least squares approximation problem (LS) is the classical regression model $R(w_{\mathrm{d},j}) = e_j$, where $e_1, \ldots, e_N$ are zero mean independent and identically distributed random variables. The least squares approximate solution is the best linear unbiased estimator for the regression model.

2.4.5 Complexity–accuracy trade-off

Data modeling is a map from the data $\mathscr{D}$ to a model $\mathscr{B}$ in the model class $\mathscr{M}$:

$$\text{data set } \mathscr{D} \subset \mathscr{U} \quad \xrightarrow{\text{data modeling problem}} \quad \text{model } \mathscr{B} \in \mathscr{M} \in 2^{\mathscr{U}}.$$

A data modeling problem is defined by specifying the model class $\mathscr{M}$ and one or more modeling criteria. Basic criteria in any data modeling problem are:

- "simple" model, measured by the model complexity $\mathsf{c}(\mathscr{B})$, and
- "good" fit of the data by the model, measured by cost function error$(\mathscr{D},\mathscr{B})$.

Small complexity $\mathsf{c}(\mathscr{B})$ and small error$(\mathscr{D},\mathscr{B})$, however, are contradicting objectives, so that a core issue throughout data modeling is the complexity–accuracy trade-off. A generic data modeling problem is: Given a data set $\mathscr{D} \in \mathscr{U}$ and a measure error for the fitting error, solve the multi-objective optimization problem:

$$\text{minimize} \quad \text{over } \mathscr{B} \in \mathscr{M} \quad \begin{bmatrix} \mathsf{c}(\mathscr{B}) \\ \text{error}(\mathscr{D},\mathscr{B}) \end{bmatrix}. \tag{DM}$$

Two possible scalarizations: low-rank approximation and rank minimization

The data set $\mathscr{D}$, can be parametrized by a real vector $p \in \mathbb{R}^{n_p}$. (Think of the vector p as a representation of the data in the computer memory.) For a linear static and autonomous linear time-invariant model $\mathscr{B}$ and exact data $\mathscr{D}$, there is a relation between the model complexity and the rank of a data matrix $\mathscr{S}(p)$:

$$\mathsf{c}\big(\mathscr{B}_{\text{mpum}}(\mathscr{D})\big) = \text{rank}\,\big(\mathscr{S}(p)\big). \tag{$*$}$$

The mapping $\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m\times n}$ from the data parameter vector p to the data matrix $\mathscr{S}(p)$ depends on the application. For example, $\mathscr{S}(p) = \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$ is unstructured in the case of linear static modeling (see Example 2.10) and $\mathscr{S}(p) = \mathscr{H}_{\ell+1}(w_{\mathrm{d}})$ is Hankel structured in the case of autonomous linear time-invariant dynamic model identification,

Let p be the parameter vector for the data $\mathscr{D}$ and $\widehat{p}$ be the parameter vector for the data approximation $\widehat{\mathscr{D}}$. The geometric distance dist$(\mathscr{D},\mathscr{B})$ can be expressed in terms of the parameter vectors p and $\widehat{p}$ as

$$\text{minimize} \quad \text{over } \widehat{p} \quad \|p-\widehat{p}\|_2 \quad \text{subject to} \quad \widehat{\mathscr{D}} \subset \mathscr{B}.$$

Moreover, the norm in the parameter space $\mathbb{R}^{n_p}$ can be chosen as weighted 1-, 2-, and ∞-(semi)norms:

$$\begin{aligned} \|\widetilde{p}\|_{v,1} &:= \|v \odot \widetilde{p}\|_1 := \textstyle\sum_{i=1}^{n_p} |v_i \widetilde{p}_i|, \\ \|\widetilde{p}\|_{v,2} &:= \|v \odot \widetilde{p}\|_2 := \sqrt{\textstyle\sum_{i=1}^{n_p} (v_i \widetilde{p})^2}, \\ \|\widetilde{p}\|_{v,\infty} &:= \|v \odot \widetilde{p}\|_\infty := \max_{i=1,\ldots,n_p} |v_i \widetilde{p}_i|, \end{aligned} \qquad (\|\cdot\|_v)$$

where w is a vector with nonnegative elements, specifying the weights, and $\odot$ is the element-wise (Hadamard) product.

Using the data parametrization ($*$) and one of the distance measures ($\|\cdot\|_v$), the data modeling problem (DM) in the case of geometric distance becomes the biobjective matrix approximation problem:

$$\text{minimize} \quad \text{over } \widehat{p} \quad \begin{bmatrix} \text{rank}\,\big(\mathscr{S}(\widehat{p})\big) \\ \|p-\widehat{p}\| \end{bmatrix}. \tag{DM'}$$

Two possible ways to scalarize the biobjective problem (DM') are:

1. misfit minimization subject to a bound r on the model complexity

$$\text{minimize} \quad \text{over } \widehat{p} \quad \|p-\widehat{p}\| \quad \text{subject to} \quad \text{rank}\,\big(\mathscr{S}(\widehat{p})\big) \leq r. \tag{SLRA}$$

2. model complexity minimization subject to a bound ε on the fitting error

$$\text{minimize} \quad \text{over } \widehat{p} \quad \text{rank}\,\big(\mathscr{S}(\widehat{p})\big) \quad \text{subject to} \quad \|p-\widehat{p}\| \leq \varepsilon. \tag{RM}$$

Problem (SLRA) is a structured low-rank approximation problem and problem (RM) is a rank minimization problem.

By varying the parameters r and ε from zero to infinity, both problems sweep the trade-off curve (set of Pareto optimal solutions) of (DM'). Note, however, that r ranges over the natural numbers and only small values are of practical interest. In addition, in applications often a "suitable" value for r can be chosen a priori or is even a part of the problem specification. In contrast, ε is a positive real number and is data dependent, so that a "suitable" value is not readily available. These considerations, suggest that the structured low-rank approximation problem is a more convenient scalarization of (DM') for solving practical data modeling problems.

Convex relaxation algorithms for solving (DM') are presented in Section 4.3.

2.5 Notes and references

Deterministic linear time-invariant systems

Linear time-invariant system theory is the basis for signal processing, communication, and control. Consequently, there are many excellent books covering system theory on different levels. A popular undergraduate level textbook is (Oppenheim and Willsky, 1996). Graduate level classics are (Brockett, 1970; Kailath, 1981;

Zadhe and Desoer, 1963). More modern graduate level texts are (Antsaklis and Michel, 1997; Sontag, 1998). (Åström and Murray, 2008; Luenberger, 1979) have unique presentation style with an outlook to applications. The behavioral approach to system theory is covered in the book (Polderman and Willems, 1998).

Impulse response realization: The impulse response realization problem was first solved in (Ho and Kalman, 1966). See also (Kalman, 1979; Kalman et al, 1969). The classic papers of what we call Kung's method are (Kung, 1978; Zeiger and McEwen, 1974). Although the impulse response realization is an important part of system theory, it is missing from most undergraduate level and some graduate level textbooks. An exposition is given in (Sontag, 1998, Sections 6.5–8).

Powerful unfalsified model: The concept of the most powerful unfalsified model is introduced in the behavioral setting (Willems, 1986b, Definition 4). See also (Antoulas and Willems, 1993; Kuijper, 1997; Kuijper and Willems, 1997; Willems, 1997). Methods for the computation of the most powerful unfalsified model are developed in the context of exact (deterministic) system identification, see, (Van Overschee and De Moor, 1996) and (Markovsky et al, 2006, Chapter 7).

Stochastic linear time-invariant systems

A classic reference on stochastic processes is (Papoulis, 1991). Introductory text on stochastic systems is (Åström, 1970). Stochastic systems are used in the formulation of classic estimation and control problems, such as the Kalman filter and the linear quadratic Gaussian (LQG) control. Later on it is shown, however, that these problems admit purely deterministic formulation ($\mathscr{H}_2$ estimation and control) (Fagnani and Willems, 1997; Willems, 2004; Zhou et al, 1996), see also (Willems, 2010).

Parametric and non-parametric estimation of the correlation sequence is a part of the spectral analysis of stochastic signals (Kay, 1988; Stoica and Moses, 2005). The stochastic realization problem is treated in (Akaike, 1974). Behavioral definition of a static stochastic system is given in (Willems, 2013). The generalization of these results to dynamical systems is still missing.

Misfit and latency

The terminology misfit and latency for the evaluation of the discrepancy between the data and the model comes from (Willems, 1987). See also (Lemmerling and De Moor, 2001), where an identification problem combining the two criteria is proposed. The computation of the misfit is treated in the context of errors-in-variables Kalman filtering in (Markovsky et al, 2002) for continuous-time systems and in (Markovsky and De Moor, 2005) for discrete-time systems.

Exercises

2.1 (Relations among $\operatorname{rank}(P)$, $\operatorname{rank}(R)$, and $\dim(\mathscr{B})$, for a linear static model $\mathscr{B}$). Prove that for a linear static model $\mathscr{B} \in \mathscr{L}_0^{\mathtt{q}}$ with image representation $\mathscr{B} = \operatorname{image}(P)$ and kernel representation $\mathscr{B} = \ker(R)$,

1. $\operatorname{rank}(P) = \dim(\mathscr{B})$,
2. $\operatorname{rank}(R) = \mathtt{q} - \dim(\mathscr{B})$.

2.2 ($\mathscr{B}_1 \stackrel{?}{=} \mathscr{B}_2$). Explain how to check whether two linear static models, specified by kernel or image representations, are equivalent. Implement the methods.

2.3 (Input/output partitions that are not possible). Give an example of a linear static model $\mathscr{B} \in \mathscr{L}_{1,0}^3$, for which neither w_2 nor w_3 can be selected as an input in an input/output partitioning of the variables. Form a conjecture of when a subset of the variables of a linear static model $\mathscr{B} \in \mathscr{L}_{\mathtt{m},0}^{\mathtt{q}}$ can not be chosen as inputs.

2.4 (Initial conditions specification by a trajectory). Consider a minimal input/state/output representation $\mathscr{B} = \mathscr{B}_{\text{i/s/o}}(A,B,C,D)$ of a system $\mathscr{L}_{\mathtt{m},\ell}$.

1. Show that $w_{\text{p}} := \big(w(-\ell+1), \ldots, w(0)\big) \in \mathscr{B}|_\ell$ uniquely determines $x(0)$.
2. Explain how to choose w_{p} in order to "set" a given initial condition $x(0)$.
3. Implement the transitions $w_{\text{p}} \leftrightarrow x(0)$ in functions `wp2x0` and `x02wp`.

2.5 ($w \stackrel{?}{\in} \mathscr{B} = \ker\big(R(\sigma)\big)$).

1. Given a finite sequence w and a polynomial matrix R, propose a method for checking numerically whether w_{d} is a trajectory of the system $\mathscr{B} = \ker\big(R(\sigma)\big)$.
2. Implement the method in a function `w_in_ker(w, r)`.
3. Test the function on the trajectory $\left(\left[\begin{smallmatrix}0\\1\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right] \right)$ and the system

$$\mathscr{B} = \ker\big(R(\sigma)\big), \quad \text{with } R(z) = \begin{bmatrix}1 & -1\end{bmatrix} + \begin{bmatrix}-1 & 1\end{bmatrix} z.$$

2.6 ($\ker\big(R(\sigma)\big) \leftrightarrow \mathscr{B}_{\text{i/o}}(p,q)$). Consider a controllable single-input single-output linear time-invariant system $\mathscr{B}$. Explain how to transform a given kernel representation $\mathscr{B} = \ker\big(R(\sigma)\big)$ into an input/output representation $\mathscr{B} = \mathscr{B}_{\text{i/o}}(p,q)$. Vice versa, explain how to transform $\mathscr{B} = \mathscr{B}_{\text{i/o}}(p,q)$ into $\mathscr{B} = \ker\big(R(\sigma)\big)$. Write MATLAB code that implements the transitions $R \mapsto (p,q)$ and $(p,q) \mapsto R$.

2.7 ($\operatorname{image}\big(P(\sigma)\big) \leftrightarrow \mathscr{B}_{\text{i/o}}(p,q)$). Consider a controllable single-input single-output linear time-invariant system $\mathscr{B}$. Explain how to transform a given image representation $\mathscr{B} = \operatorname{image}\big(P(\sigma)\big)$ into an input/output representation $\mathscr{B} = \mathscr{B}_{\text{i/o}}(p,q)$. Vice versa, explain how to transform $\mathscr{B} = \mathscr{B}_{\text{i/o}}(p,q)$ into $\mathscr{B} = \operatorname{image}\big(P(\sigma)\big)$. Write MATLAB code that implements the transitions $P \mapsto (p,q)$ and $(p,q) \mapsto P$.

References

Abelson H, diSessa A (1986) Turtle Geometry. MIT Press

Akaike H (1974) Stochastic theory of minimal realization. IEEE Trans Automat Contr 19:667–674

Anderson BDO, Moore JB (1979) Optimal Filtering. Prentice Hall

Antoulas A, Willems JC (1993) A behavioral approach to linear exact modeling. IEEE Trans Automat Contr 38(12):1776–1802

Antsaklis PJ, Michel A (1997) Linear Systems. McGraw-Hill

Åström K, Murray R (2008) Feedback systems: An introduction for scientists and engineers. Princeton University Press

Åström KJ (1970) Introduction to Stochastic Control Theory. Academic Press

Brockett R (1970) Finite Dimensional Linear Systems. John Wiley, New York

Fagnani F, Willems JC (1997) Deterministic Kalman filtering in a behavioral framework. Control Lett 32:301–312

Ho BL, Kalman RE (1966) Effective construction of linear state-variable models from input/output functions. Regelungstechnik 14(12):545–592

Kailath T (1981) Linear Systems. Prentice Hall

Kailath T, Sayed AH, Hassibi B (2000) Linear Estimation. Prentice Hall

Kalman RE (1979) On partial realizations, transfer functions, and canonical forms. Acta Polytechnica Scandinavica 31:9–32

Kalman RE, Falb PL, Arbib MA (1969) Topics in Mathematical System Theory. McGraw-Hill

Kay S (1988) Modern spectral estimation: Theory and applications. Prentice-Hall

Kucera V (1991) Factorization of rational spectral matrices: a survey of methods. In: International Conference on Control, IET, pp 1074–1078

Kuijper M (1997) An algorithm for constructing a minimal partial realization in the multivariable case. Control Lett 31(4):225–233

Kuijper M, Willems JC (1997) On constructing a shortest linear recurrence relation. IEEE Trans Automat Contr 42(11):1554–1558

Kung S (1978) A new identification method and model reduction algorithm via singular value decomposition. In: Proc. 12th Asilomar Conf. Circuits, Systems, Computers, Pacific Grove, pp 705–714

Lemmerling P, De Moor B (2001) Misfit versus latency. Automatica 37:2057–2067

Ljung L (1999) System Identification: Theory for the User. Prentice-Hall

Luenberger DG (1979) Introduction to Dynamical Systems: Theory, Models and Applications. John Wiley

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

Markovsky I, Willems JC, De Moor B (2002) Continuous-time errors-in-variables filtering. In: Proc. of the 41st Conf. on Decision and Control, Las Vegas, NV, pp 2576–2581

Markovsky I, Willems JC, Van Huffel S, De Moor B (2006) Exact and Approximate Modeling of Linear Systems: A Behavioral Approach. SIAM

Oppenheim A, Willsky A (1996) Signals and Systems. Prentice Hall

Papoulis A (1991) Probability, random variables, and stochastic processes. McGraw-Hill

Picci G (1991) Stochastic Realization Theory, Springer, pp 213–229

Polderman J, Willems JC (1998) Introduction to Mathematical Systems Theory. Springer-Verlag, New York

Rapisarda P, Willems JC (1997) State maps for linear systems. SIAM J Control Optim 35(3):1053–1091

Söderström T, Stoica P (1989) System Identification. Prentice Hall

Sontag ED (1998) Mathematical control theory: Deterministic finite dimensional systems. Springer-Verlag

Stoica P, Moses R (2005) Spectral Analysis of Signals. Prentice Hall

Van Overschee P, De Moor B (1996) Subspace identification for linear systems: Theory, implementation, applications. Kluwer, Boston

Willems JC (1986a) From time series to linear system—Part I. Finite dimensional linear time invariant systems. Automatica 22(5):561–580

Willems JC (1986b) From time series to linear system—Part II. Exact modelling. Automatica 22(6):675–694

Willems JC (1987) From time series to linear system—Part III. Approximate modelling. Automatica 23(1):87–115

Willems JC (1989) Models for dynamics. Dynamics reported 2:171–269

Willems JC (1991) Paradigms and puzzles in the theory of dynamical systems. IEEE Trans Automat Control 36(3):259–294

Willems JC (1996) Fitting data sequences to linear systems. In: Byrnes C, Datta B, Gilliam D, Martin C (eds) Progress in Systems and Control, Birkhäuser, pp 405–416

Willems JC (1997) On interconnections, control, and feedback. IEEE Trans Automat Contr 42:326–339

Willems JC (2004) Deterministic least squares filtering. J Econometrics 118:341–373

Willems JC (2006) Control of Uncertain Systems: Modelling, Approximation, and Design, vol 329, Springer, chap Thoughts on system identification, pp 389–416

Willems JC (2007) The behavioral approach to open and interconnected systems: Modeling by tearing, zooming, and linking. Control Systems Magazine 27:46–99

Willems JC (2010) Probability in control? European Journal on Control 16:436–440

Willems JC (2013) Open stochastic systems. IEEE Trans Automat Control 58(2):406–421

Zadhe LA, Desoer CA (1963) Linear System Theory : The State Space Approach. Prentice Hall

Zeiger H, McEwen A (1974) Approximate linear realizations of given dimension via Ho's algorithm. IEEE Trans Automat Contr 19:153–153

Zhou K, Doyle J, Glover K (1996) Robust and Optimal Control. Prentice Hall

Chapter 3
Exact modeling

> *A given phenomenon can be described by many models. A well known example is the Ptolemaic and the Copernican models of the solar system. Both described the observed plant movements up to the accuracy of the instruments of the 15th century.*
> *A reasonable scientific attitude to a model to be used is that it has so far not been* falsified*. That is to say that the model is not in apparent contradiction to measurements and observations.*
> ...
> *Another valid scientific attitude is that* among all the (so far) unfalsified possible models we should use the simplest one. *This statement is a variant of Occam's razor. It obviously leaves room for subjective aesthetic and pragmatic interpretations of what "simplest" should mean. It should be stressed that all unfalsified models are legitimate, and that any discussion on which one of these is the "true" one is bound to be futile.*
>
> *K. Lindgren*

This chapter develops methods for computing the most powerful unfalsified model for the data in the model class of linear time-invariant systems. The first problem considered is a realization of an impulse response. This is a special system identification problem when the given data is an impulse response. The method presented, known as Kung's method, is the basis for more general exact identification methods known as subspace methods.

The second problem considered is computation of the impulse response from a general trajectory of the system. We derive an algorithm that is conceptually simple—it requires only a solution of a system of linear equations. The main idea of computing a special trajectory of the system from a given general one is used also in data-driven simulation and control problems (Chapter 6).

The next topic is identification of stochastic systems. We show that the problem of ARMAX identification splits into three subproblems: 1) identification of the deterministic part, 2) identification of the AR-part, and 3) identification of the MA-part. Subproblems 1 and 2 are equivalent to deterministic identification problem.

The last topic considered in the chapter is computation of the most powerful unfalsified model from a trajectory with missing values. This problem can be viewed as Hankel matrix completion or, equivalently, computation of the kernel of a Hankel matrix with missing values. Methods based on matrix completion using the nuclear norm heuristic and ideas from subspace identification are presented.

3.1 Kung's realization method

The aim of the impulse response realization problem is to compute a state space representation $\mathscr{B}_{\text{i/s/o}}(A,B,C,D)$ of the most powerful unfalsified model $\mathscr{B}_{\text{mpum}}(H)$, see Section 2.2. Finding the model parameters A,B,C,D can be done by computing a rank revealing factorization of a Hankel matrix constructed from the data. Let

$$\mathscr{H}_{\mathtt{n}+1,\mathtt{n}+1}(\sigma H) = \Gamma\Delta, \quad \text{where } \Gamma \in \mathbb{R}^{\mathtt{p}(\mathtt{n}+1)\times\mathtt{n}} \text{ and } \Delta \in \mathbb{R}^{\mathtt{n}\times\mathtt{m}(\mathtt{n}+1)}$$

be a rank revealing factorization of the Hankel matrix $\mathscr{H}_{\mathtt{n}+1,\mathtt{n}+1}(\sigma H)$. The Hankel structure implies that the factors Γ and Δ are observability and controllability matrices, *i.e.*, there are matrices $A \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}$, $C \in \mathbb{R}^{\mathtt{p}\times\mathtt{n}}$, and $B \in \mathbb{R}^{\mathtt{n}\times\mathtt{m}}$, such that

$$\Gamma = \mathscr{O}_{\mathtt{n}+1}(A,C) \quad \text{and} \quad \Delta = \mathscr{C}_{\mathtt{n}+1}(A,B).$$

Then, $\mathscr{B}_{\text{i/s/o}}(A,B,C,D)$ is the minimal realization of H.

A rank revealing factorization is not unique. For any $\mathtt{n}\times\mathtt{n}$ nonsingular matrix T, a new factorization

$$\mathscr{H}_{\mathtt{n}+1,\mathtt{n}+1}(\sigma H) = \Gamma\Delta = \underbrace{\Gamma T}_{\Gamma'}\underbrace{T^{-1}\Delta}_{\Delta'}$$

is obtained with the same inner dimension. The nonuniqueness of the factorization corresponds to the nonuniqueness of the input/state/output representation of the minimal realization due to a change of the state space bases:

$$\mathscr{B}_{\text{i/s/o}}(A,B,C,D) = \mathscr{B}_{\text{i/s/o}}(T^{-1}AT, T^{-1}B, CT, D).$$

The observability/controllability matrix structure is referred to as the *shift structure*. The parameters B and C of an input/state/output representation of the realization can be obtained directly from the first block elements of Γ and Δ, respectively.

72a $\langle \Gamma,\Delta\rangle \mapsto (A,B,C)$ 72a⟩≡ (74c) 72b▷

```
b = C(:, 1:m); c = O(1:p, :);
```

The parameter A is computed from the overdetermined system of linear equations

$$\sigma^{-1}\Gamma A = \sigma\Gamma. \tag{SE$_1$}$$

(Acting on block matrices, σ / σ^{-1} remove, the first / last block element.)

72b $\langle \Gamma,\Delta\rangle \mapsto (A,B,C)$ 72a⟩+≡ (74c) ◁72a

```
a = O(1:end - p, :) \ O((p + 1):end, :);
```

Note 3.1 (Solution of the shift equation). When a unique solution exists, the code in chunk 72b computes the exact solution. When a solution A of (SE$_1$) does not exist, the same code computes a least squares approximate solution.

Equivalently (in the case of exact data), A can be computed from the Δ factor

$$A\sigma^{-1}\Delta = \sigma\Delta. \tag{SE$_2$}$$

In the case of noisy data (approximate realization problem) or data from a high order system (model reduction problem), (SE$_1$) and (SE$_2$) generically have no exact solutions and their least squares approximate solutions are different.

Implementation

As square as possible Hankel matrix $\mathscr{H}_{i,j}(\sigma H)$ is formed, using all data points, *i.e.*,

$$i = \left\lceil \frac{T\mathtt{m}}{\mathtt{m}+\mathtt{p}} \right\rceil \quad \text{and} \quad j = T - i. \tag{i,j}$$

73a ⟨*dimension of the Hankel matrix* 73a⟩≡ (74c 78e)

```
if ~exist('i', 'var') | ~isreal(i) | isempty(i)
  i = ceil(T * m / (m + p));
end
if ~exist('j', 'var') | ~isreal(j) | isempty(j)
  j = T - i;
elseif j > T - i
  error('Not enough data.')
end
```

The choice (i,j) for the dimension of the Hankel matrix maximazes the order of the realization that can be computed. Indeed, a realization of order $\mathtt{n}$ can be computed from the block-Hankel matrix $\mathscr{H}_{i,j}(\sigma H)$ provided that

$$\mathtt{n} \le \mathtt{n}_{\max} := \min(\mathtt{p}i - 1, \mathtt{m}j).$$

73b ⟨*check* $\mathtt{n} < \min\left(\mathtt{p}i-1, \mathtt{m}j\right)$ 73b⟩≡ (74c)

```
if n > min(i * p - 1, j * m), error('Not enough data'), end
```

The minimal number of samples T of the impulse response that allows identification of a system of order $\mathtt{n}$ is

$$T_{\min} := \left\lceil \frac{\mathtt{n}}{\mathtt{p}} \right\rceil + \left\lceil \frac{\mathtt{n}}{\mathtt{m}} \right\rceil + 1. \tag{$T_{\min}$}$$

The key computational step of the realization algorithm is the factorization of the Hankel matrix. In particular, this step involves rank determination. In finite precision arithmetic, however, rank determination is a nontrivial problem. A numerically reliable way of computing rank is the singular value decomposition

$$\mathscr{H}_{i,j}(\sigma H) = U\Sigma V^\top.$$

73c ⟨*singular value decomposition of* $\mathscr{H}_{i,j}(\sigma H)$ 73c⟩≡ (74c)

```
[U, S, V] = svd(blkhank(h(:, :, 2:end), i, j), 0); s = diag(S);
```

Uses blkhank 26a.

The order $\mathtt{n}$ of the realization is theoretically equal to the rank of the Hankel matrix, which is equal to the number of nonzero singular values $\sigma_1, \ldots, \sigma_{\min(i,j)}$. In practice,

the system's order is estimated as the numerical rank of the Hankel matrix, *i.e.*, the number of singular values greater than a user specified tolerance.

74a ⟨*order selection* 74a⟩≡ (74c)

```
⟨default tolerance tol 41⟩, n = sum(s > tol);
```

Defining the partitioning

$$U =: \overset{\mathtt{n}}{\begin{bmatrix} U_1 & U_2 \end{bmatrix}}, \quad \Sigma =: \overset{\mathtt{n}}{\begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}}^{\mathtt{n}}, \quad \text{and} \quad V =: \overset{\mathtt{n}}{\begin{bmatrix} V_1 & V_2 \end{bmatrix}},$$

the factors Γ and Δ of the rank revealing factorization are chosen as follows

$$\Gamma := U_1\sqrt{\Sigma_1} \quad \text{and} \quad \Delta := \sqrt{\Sigma_1}V_1^\top. \tag{Γ,Δ}$$

74b ⟨*define* Δ *and* Γ 74b⟩≡ (74c)

```
sqrt_s = sqrt(s(1:n))';
O =  sqrt_s(ones(size(U, 1), 1), :) .* U(:, 1:n);
C = (sqrt_s(ones(size(V, 1), 1), :) .* V(:, 1:n))';
```

This choice leads to a *finite-time balanced* realization of $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$, *i.e.*, the finite time controllability and observability Gramians

$$\mathscr{O}_i^\top(A,C)\mathscr{O}_i(A,C) = \Gamma^\top\Gamma \quad \text{and} \quad \mathscr{C}_j(A,B)\mathscr{C}_j^\top(A,B) = \Delta\Delta^\top$$

are equal,

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

The combination of the described realization algorithm with the singular value decomposition based rank revealing factorization (Γ,Δ), *i.e.*, unstructured low-rank approximation, is referred to as *Kung's algorithm*.

74c ⟨$H \mapsto \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$ 74c⟩≡

```
function [sys, hh] = h2ss(h, n, tol, i ,j)
⟨reshape H and define m, p, T 75⟩
⟨dimension of the Hankel matrix 73a⟩
⟨singular value decomposition of H_{i,j}(σH) 73c⟩
if ~exist('n', 'var') | isempty(n)
  ⟨order selection 74a⟩
else
  ⟨check n < min(pi-1, mj) 73b⟩
end
⟨define Δ and Γ 74b⟩
⟨(Γ,Δ) ↦ (A,B,C) 72a⟩, sys = ss(a, b, c, h(:, :, 1), -1);
if nargout > 1, hh = shiftdim(impulse(sys, T), 1); end
```

Defines:
h2ss, used in chunks 78e, 124a, 139b, 140a, 142, and 255.

Following the convention for representing vector and matrix valued trajectories in MATLAB (see, page 26), the impulse response H is stored as an

- $\mathtt{p} \times T$ matrix in the single-input case, and
- $\mathtt{p} \times \mathtt{m} \times T$ tensor $\mathtt{h}(:,:,\mathtt{t}) = H(t)$, in the multi-output case.

75 ⟨*reshape H and define* m, p, *T* 75⟩≡ (74c 138a)

```
if length(size(h)) == 2
  [p, T] = size(h); if p > T, h = h'; [p, T] = size(h); end
  h = reshape(h, p, 1, T);
end
[p, m, T] = size(h);
```

Note 3.2 (Approximate realization by Kung's algorithm). When H is not realizable by a linear time-invariant system of order less than or equal to $\mathtt{n}_{\max}$, *i.e.*, when $\mathscr{H}_{i,j}(\sigma H)$ is full rank, `h2ss` computes an approximate realization $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$ of order $\mathtt{n} \leq \mathtt{n}_{\max}$. The link between the realization problem and Hankel structured low-rank approximation implies that

> Kung's algorithm, implemented in the function `h2ss`, is a method for Hankel structured low-rank approximation: $\mathscr{H}_{i,j}(\sigma H)$ is approximated by $\mathscr{H}_{i,j}(\sigma \widehat{H})$, where $\widehat{H}$ is the impulse response of the $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$.

Note 3.3 (Suboptimality of Kung's algorithm). Used as a method for Hankel structured low-rank approximation, Kung's algorithm is suboptimal. The reason for this is that the factorization $\mathscr{H}_{i,j}(\sigma H) \approx \Gamma\Delta$, performed by the singular value decomposition is an unstructured low-rank approximation and unless the data is exact, Γ and Δ are not extended observability and controllability matrices, respectively. As a result, the shift equations (SE_1) and (SE_2) do not have solutions. Kung's algorithm computes an approximate solution in the least squares sense.

Note 3.4 (Structure enforcing methods). The two approximation steps.

1. unstructured low rank approximation of the Hankel matrix $\mathscr{H}_{i,j}(\sigma H)$, and
2. approximate solution of (SE_1) and (SE_2).

are common to the subspace methods. In contrast, methods based on Hankel structured low-rank approximation do not involve approximation on a second step. The approximation computed on the first step guarantees existence of an exact solution on the second step.

Note 3.5 (Connection between Kung's algorithm and balanced model reduction). Kung's algorithm computes a realization in a finite time $\min(i,j)$ balanced bases. In the case of noisy data or data obtained from a high order model, the computed realization is obtain by truncation of the realization. Therefore, Kung's algorithm can be viewed as a *data-driven method* for finite-time balanced model reduction. The term data-driven refers to the fact that a state space model of the full order system is not constructed. Instead the model reduction is done directly from data from the full order system. The link between Kung's algorithm and model reduction further justifies the choice (i,j) on page 73 for the shape of the Hankel matrix — the choice (i,j) maximizes the horizon for the finite-time balanced realization.

3.2 Impulse response computation

In the realization problem the given data is a special trajectory—the impulse response of the model. Therefore, the realization problem is a special exact identification problem. This section solves the general exact identification problem:

$$w_{\mathrm{d}} \xrightarrow[\text{identification}]{\text{exact}} \mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$$

by reducing it to the realization problem:

$$w_{\mathrm{d}} \xrightarrow[\text{computation (w2h)}]{\text{impulse response}} H_{\mathrm{mpum}} \xrightarrow{\text{realization (h2ss)}} \mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}}). \quad (w_{\mathrm{d}} \mapsto H \mapsto \widehat{\mathscr{B}})$$

First, the impulse response H_{mpum} of the most powerful unfalsified model $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ (see Section 2.4.2) is computed from the given general trajectory w_{d}. Then, an input/state/output representation of $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ is computed from H_{mpum} by a realization algorithm, *e.g.*, Kung's algorithm.

The key observation in finding an algorithm for the computation of the impulse response is that the image of the Hankel matrix $\mathscr{H}_{i,j}(w_{\mathrm{d}})$, with $j > \mathtt{q}i$, constructed from the data w_{d} is included in the restriction $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})|_i$ of the most powerful unfalsified model on the interval $[1,i]$, *i.e.*,

$$\operatorname{span}\big(\mathscr{H}_i(w_{\mathrm{d}})\big) \subseteq \mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})|_i. \qquad (*)$$

The question occurs: under what conditions on the data w_{d} and the data, $(*)$ holds with equality? This question is important because if equality holds, *any* i-samples long trajectories w of $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$ can be constructed as a linear combination of the columns of the Hankel matrix, *i.e.*, there is g such that

$$w = \mathscr{H}_i(w_{\mathrm{d}})g.$$

In particular, we can use this equation to compute the first i samples of the impulse response. As in the realization problem, in what follows, the default input/output partitioning $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ is assumed.

It turns out that a necessary condition for equality in $(*)$ is persistency of excitation of the input component u_{d} of the given data.

Definition 3.6 (Persistency of excitation). The time series $u_{\mathrm{d}} = \big(u_{\mathrm{d}}(1),\ldots,u_{\mathrm{d}}(T)\big)$ is persistently exciting of order L if the Hankel matrix $\mathscr{H}_L(u_{\mathrm{d}})$ is of full row rank.

Lemma 3.7 (Fundamental lemma (Willems et al, 2005)). *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.,

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

2. the system $\mathscr{B}$ be controllable; and
3. the input sequence u_d be persistently exciting of order $L+\mathtt{n}(\mathscr{B})$.

Then any L samples long trajectory $w=\left[\begin{smallmatrix}u\\y\end{smallmatrix}\right]$ 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., *$\mathit{span}\big(\mathscr{H}_L(w_\mathrm{d})\big)=\mathscr{B}|_{[1,L]}$.*

Let e_k be the kth column of the $\mathtt{m}\times\mathtt{m}$ identity matrix and δ be the unit pulse function. The impulse response is a matrix valued trajectory, the columns of which are the $\mathtt{m}$ trajectories corresponding to zero initial conditions and inputs $e_1\delta$, …, $e_\mathtt{m}\delta$. Therefore, the problem of computing the impulse response is reduced to the problem of finding a vector g_k, such that $\mathscr{H}_i(w_\mathrm{d})g_k$ is of the form $(e_k\delta,h_k)$, where $h_k(\tau)=0$, for all $\tau<0$. The identically zero input/output trajectory for negative time (in the past) implies that the response from time zero on (in the future) is the impulse response.

Let ℓ be the lag of the most powerful unfalsified model $\mathscr{B}_\mathrm{mpum}(w_\mathrm{d})$. In order to describe the construction of a vector g_k that achieves the impulse response h_k, define the "past" Hankel matrix

$$\mathbf{H}_\mathrm{p}:=\mathscr{H}_{\ell,j}(w_\mathrm{d}),\quad \text{where } j:=T-(\ell+i)$$

and the "future" input and output Hankel matrices

$$\mathbf{H}_{\mathrm{f},u}:=\mathscr{H}_{i,j}(\sigma^\ell u_\mathrm{d})\quad\text{and}\quad \mathbf{H}_{\mathrm{f},y}:=\mathscr{H}_{i,j}(\sigma^\ell y_\mathrm{d}).$$

77 ⟨*define* $\mathbf{H}_\mathrm{p}$, $\mathbf{H}_{\mathrm{f},u}$, *and* $\mathbf{H}_{\mathrm{f},y}$ 77⟩≡ (78b)
```
j = T - (l + i);
Hp  = blkhank(w, l, j);
Hfu = blkhank(u(:, (l + 1):end), i, j);
Hfy = blkhank(y(:, (l + 1):end), i, j);
```
Uses blkhank 26a.

With these definitions, a vector g_k that achieves the impulse response h_k must satisfy the system of linear equations

$$\begin{bmatrix}\mathbf{H}_\mathrm{p}\\ \mathbf{H}_{\mathrm{f},u}\end{bmatrix} g_k=\begin{bmatrix}0_{\mathtt{q}\ell\times 1}\\ \begin{bmatrix}e_k\\ 0_{(t-1)\mathtt{m}\times 1}\end{bmatrix}\end{bmatrix}. \qquad (*)$$

By construction, any solution g_k of (∗) is such that

$$\mathbf{H}_{\mathrm{f},y}g_k=h_k.$$

Choosing the least norm solution as a particular solution and using matrix notation, the first i samples of the impulse response are given by

$$H=\mathbf{H}_{\mathrm{f},y}\begin{bmatrix}\mathbf{H}_\mathrm{p}\\ \mathbf{H}_{\mathrm{f},u}\end{bmatrix}^+\begin{bmatrix}0_{\mathtt{q}\ell\times\mathtt{m}}\\ \begin{bmatrix}I_\mathtt{m}\\ 0_{(i-1)\mathtt{m}\times\mathtt{m}}\end{bmatrix}\end{bmatrix},$$

where A^+ is the pseudo-inverse of A.

78a ⟨*data driven computation of the impulse response* 78a⟩≡ (78b)
```
wini_uf = [zeros(l * q, m); eye(m); zeros((i - 1) * m, m)];
h_ = Hfy * pinv([Hp; Hfu]) * wini_uf;
```
We have the following function for computation of the impulse response of the most powerful unfalsified model from data:

78b ⟨$w\mapsto H$ 78b⟩≡
```
function h = w2h(w, m, n, i)
⟨reshape w and define q, T 27a⟩, p = q - m; l = ceil(n / p);
u = w(1:m, :); y = w((m + 1):q, :);
⟨define Hp, Hf,u, and Hf,y 77⟩
⟨data driven computation of the impulse response 78a⟩
for ii = 1:i
  h(:, :, ii) = h_(((ii - 1) * p + 1):(ii * p), :);
end
```
Defines:
w2h, used in chunk 78e.

As in the case of the realization problem, when the data is noisy or generated by a high order model (higher than the specified order n), w2h computes a (suboptimal) approximation.

Using the functions w2h and h2ss, we obtain a method for exact identification ($w_\mathrm{d}\mapsto H\mapsto\widehat{\mathscr{B}}$)

78c ⟨*Most powerful unfalsified model in* $\mathscr{L}^{\mathtt{q},\mathtt{n}}_\mathtt{m}$ 78c⟩≡ 78d▷
```
function sys = w2h2ss(w, m, n, Th, i, j)
⟨reshape w and define q, T 27a⟩, p = q - m;
```
Defines:
w2h2ss, used in chunks 145–47.

The optional parameter Th specifies the number of samples of the impulse response, to be computed on the first step $w_\mathrm{d}\mapsto H$. The default value is the minimum number (T_min), defined on page 73.

78d ⟨*Most powerful unfalsified model in* $\mathscr{L}^{\mathtt{q},\mathtt{n}}_\mathtt{m}$ 78c⟩+≡ ◁78c 78e▷
```
if ~exist('Th') | isempty(Th)
  Th = ceil(n / p) + ceil(n / m) + 1;
end
```
The dimensions i and j of the Hankel matrix $\mathscr{H}_{i,j}(\sigma H)$, to be used at the realization step $w_\mathrm{d}\mapsto H$, are also optional input parameters. With exact data, their values are irrelevant as long as the Hankel matrix has the minimum number n of rows and columns. However, with noisy data, the values of Th, i, and j affect the approximation. Empirical results suggest that the default choice (i,j) gives best approximation.

78e ⟨*Most powerful unfalsified model in* $\mathscr{L}^{\mathtt{q},\mathtt{n}}_\mathtt{m}$ 78c⟩+≡ ◁78d
```
T = Th; ⟨dimension of the Hankel matrix 73a⟩
sys = h2ss(w2h(w, m, n, Th), n, [], i, j);
```
Uses h2ss 74c and w2h 78b.

Note 3.8 (Using prior knowledge). In case of inexact data, the accuracy of the impulse response computation can be improved by using prior knowledge about the model, *e.g.*, steady-state gain, settling time, and dominant time constant. Such prior

knowledge can be expressed as linear equality and inequality constraints $EH = F$ and $E'H \leq F'$. Then, the computation of H taking into account the prior knowledge leads to a convex quadratic programming problem. For the solution of this problem there are fast and efficient methods (Gill et al, 1999). In the case of prior knowledge expressed as equality constraints only, the problem is a constrained least squares. This problem admits a closed form solution (Markovsky and Mercère, 2017).

3.3 Stochastic system identification

This section considers ARMAX system representations and identification problems. Identifiability conditions in terms of the correlation function are given. One of the conditions is persistency of excitation of an input component of the process and another one is a rank condition for a pair of block-Hankel matrices.

First, we study the linear combinations of the process and its shifts that produce a process independent of the input. The set of all such linear combinations, called the orthogonalizers, has a module structure. Under identifiability conditions, it completely specifies the deterministic part of the ARMAX system. Computing a module basis for the orthogonalizers is a deterministic identification problem.

The proposed ARMAX identification method has three steps: 1) compute the deterministic part of the system via the orthogonalizers, 2) the AR part, which also has a module structure, and 3) the MA part. The method is developed first for the case of infinite data, in which case the ARMAX system is identified exactly. In the practical case of finite data the identification method necessarily involves approximation.

3.3.1 ARMAX representability, identifiability, and estimation

Given an observed infinite realization w_d of a process w, or equivalently given its correlation function R_{ww}, the following questions arise.

Representability: Is there an ARMAX system $\mathscr{B}$, such that $w \in \mathscr{B}$?
Identifiability: If w is representable, is a system $\mathscr{B}$, such that $w \in \mathscr{B}$, unique?
Identification: If w is identifiable, obtain a representation of $\mathscr{B}$ from w_d.

In a more practical situation when the given realization is finite

$$w_\mathrm{d} = \big(w_\mathrm{d}(1), \ldots, w_\mathrm{d}(T)\big),$$

the *exact* stochastic identification problem, stated above, becomes an *approximate* stochastic identification problem (see Figure 1.2). Assuming that w is identifiable, the following estimation problem and related question are of interest.

Estimation: Assuming that the system $\mathscr{B}$ is identifiable from the data w_d, find a representation of an estimate $\widehat{\mathscr{B}}$ of $\mathscr{B}$.
Consistency: Study the asymptotic behavior of the estimate $\widehat{\mathscr{B}}$, as $T \to \infty$.

Recall from Section 2.3 that an ARMAX system admits a representation by a stochastic difference equation

$$A(\sigma)\big(P(\sigma)y + Q(\sigma)u\big) = M(\sigma)\varepsilon, \tag{ARMAX}$$

with parameters the polynomial matrices A, P, Q, M and with ε white normalized process that is independent of u. We approach the ARMAX identification and estimation problems by first computing the deterministic (X-part) of the ARMAX system, then the AR-part, and finally the MA-part, *i.e.*, we consider algorithms that decompose the original problem

$$w_\mathrm{d} \xrightarrow{\text{ARMAX system identification}} (P, Q, A, M)$$

into three sequential subproblems as follows:

$$w_\mathrm{d} = \begin{bmatrix} u_\mathrm{d} \\ y_\mathrm{d} \end{bmatrix} \xrightarrow{\text{find the X-part}} (P, Q) \xrightarrow{\text{find the AR-part}} A \xrightarrow{\text{find the MA-part}} M.$$

First, we consider the case when w_d is infinite and then propose a modification for the more difficult estimation problem when w_d is finite.

The basic idea is that the finite linear combinations of the rows of the Hankel matrix composed of w_d

$$W := \begin{bmatrix} w_\mathrm{d}(1) & w_\mathrm{d}(2) & w_\mathrm{d}(3) & \cdots & w_\mathrm{d}(t) & \cdots \\ w_\mathrm{d}(2) & w_\mathrm{d}(3) & w_\mathrm{d}(4) & \cdots & w_\mathrm{d}(t+1) & \cdots \\ w_\mathrm{d}(3) & w_\mathrm{d}(4) & w_\mathrm{d}(5) & \cdots & w_\mathrm{d}(t+2) & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \end{bmatrix} \tag{3.1}$$

that are orthogonal to the rows of the Hankel matrix composed of inputs u_d

$$U := \begin{bmatrix} u_\mathrm{d}(1) & u_\mathrm{d}(2) & u_\mathrm{d}(3) & \cdots & u_\mathrm{d}(t) & \cdots \\ u_\mathrm{d}(2) & u_\mathrm{d}(3) & u_\mathrm{d}(4) & \cdots & u_\mathrm{d}(t+1) & \cdots \\ u_\mathrm{d}(3) & u_\mathrm{d}(4) & u_\mathrm{d}(5) & \cdots & u_\mathrm{d}(t+2) & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \end{bmatrix} \tag{3.2}$$

determine $R = \begin{bmatrix} P & Q \end{bmatrix}$. However, there are infinite number of such "orthogonalizing" linear combinations. The questions occur: What structure do they have in order to be generated by a finite number of them—the rows of R? What algorithms can be used for computing an estimate of R from a finite realization w_d?

3.3.2 Conditions for representability and identifiability

In this section, we study the following problems.

Problem 3.9 (ARMAX representability). Given a process w with a correlation R_{ww}, determine under what conditions $w \in \mathscr{B}$, where $\mathscr{B}$ is an ARMAX system.

The ARMAX representability problem is to find under what conditions $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ is a solution of (ARMAX) with ε a white normalized process, $\varepsilon \perp u$, and A Schur.

Problem 3.10 (ARMAX identifiability). Given a process w with a correlation function R_{ww}, determine under what conditions on the data and the data generating system, an ARMAX representation $\mathscr{B}$ of w is unique.

Define the partitioning of of the correlation function as

$$R_{ww} = \begin{bmatrix} \mathtt{m} & \mathtt{p} \\ R_{uu} & R_{uy} \\ R_{yu} & R_{yy} \end{bmatrix} \begin{matrix} \mathtt{m} \\ \mathtt{p} \end{matrix}.$$

A key condition for ARMAX representability is the notion of a *rank increment* of a Hankel matrix composed of R_{uu} with respect to a Hankel matrix composed of R_{yu}.

Definition 3.11 (Rank increment). The rank increment of A with respect to B, where A and B have the same number of columns, is

$$\operatorname{rank}\operatorname{inc}(A,B) := \operatorname{rank}\big(\operatorname{col}(A,B)\big) - \operatorname{rank}(B).$$

The notion of rank increment is well defined for infinite matrices as well. Let $A_{k,q}$ denotes the submatrix of A formed of the first k block rows and the first q block columns. The rank increment of the (two sided) infinite matrix A with respect to the (two sided) infinite matrix B is $\lim_{k,q\to\infty}\big(\operatorname{rank}\big(\operatorname{col}(A_{kq},B_{kq})\big) - \operatorname{rank}(A_{kq})\big)$.

Theorem 3.12 (ARMAX representability (Markovsky et al, 2006a)). *A process $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ with a correlation function R_{ww} is ARMAX representable if and only if*

$$\operatorname{rank}\operatorname{inc}\left(\begin{bmatrix} R_{uu}(1) & R_{uu}(2) & \cdots \\ R_{uu}(2) & R_{uu}(3) & \cdots \\ \vdots & \vdots & \end{bmatrix}, \begin{bmatrix} R_{yu}(1) & R_{yu}(2) & \cdots \\ R_{yu}(2) & R_{yu}(3) & \cdots \\ \vdots & \vdots & \end{bmatrix}\right) < \infty.$$

For identifiability we need, in addition, the assumption that the input u is persistently exciting of a sufficiently high order.

Theorem 3.13 (ARMAX identifiability (Markovsky et al, 2006a)). *A process $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ with a correlation function R_{ww} is ARMAX identifiable if the condition of Theorem 3.12 holds and if, in addition, u is persistently exciting of any order.*

As in the deterministic case, see Lemma 3.7 and (Willems et al, 2005), if an upper bounds $\mathtt{n}$ and ℓ for the order and the lag of the deterministic part of the ARMAX system are a priori given, the persistency of excitation assumption of Theorem 3.13 can be relaxed to "u is persistently exciting of order $\mathtt{n}+\ell$".

3.3.3 The module of orthogonalizers

The set $\mathbb{R}[z]^n$ of n-dimensional vector polynomials with real coefficients in the indeterminate z has the structure of a module over the ring $\mathbb{R}[z]$. A submodule of $\mathbb{R}[z]^n$ is a subset of $\mathbb{R}[z]^n$ that is also a module, *e.g.*, the submodule $p_1v_1 + \cdots + p_kv_k$ generated by $v_1,\ldots,v_k \in \mathbb{R}[z]^n$, where $v_1,\ldots,v_k \in \mathbb{R}[z]$. Every submodule of $\mathbb{R}[z]^n$ is of this (finitely generated) form. The minimal number of generators is by definition the dimension of the module. A submodule is called *slim* if it does not contain another submodule of the same dimension. This is equivalent to the set of generators $V := \begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix}$ of the submodule being right prime.

The importance of modules in systems theory stems from the fact that submodules are in a one-to-one relation with linear time-invariant systems and slim submodules are in a one-to-one relation with controllable linear time-invariant systems.

Lemma 3.14. *The left kernel of a Hankel matrix $\mathscr{H}(w)$ has a module structure.*

Proof. Consider the matrix W in (3.1) and let $\mathscr{A}$ denote its left kernel. We need to show that if $u,v \in \mathscr{A}$ then

1. $u+v \in \mathscr{A}$, and
2. $\alpha u \in \mathscr{A}$, for all $\alpha \in \mathbb{R}[z]$.

Item 1 is trivial and item 2 follows from $\sigma^t u \in \mathscr{A}$, for all $t \in \mathbb{Z}_+$, which is a simple consequence of the Hankel structure.

An element of the left kernel of W (defined in (3.1)) is called an *annihilator* of w_{d}. Consider a process w with a partition $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$. An *orthogonalizer* of w with respect to u is a polynomial $n \in \mathbb{R}^{\mathtt{q}}[z]$ that makes the process $n(\sigma)w$ independent of u.

Example 3.15. The rows of $R = \begin{bmatrix} P & Q \end{bmatrix}$ are orthogonalizers of the ARMAX process w with parameters (P,Q,A,M). To see this, note that

$$A(\sigma)\big(P(\sigma)y + Q(\sigma)u\big) = M(\sigma)\varepsilon \quad \Longrightarrow \quad P(\sigma)y + Q(\sigma)u = \sum_{t=-\infty}^{\infty} H(t)\sigma^t\varepsilon,$$

where the elements of H are the Markov parameters of the system $A(\sigma)a = M(\sigma)\varepsilon$, viewed as a system with input ε and output a. Now, since $\varepsilon \perp u$, this implies that

$$\big(P(\sigma)y + Q(\sigma)u\big) \perp u.$$

Moreover, every element of the module generated by the rows of R is an orthogonalizer of w. As shown in the following theorem, these are all orthogonalizers.

Theorem 3.16 ((Markovsky et al, 2006a)). *Consider an identifiable ARMAX process $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ with parameters (P,Q,A,M). The module of the orthogonalizers of w with respect to u is generated by the rows of $R = \begin{bmatrix} P & Q \end{bmatrix}$.*

3.3.4 Identification algorithm: infinite-time case

Theorem 3.16 suggests an algorithm for the computation of the deterministic part of the ARMAX system, *i.e.*, an algorithm that realizes the mapping $w_\mathrm{d} \mapsto R$. We need to compute the module of the orthogonalizers of w_d. This turns out to be a deterministic identification problem. In this section, we consider the infinite-time case. In the following section, we adapt the algorithm for the finite-time case.

X-part: $w_\mathrm{d} \mapsto (P,Q)$

We aim to find a module basis for the linear combinations of W (see (3.1)) that are orthogonal to U (see (3.2)). This question is equivalent to the question of computing a module basis for the left kernel of $WU^\top$. Observe that

$$WU^\top = \begin{bmatrix} R_{wu}(0) & R_{wu}^\top(1) & R_{wu}^\top(2) & \cdots & R_{wu}^\top(t) & \cdots \\ R_{wu}(1) & R_{wu}(0) & R_{wu}^\top(1) & \cdots & R_{wu}^\top(t-1) & \cdots \\ R_{wu}(2) & R_{wu}(1) & R_{wu}(0) & \cdots & R_{wu}^\top(t-2) & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \end{bmatrix}.$$

Computing a module basis for the left kern of a Hankel or Toeplitz matrix is a deterministic identification problem, see (Markovsky et al, 2006b, Section 8.5), so that computing the orthogonalizers is a deterministic identification problem for the correlation function R_{wu}. There are many algorithms developed for solving this problem, *e.g.*, subspace algorithms based on state construction (Van Overschee and De Moor, 1996a) or an observability matrix (Verhaegen and Dewilde, 1992).

How many correlation coefficients are needed in order to compute a set of generators of the left kernel of $WU^\top$, *i.e.*, can we limit the number of rows and columns of $WU^\top$? Let $\mathtt{n}$ and ℓ be given upper bounds for the order and the lag of the ARMAX system's deterministic part. Using the result of (Willems et al, 2005), we have that if u is persistently exciting of order $\mathtt{n}+\ell$, then the left kernel of the Hankel matrix

$$\begin{bmatrix} R_{wu}(-\ell-\mathtt{n}) & \cdots & R_{wu}(0) & \cdots & R_{wu}(\mathtt{n}) \\ R_{wu}(-\ell-\mathtt{n}+1) & \cdots & R_{wu}(1) & \cdots & R_{wu}(\mathtt{n}+1) \\ \vdots & & \vdots & & \vdots \\ R_{wu}(\mathtt{n}) & \cdots & R_{wu}(\ell) & \cdots & R_{wu}(\ell+\mathtt{n}) \end{bmatrix} \tag{3.3}$$

determines all orthogonalizing polynomials of degree at most ℓ and is therefore equal to the module generated by R. Hence it determines R uniquely. Exploiting symmetry, this means that $T = \mathtt{n}+\ell+1$ correlation coefficients are sufficient.

AR-part: $\big(w_\mathrm{d}, (P,Q)\big) \mapsto A$

Once R is determined, we consider the ARMA identification problem

$$A(\sigma)a = M(\sigma)\varepsilon, \quad \text{where } a := R(\sigma)w,$$

with a realization $a_\mathrm{d} := R(\sigma)w_\mathrm{d}$. Let R_{aa} be the correlation function of a. The set of annihilators of R_{aa} is the module generated by the rows of A. Therefore, we can determine the AR-part of the ARMAX system by computing a module basis of the left kernel of the two sided infinite Hankel matrix composed of R_{aa}. As in the previous subsection, however, knowing an upper bound ℓ for the degree of A, we can consider a finite Hankel matrix

$$\begin{bmatrix} R_{aa}(1) & R_{aa}(2) & \cdots & R_{aa}(\ell+1) \\ R_{aa}(2) & R_{aa}(3) & \cdots & R_{aa}(\ell+2) \\ \vdots & \vdots & & \vdots \\ R_{aa}(\ell+1) & R_{aa}(\ell+2) & \cdots & R_{aa}(2\ell+1) \end{bmatrix}. \tag{3.4}$$

This is also a deterministic identification problem that can be solved by standard algorithms (Markovsky et al, 2006b, Chapter 8).

MA-part: $(w_\mathrm{d}, P, Q, A) \mapsto M$

Once A is determined, we consider the MA identification problem

$$m = M(\sigma)\varepsilon, \quad \text{where } m := A(\sigma)a,$$

with a realization $m_\mathrm{d} := A(\sigma)a_\mathrm{d}$. There are efficient MA system identification methods proposed in the literature, see, *e.g.*, (Stoica and Moses, 2005, Section 3.6). A summary of the complete method for ARMAX system identification in the infinite-time case is given in Algorithm 2.

Algorithm 2 ARMAX identification: infinite-time case.

Input: Time series $w_\mathrm{d} = (u_\mathrm{d}, y_\mathrm{d})$ and upper bound ℓ for the degrees of the X and AR parts.
1: Compute the first $\ell+\mathtt{n}+1$ correlation coefficients of w_d.
2: Compute a module basis $\widehat{R}$ for the left kernel of (3.3).
3: Let $a_\mathrm{d} := \widehat{R}(\sigma)w_\mathrm{d}$.
4: Compute the first $2\ell+1$ correlation coefficients of a_d.
5: Compute a module basis $\widehat{A}$ for the left kernel of (3.4).
6: Let $m_\mathrm{d} := \widehat{A}(\sigma)a_\mathrm{d}$.
7: Compute the parameter $\widehat{M}$ of an MA system for m_d.
Output: ARMAX system $\widehat{\mathscr{B}}$ defined by the parameters $(\widehat{P},\widehat{Q},\widehat{A},\widehat{M})$, where $\begin{bmatrix}\widehat{P} & \widehat{Q}\end{bmatrix} := \widehat{R}$.

3.3.5 Identification algorithm: finite-time case

In the infinite time case, under the identifiability assumption, we can recover the true data generating system exactly. Of course, this is no longer possible in the finite-time case. The question of main interest, considered in the literature, is the *consistency* property of the estimate produced by an algorithm: Does the finite-time estimate converge to the true system as the time horizon goes to infinity? For a fixed time horizon, however, the identification problem necessarily involves approximation. Estimators that achieve statistically optimal approximation are called *efficient*. Another point of view of finite-time ARMAX identification problem is the *bias–variance* decomposition of the approximation error.

In this section, we are not aiming at an optimal finite-time approximation, *i.e.*, the proposed algorithm is not efficient. We are looking instead at a heuristic adaptation of the exact (in the infinite-time case) Algorithm 2 for the finite-time case. This is similar to the application of exact deterministic subspace algorithms for approximate deterministic and/or stochastic identification problems.

A straightforward finite-time version of Algorithm 2 is obtained by replacing

- the true correlation coefficients on steps 1 and 4 by the standard biased estimates,
- a module basis of an exact left kernel on steps 2 and 5 by a module basis of an approximate left kernel, computed, *e.g.*, by the singular value decomposition,
- the exact MA model on step 7 by an approximate MA model, computed, *e.g.*, by the algorithms of (Stoica et al, 2000) or (Alkire and Vandenberghe, 2002).

The quality of the correlation estimates affects the accuracy of the parameter estimates. Which estimates $\widehat{R}_{ww}$, $\widehat{R}_{aa}$ yield optimal efficiency is an open problem.

The matrices W and U, defined for the infinite-time case in in (3.1) and (3.2), are redefined for a finite realization w_{d} as follows

$$W := \begin{bmatrix} w_{\mathrm{d}}(\mathtt{n}+1) & w_{\mathrm{d}}(\mathtt{n}+2) & \cdots & w_{\mathrm{d}}(T-\ell-\mathtt{n}) \\ w_{\mathrm{d}}(\mathtt{n}+2) & w_{\mathrm{d}}(\mathtt{n}+3) & \cdots & w_{\mathrm{d}}(T-\ell-\mathtt{n}+1) \\ \vdots & \vdots & & \vdots \\ w_{\mathrm{d}}(\mathtt{n}+\ell+1) & w_{\mathrm{d}}(\mathtt{n}+\ell+2) & \cdots & w_{\mathrm{d}}(T-\ell+1) \end{bmatrix}$$

and

$$U := \begin{bmatrix} u_{\mathrm{d}}(1) & u_{\mathrm{d}}(2) & \cdots & u_{\mathrm{d}}(T-\ell-2\mathtt{n}) \\ u_{\mathrm{d}}(2) & u_{\mathrm{d}}(3) & \cdots & \\ \vdots & \vdots & & \vdots \\ u_{\mathrm{d}}(\ell+2\mathtt{n}+1) & u_{\mathrm{d}}(\ell+2\mathtt{n}+2) & \cdots & u_{\mathrm{d}}(T) \end{bmatrix},$$

Our experience is that on step 2 it is better to compute the left kernel of the matrix $WU^\top$ instead of the Hankel matrix composed of the standard biased correlation estimates $\widehat{R}_{wu}$. A possible explanation for the superior results obtained from $WU^\top$ is that the standard biased estimator ($\widehat{R}_{ww}$) implicitly extends w_{d} with zeros, which changes the data. For small sample size, the change of the data due to the extension gives inferior estimates compared to the computation from $WU^\top$, where the data is

not extended. For example, in deterministic identification problem, *i.e.*, $\mathrm{var}(\varepsilon) = 0$, the left kernel of $WU^\top$ gives exact result, while the approach using ($\widehat{R}_{ww}$) gives biased result. The same observation is made in (Stoica and Moses, 2005, pages 98–99) and is supported by a statistical argument (bias vs variance trade-off).

Similarly on step 5, we replace the Hankel matrix formed from the standard biased correlation estimates $\widehat{R}_{aa}$ (see (3.4)) by the matrix obtained from the product

$$\begin{bmatrix} a_{\mathrm{d}}(\ell+1) & a_{\mathrm{d}}(\ell+2) & \cdots & a_{\mathrm{d}}(T-\ell-1) \\ \vdots & \vdots & & \vdots \\ a_{\mathrm{d}}(2) & a_{\mathrm{d}}(3) & \cdots & a_{\mathrm{d}}(T-2\ell+2) \\ a_{\mathrm{d}}(1) & a_{\mathrm{d}}(2) & \cdots & a_{\mathrm{d}}(T-2\ell+1) \end{bmatrix} \begin{bmatrix} a_{\mathrm{d}}(\ell+2) & a_{\mathrm{d}}(\ell+3) & \cdots & a_{\mathrm{d}}(2\ell+2) \\ a_{\mathrm{d}}(\ell+3) & a_{\mathrm{d}}(\ell+4) & \cdots & a_{\mathrm{d}}(2\ell+3) \\ \vdots & \vdots & & \vdots \\ a_{\mathrm{d}}(T-\ell) & a_{\mathrm{d}}(T-\ell+1) & \cdots & a_{\mathrm{d}}(T) \end{bmatrix}, \tag{3.5}$$

which corresponds for estimation of R_{aa} without extending the data with zeros.

Algorithm 3 ARMAX identification: finite-time case.

Input: Time series $w_{\mathrm{d}} = (u_{\mathrm{d}}, \mathrm{y}_{\mathrm{d}})$, upper bound ℓ for the degrees of the X and AR parts.

1: Compute a module basis $\widehat{R}$ for an approximate left kernel of $WU^\top$, using the SVD.
2: Let $a_{\mathrm{d}} := \widehat{R}(\sigma) w_{\mathrm{d}}$.
3: Compute a module basis $\widehat{A}$ for the approximate left kernel of (3.5), using the SVD.
4: Let $m_{\mathrm{d}} := \widehat{A}(\sigma) a_{\mathrm{d}}$.
5: Compute the parameter $\widehat{M}$ of an approximate MA system for m_{d}, using the method of (Stoica et al, 2000).

Output: ARMAX system $\widehat{\mathscr{B}}$ defined by the parameters $(\widehat{P}, \widehat{Q}, \widehat{A}, \widehat{M})$, where $\begin{bmatrix} \widehat{P} & \widehat{Q} \end{bmatrix} := \widehat{R}$.

Proposition 3.17 (Consistency (Markovsky et al, 2006a)) *If $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{q}})^T$ is a realization of an identifiable ARMAX process $w \in \mathscr{B}$, where the deterministic and autoregressive parts of $\mathscr{B}$ have lags less than or equal to ℓ, Algorithm 3 with inputs w_{d} and ℓ yields a consistent estimator for the true data generating system $\mathscr{B}$.*

3.4 Missing data recovery

This section studies the problem of computing an exact linear time-invariant model from data with missing values. We do not make assumptions about the nature or pattern of the missing values apart from the basic one that they are a part of a trajectory of a linear time-invariant system. Identification with missing data is equivalent to Hankel structured low-rank matrix completion. Two methods for solving the matrix completion problem are presented. The first one is based on the nuclear norm relaxation and leads to the solution of a convex optimization problem. The second method generalizes subspace identification methods to missing data. As a matrix completion problem, the nuclear norm heuristic is more general but less efficient (because it does not exploit the Hankel structure) than the subspace method.

3.4.1 Problem formulation

The problem considered is defined informally as follows: complete a given sequence without knowledge of the data generating system, apart from the fact that it is linear time-invariant and has bounded lag. In order to state the problem formally, next, we introduce notation for missing values in a time series and vector/matrix indexing.

Missing data values are denoted by the symbol `NaN` ("not a number"). The extended set of real numbers $\mathbb{R}_e$ is the union of the set of the real numbers $\mathbb{R}$ and `NaN`:

$$\mathbb{R}_e := \mathbb{R} \cup \texttt{NaN}.$$

The data for the considered problem is a $\mathtt{q}$-variate sequence

$$w_d = \big(w_d(1), \dots, w_d(T)\big), \quad \text{where } w_d(t) \in \mathbb{R}_e^{\mathtt{q}}.$$

The data sequence w_d is parameterized by the $T\mathtt{q} \times 1$-dimensional vector

$$\operatorname{vec}(w_d) := \operatorname{col}\big(w_d(1), \dots, w_d(T)\big).$$

The subvector of a vector a with indexes in the set $\mathscr{I}$ is denoted by $a_{\mathscr{I}}$. Similarly, for a matrix A, $A_{\mathscr{I}}$ is the submatrix formed by the rows with indexes in the set $\mathscr{I}$, and $A_{\mathscr{I},\mathscr{J}}$ is the submatrix of A with elements a_{ij}, such that $i \in \mathscr{I}$ and $j \in \mathscr{J}$. The set of integers from i to j is denoted by $i{:}j$. The set of the indexes of the missing elements of the data vector $\operatorname{vec}(w_d)$ is denoted by $\mathscr{I}_m$ and the set of the indexes of the given elements is denoted by $\mathscr{I}_g$. The set indexing notation is used to select a complete submatrix/subvector of a matrix/vector with missing values. A submatrix/subvector is called *complete* if it has no missing values.

Problem 3.18 (Exact system identification from trajectory with missing values). Given a sequence with missing values $w_d \in (\mathbb{R}_e^{\mathtt{q}})^T$ and a natural number $\mathtt{m} < \mathtt{q}$,

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{\mathscr{B}} \in \mathscr{L}_{\mathtt{m}} \text{ and } \widehat{w} \in (\mathbb{R}^{\mathtt{q}})^T \quad \ell(\widehat{\mathscr{B}}) \\ &\text{subject to} \quad \widehat{w}_{\mathscr{I}_g} = w_{d,\mathscr{I}_g} \quad \text{and} \quad \widehat{w} \in \widehat{\mathscr{B}}|_T. \end{aligned} \tag{MPUM}$$

Without missing values, (MPUM) coincides with the definition of of the most powerful unfalsified model $\mathscr{B}_{\mathrm{mpum}}(w_d)$ for the data w_d. With missing values, the constraint $\widehat{w} \in \widehat{\mathscr{B}}$ requires completion of the missing values. If a completion exists for a given model $\widehat{\mathscr{B}}$, then this model is a feasible solution of problem (MPUM). The aim is to find the least complicated feasible model. This is a generalization of the notion of the most powerful unfalsified model for data with missing values.

3.4.2 Nuclear norm heuristic for missing data recovery

The method presented in this section computes a completion $\widehat{w}$ of the given trajectory with missing elements w_d. Once $\widehat{w}$ is computed, the problem of finding $\mathrm{mpum}(w_d)$ reduces to the problem of finding $\mathrm{mpum}(\widehat{w})$, *i.e.*, finding the MPUM *without* missing values. The latter is a well studied problem, for which many solutions exist (see, the notes and references section). In this sense, the completion method in this section solves the system identification problem with missing data.

Link to rank deficient Hankel matrices

Both the nuclear norm minimization method and the subspace method of the following section are based on a representation of a trajectory w of a linear time-invariant systems with bounded complexity as a rank deficiency of a Hankel matrix $\mathscr{H}_L(w)$.

Lemma 3.19. *There is $\mathscr{B} \in \mathscr{L}_{\mathtt{m},\ell}$, such that $w \in \mathscr{B}$ if and only if*

$$\operatorname{rank}\big(\mathscr{H}_L(w)\big) \leq \mathtt{m}L + \mathtt{p}\ell, \quad \textit{for any } L \geq \ell + 1.$$

Using Lemma 3.19, we rewrite Problem 3.18 as a matrix completion problem:

$$\text{minimize} \quad \text{over } \widehat{w} \in (\mathbb{R}^{\mathtt{q}})^T \quad \operatorname{rank}\big(\mathscr{H}_L(\widehat{w})\big) \quad \text{subject to} \quad \widehat{w}_{\mathscr{I}_g} = w_{d,\mathscr{I}_g}. \tag{HMC}$$

Due to the rank constraint, (HMC) is a nonconvex problem. A convex relaxation approach (Fazel, 2002; Liu and Vandenberghe, 2009), called the nuclear norm heuristic, is to replace the rank by the nuclear norm $\|\cdot\|_*$ (sum of the singular values):

$$\text{minimize} \quad \text{over } \widehat{w} \in (\mathbb{R}^{\mathtt{q}})^T \quad \big\|\mathscr{H}_L(\widehat{w})\big\|_* \quad \text{subject to} \quad \widehat{w}_{\mathscr{I}_g} = w_{d,\mathscr{I}_g}. \tag{NN}$$

Problem (NN) is shown to be the tightest convex relaxation of the rank constraint minimization problem (HMC). Its advantage over alternative heuristic methods is the guaranteed convergence to a global minimum. In (Usevich and Comon, 2015), it is shown that certain (HMC) problems can be solved exactly by the nuclear norm minimization heuristic (NN), however, at present there are no general results about the relation of the solutions of (HMC) and (NN).

Problem (NN) is equivalent to the following semidefinite programming problem

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{w}, U, \text{ and } V \quad \operatorname{trace}(U) + \operatorname{trace}(V) \\ &\text{subject to} \quad \begin{bmatrix} U & \mathscr{H}_L(\widehat{w})^\top \\ \mathscr{H}_L(\widehat{w}) & V \end{bmatrix} \succeq 0 \quad \text{and} \quad \widehat{w}_{\mathscr{I}_g} = w_{d,\mathscr{I}_g}. \end{aligned}$$

It can be solved globally and efficiently by existing methods and software (Boyd and Vandenberghe, 2001). Using general purpose semidefinite programming solvers, however, the computational cost is at least cubic in the length T of the sequence.

Efficient solvers that exploit the Hankel structure as well as efficient first order optimization methods are developed in (Fazel et al, 2013; Liu and Vandenberghe, 2009).

Example 3.20 (Completing the sequence of the Fibonacci numbers). Consider the sequence of the first 8 Fibonacci numbers with missing 3rd and 6th element:

$$w_{\mathrm{d}} = (1, 1, \texttt{NaN}, 3, 5, \texttt{NaN}, 13, 21)\,. \tag{w_{d}}$$

The complete sequence is $\bar{w} \in \bar{\mathscr{B}}|_8 \in \mathscr{L}_{0,\ell}$, where

$$\bar{\mathscr{B}} = \ker\big(\bar{R}(\sigma)\big), \quad \text{where } \bar{R}(z) = 1 + z - z^2.$$

Therefore, the Hankel matrix $\mathscr{H}_L(\bar{w})$ has rank at most 2. The nuclear norm minimization problem (NN) for the example is

$$\text{minimize} \quad \text{over } \widehat{w}(3) \text{ and } \widehat{w}(6) \quad \left\| \begin{bmatrix} 1 & 1 & \widehat{w}(3) & 3 & 5 \\ 1 & \widehat{w}(3) & 3 & 5 & \widehat{w}(6) \\ \widehat{w}(3) & 3 & 5 & \widehat{w}(6) & 13 \\ 3 & 5 & \widehat{w}(6) & 13 & 21 \end{bmatrix} \right\|_*.$$

We solve it with CVX (Grant and Boyd, 2017):

89 ⟨*Completing the sequence of the Fibonacci numbers using nuclear norm optimization* 89⟩≡

```
cvx_begin sdp;
  variables wh3 wh6;
  minimize(norm_nuc([ 1   1   wh3 3   5   ;
                      1   wh3 3   5   wh6 ;
                      wh3 3   5   wh6 13  ;
                      3   5   wh6 13  21  ]));
cvx_end
```

The solution $\widehat{w}(3) = 2$ and $\widehat{w}(6) = 8$ is found by the SDPT3 solver in 10 iterations.

Example 3.21 (Lightly damped mechanical system). As a realistic engineering example, consider a lightly damped autonomous mechanical system, with one output and poles at

$$z_{1,2} = 0.8889 \pm i0.4402, \quad z_{3,4} = 0.4500 \pm i0.8801, \quad z_{5,6} = 0.6368 \pm i0.7673.$$

The identification data w_{d} is a $T = 200$ samples long trajectory, of which 86 samples are missing in a periodic pattern. Using SDPT3 to solve the nuclear norm minimization problem (NN) for this data takes 96 seconds on a computer with 2.60GHz Intel i7 processor and 8G RAM. The solution is correct up to the default convergence tolerance of the optimization method.

3.4.3 Subspace method for identification with missing data

The method presented in this section is based on ideas from subspace identification. It derives a nonminimal kernel representation from the data with missing values and

reduces it to a minimal one. The derivation of the nonminimal representation is done by finding completely specified submatrices of the incomplete Hankel matrix of the data and computing their kernels. The resulting algorithm uses only standard linear algebra operations and does not require iterative nonlinear optimization. First, we illustrate the method on two examples. Then, we present the general algorithm.

Example: autonomous system

Consider again the example of completing the sequence of the Fibonacci numbers (w_{d}). By grouping together the columns of the incomplete Hankel matrix

$$\mathscr{H}_4(w_{\mathrm{d}}) = \begin{bmatrix} 1 & 1 & \texttt{NaN} & 3 & 5 \\ 1 & \texttt{NaN} & 3 & 5 & \texttt{NaN} \\ \texttt{NaN} & 3 & 5 & \texttt{NaN} & 13 \\ 3 & 5 & \texttt{NaN} & 13 & 21 \end{bmatrix}$$

that have missing elements in the same rows, we obtain the following submatrices

$$\widetilde{H}^1 = \begin{bmatrix} 1 & 3 \\ 1 & 5 \\ \texttt{NaN} & \texttt{NaN} \\ 3 & 13 \end{bmatrix} \quad \text{and} \quad \widetilde{H}^2 = \begin{bmatrix} 1 & 5 \\ \texttt{NaN} & \texttt{NaN} \\ 3 & 13 \\ 5 & 21 \end{bmatrix}.$$

Skipping the missing rows, we obtain complete submatrices of $\mathscr{H}_4(w_{\mathrm{d}})$

$$H^1 = \begin{bmatrix} 1 & 3 \\ 1 & 5 \\ 3 & 13 \end{bmatrix} \quad \text{and} \quad H^2 = \begin{bmatrix} 1 & 5 \\ 3 & 13 \\ 5 & 21 \end{bmatrix}.$$

Since $\operatorname{rank}\big(\mathscr{H}_4(\bar{w})\big) = 2$ and $H^1, H^2 \in \mathbb{R}^{3\times 2}$, H^1 and H^2 have nontrivial left kernels. Indeed,

$$\begin{bmatrix} 1 & 2 & -1 \end{bmatrix} H^1 = 0 \quad \text{and} \quad \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} H^2 = 0.$$

With the convention $0 \times \texttt{NaN} = 0$, we extend the vectors in the left kernel of H^1 and H^2 to vectors in the left kernel of $\mathscr{H}_4(w_{\mathrm{d}})$ by inserting zeros for the missing values

$$\underbrace{\begin{bmatrix} 1 & 2 & 0 & -1 \\ 1 & 0 & -2 & 1 \end{bmatrix}}_{\widetilde{R}} \mathscr{H}_4(w_{\mathrm{d}}) = 0.$$

This shows that, the polynomial matrix

$$\widetilde{R}(z) = \begin{bmatrix} \widetilde{R}_1(z) \\ \widetilde{R}_2(z) \end{bmatrix} = \begin{bmatrix} 1z^0 + 2z^1 + 0z^2 - 1z^3 \\ 1z^0 + 0z^1 - 2z^2 + 1z^3 \end{bmatrix}$$

obtained from the left $\ker\left(\mathscr{H}_4(w_{\mathrm{d}})\right)$ defines a kernel representation of $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$. Indeed the greatest common divisor of $R_1(z)$ and $R_2(z)$:

$$\widehat{R}(z) := \mathrm{GCD}(1+2z-z^3, 1-2z^2+z^3) = 1+z-z^2$$

is a minimal kernel representation of the data generating system $\bar{\mathscr{B}}$.

Example: open system

Consider the sequence with periodic missing values in the first variable

$$w_{\mathrm{d}} = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} \mathtt{NaN} \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} \mathtt{NaN} \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \end{bmatrix}, \begin{bmatrix} \mathtt{NaN} \\ 5 \end{bmatrix}, \begin{bmatrix} 4 \\ 10 \end{bmatrix}, \begin{bmatrix} \mathtt{NaN} \\ 14 \end{bmatrix}, \begin{bmatrix} 5 \\ 9 \end{bmatrix} \right)$$

that is generated by the following linear time-invariant system

$$\bar{\mathscr{B}} = \{ w \mid \underbrace{\begin{bmatrix} 1 & 1 \end{bmatrix}}_{\bar{R}_0} w(t) + \underbrace{\begin{bmatrix} 0 & -1 \end{bmatrix}}_{\bar{R}_1} w(t+1) = 0, \}$$

with $\mathtt{m} = 1$ input and lag $\ell = 1$. The unknown complete sequence is

$$\bar{w} = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \end{bmatrix}, \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \begin{bmatrix} 4 \\ 10 \end{bmatrix}, \begin{bmatrix} -5 \\ 14 \end{bmatrix}, \begin{bmatrix} 5 \\ 9 \end{bmatrix} \right) \in \bar{\mathscr{B}}|_T$$

Our goal is to find $\bar{\mathscr{B}}$ from the data w_{d}.

First, we revise the classical approach of finding $\bar{\mathscr{B}}$ from a complete trajectory $\bar{w}$. The parameter vector $\bar{R} = \begin{bmatrix} \bar{R}_0 & \bar{R}_1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & -1 \end{bmatrix}$ of a kernel representation $\ker\left(\bar{R}(\sigma)\right)$ of the data generating system $\bar{\mathscr{B}}$ satisfies the equation

$$\bar{R}\mathscr{H}_2(\bar{w}) = 0.$$

Provided that the left kernel of $\mathscr{H}_2(\bar{w})$ is one dimensional (persistency of excitation assumption (Willems et al, 2005), satisfied by $\bar{w}$), the kernel parameter $\bar{R}$ can be found up to a scaling factor from $\bar{w}$ by computing a basis for the left kernel of $\mathscr{H}_2(\bar{w})$. The Hankel matrix $\mathscr{H}_2(w_{\mathrm{d}})$ of the given data

$$\mathscr{H}_2(w_{\mathrm{d}}) = \begin{bmatrix} 1 & \mathtt{NaN} & 2 & \mathtt{NaN} & 3 & \mathtt{NaN} & 4 & \mathtt{NaN} \\ 0 & 1 & 1 & 3 & 2 & 5 & 10 & 14 \\ \mathtt{NaN} & 2 & \mathtt{NaN} & 3 & \mathtt{NaN} & 4 & \mathtt{NaN} & 5 \\ 1 & 1 & 3 & 2 & 5 & 10 & 14 & 9 \end{bmatrix},$$

however, has unspecified entries in every column, so that its left kernel can not be computed. Therefore, the classical method is not applicable.

The idea behind the subspace method for exact identification with missing data is to consider the *extended Hankel matrix*

$$\mathscr{H}_3(w_{\mathrm{d}}) = \begin{bmatrix} 1 & \mathtt{NaN} & 2 & \mathtt{NaN} & 3 & \mathtt{NaN} & 4 \\ 0 & 1 & 1 & 3 & 2 & 5 & 10 \\ \mathtt{NaN} & 2 & \mathtt{NaN} & 3 & \mathtt{NaN} & 4 & \mathtt{NaN} \\ 1 & 1 & 3 & 2 & 5 & 10 & 14 \\ 2 & \mathtt{NaN} & 3 & \mathtt{NaN} & 4 & \mathtt{NaN} & 5 \\ 1 & 3 & 2 & 5 & 10 & 14 & 9 \end{bmatrix}$$

and select the two submatrices of $\mathscr{H}_3(w_{\mathrm{d}})$

$$\widetilde{H}^1 = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 10 \\ \mathtt{NaN} & \mathtt{NaN} & \mathtt{NaN} & \mathtt{NaN} \\ 1 & 3 & 5 & 14 \\ 2 & 3 & 4 & 5 \\ 1 & 2 & 10 & 9 \end{bmatrix} \quad \text{and} \quad \widetilde{H}^2 = \begin{bmatrix} \mathtt{NaN} & \mathtt{NaN} & \mathtt{NaN} \\ 1 & 3 & 5 \\ 2 & 3 & 4 \\ 1 & 2 & 10 \\ \mathtt{NaN} & \mathtt{NaN} & \mathtt{NaN} \\ 3 & 5 & 14 \end{bmatrix},$$

that have missing values in the same rows. The matrices H^1 and H^2, obtained from $\widetilde{H}_1$ and $\widetilde{H}_2$, respectively, by removing the missing rows have nontrivial left kernels

$$\underbrace{\begin{bmatrix} -1 & -1 & 1 & 0 & 0 \end{bmatrix}}_{\widetilde{R}^1} \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 10 \\ 1 & 3 & 5 & 14 \\ 2 & 3 & 4 & 5 \\ 1 & 2 & 10 & 9 \end{bmatrix} = 0 \quad \text{and} \quad \underbrace{\begin{bmatrix} 0 & -1 & -1 & 1 \end{bmatrix}}_{\widetilde{R}^2} \begin{bmatrix} 1 & 3 & 5 \\ 2 & 3 & 4 \\ 1 & 2 & 10 \\ 3 & 5 & 14 \end{bmatrix} = 0.$$

Inserting zeros in the R^1 and R^2 at the location of the missing values, we obtain vectors $\widetilde{R}^1$ and $\widetilde{R}^2$ in the left kernels of $\widetilde{H}^1$ and $\widetilde{H}^2$

$$\underbrace{\begin{bmatrix} -1 & -1 & 0 & 1 & 0 & 0 \end{bmatrix}}_{\widetilde{R}^1} \widetilde{H}^1 = 0 \quad \text{and} \quad \underbrace{\begin{bmatrix} 0 & 0 & -1 & -1 & 0 & 1 \end{bmatrix}}_{\widetilde{R}^2} \widetilde{H}^2 = 0.$$

By construction $\begin{bmatrix} \widetilde{R}^1 \\ \widetilde{R}^2 \end{bmatrix} \mathscr{H}_3(\bar{w}) = 0$, so that, the polynomial matrix

$$\widetilde{R}(z) = \begin{bmatrix} \widetilde{R}^1(z) \\ \widetilde{R}^2(z) \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -1 & -1 \end{bmatrix} z^0 + \begin{bmatrix} 0 & 1 \end{bmatrix} z^1 + \begin{bmatrix} 0 & 0 \end{bmatrix} z^2 \\ \begin{bmatrix} 0 & 0 \end{bmatrix} z^0 + \begin{bmatrix} -1 & -1 \end{bmatrix} z^1 + \begin{bmatrix} 0 & 1 \end{bmatrix} z^2 \end{bmatrix}$$

is a (nonminimal) kernel representation of $\widehat{\mathscr{B}}$. In the example, $\widetilde{R}^2(z) = z\widetilde{R}^1(z)$, so that a minimal kernel representation is

$$\widehat{R}(z) = \begin{bmatrix} 1 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -1 \end{bmatrix} z = \bar{R}(z).$$

The subspace algorithm

The method illustrated in the examples leads to Algorithm 4. The selection of complete submatrices H^i of the Hankel matrix constructed from the data w_{d} is specified in Algorithm 5. The number k of the submatrices selected by the algorithm depends in a complicated way on the number and the pattern of the missing elements of w_{d}. For recovery of $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$, the total number of annihilators $g := g_1 + \cdots + g_k$ computed, *i.e.*, the sum of the dimensions of the left kernels of the submatrices H^i is of interest. A necessary condition for recovery of an MPUM with lag ℓ is $g \geq \mathtt{p}\sum_{i=1}^{k}(L_i - \ell)$, *i.e.*, existence of a sufficient number of annihilators to specify a kernel representation of $\mathscr{B}_{\mathrm{mpum}}(w_{\mathrm{d}})$.

Algorithm 4 Subspace algorithm for linear time-invariant system identification with missing data.

Input: A sequence $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{q}}_{\mathrm{e}})^T$ and natural numbers $\mathtt{m}$.

1: Let $\mathscr{H}_T(w_{\mathrm{ext}})$ be the Hankel matrix of the extended data sequence with T `NaN`'s

$$w_{\mathrm{ext}} = (w_{\mathrm{d}}, \underbrace{\mathtt{NaN}, \ldots, \mathtt{NaN}}_{T}).$$

2: Using Algorithm 5, select real valued submatrices $H^1, \ldots, H^k$ of $\mathscr{H}_T(w_{\mathrm{ext}})$.
3: **for** $i = 1 : k$ **do**
4: Compute bases R^i for the left kernels of H^i, *i.e.*, full row rank matrices $R^i \in \mathbb{R}^{g_i \times m_i}$ of maximum row dimension g_i, such that $R^i H^i = 0$.
5: Extend $R^i \in \mathbb{R}^{g_i \times m_i}$ to $\widetilde{R}^i \in \mathbb{R}^{g_i \times Tq}$ by inserting zero columns at the location of the rows removed from $\mathscr{H}_T(w_{\mathrm{ext}})$ in the selection of H^i.
6: **end for**
7: Compute a minimal kernel representation $\widehat{R}(z)$ for the system $\widehat{\mathscr{B}} := \ker\left(\begin{bmatrix} \widetilde{R}^1 \\ \vdots \\ \widetilde{R}^k \end{bmatrix}(\sigma)\right)$.
8: Convert $\widehat{R}(z)$ to an input/state/otuput representation and compute the completion $\widehat{w}$ of w_{d} (see Exercise 6.1).

Output: Minimal kernel representation $\widehat{R}(z)$ of the data generating system and $\widehat{w}$.

Example 3.22 (Lightly damped mechanical system). Solving the identification problem of Example 3.21 on the same computer, the exact system is recovered up to numerical precision errors, in 0.27 seconds. This result demonstrates the lower computation cost of Algorithm 4 in comparison to the nuclear norm minimization method.

Algorithm 5 Selection of complete submatrices of $\mathscr{H}_T(w_{\mathrm{ext}})$.

Input: Hankel matrix H.

1: Let $k := 0$.
2: **for** $L_k = 1 : T/2$ **do**
3: **for** $j = 1 : T - L_k$ **do**
4: **if** the missing values pattern of $H(1 : L_k, j)$ has not been found before **then**
5: Let $k = k + 1$.
6: Let $H^k := H(1 : L_k, j)$. {Start a new matrix H^k with the missing values pattern of $H(1 : L_k, j)$.}
7: **else**
8: Let $i :=$ index of the matrix H^i with missing values pattern as the one of $H(1 : L_k, j)$.
9: Let $H^i := \begin{bmatrix} H^i & H(1 : L_k, j) \end{bmatrix}$. {Augment H^i with the newly found column.}
10: **end if**
11: **end for**
12: **end for**
13: Remove the submatrices $H^i \in \mathbb{R}^{m_i \times n_i}$, for which $m_i < n_i + 1$ and redefine k.

Output: Submatrices $H^1, \ldots, H^k$ of H.

3.5 Notes and references

The most powerful unfalsified model

One of the key ideas that came out from the original work Willems (1986a,b, 1987) of Jan Willems on the behavioral approach to systems and control is the notion of the most powerful unfalsified model, or MPUM for short. The MPUM is "unfalsified" in the sense that it is an exact model for the given data and "most powerful" in the sense that it is the least complicated exact model. Thus, the MPUM is an optimal exact model for the data. The notion of the MPUM plays a key role in system identification and since its introduction in the mid 80's, many methods have been developed for its numerical computation. Most of the currently used methods for computing the MPUM assume an a priori given input/output partitioning of the variables and fall into the class of the (deterministic) subspace identification methods.

Subspace identification

"Subspace methods" refer to a wide spectrum of methods, which common feature is noniterative application of classical linear algebra operations, such as the singular value decomposition and the approximate solution of a system of linear equations in the least squares sense. The singular value decomposition is a computation tool for unstructured low-rank approximation. The key idea of the subspace methods is to relax the underlying structured low-rank approximation problem by ignoring the structure. After finding the unstructured low-rank approximation, in a second step, the subspace methods estimate the model parameters by solving a least squares approximation problem. The overall two-step procedure, however, is suboptimal with respect to the original optimization problem.

Subspace identification has its origin in realization theory. The motivation for the development of the first subspace methods (Budin, 1971; Gopinath, 1969) is generalization of Kung's method. This initial work considers exact (deterministic) identification problem. The idea of finding a state sequence from a general trajectory of the system is due to (Willems, 1986b). This idea is effectively used by Van Overschee and De Moor (1996b) in what is known as the N4SID method. An alternative approach, known as the MOESP method, based on finding an observability matrix instead of a state sequence is developed at the same time by Verhaegen and Dewilde (1992). The steps of state estimation and observability matrix estimation in the N4SID and MOESP methods are special cases of data driven simulation (Markovsky et al, 2006b, Chapter 7).

The class of the subspace methods was generalized to identification

- of dissipative (Goethals et al, 2003; Rapisarda and Trentelman, 2011) systems,
- in the ARMAX (Van Overschee and De Moor, 1996b; Verhaegen and Dewilde, 1992) and errors-in-variables (Chou and Verhaegen, 1997) settings, and
- of closed-loop (Chiuso, 2006) and nonlinear (Noël and Kerschen, 2013) systems.

Subspace identification with prior knowledge

Prior knowledge about stability and passivity of the model is considered in (Goethals et al, 2003; Maciejowski, 1995). The approach used in (Goethals et al, 2003; Maciejowski, 1995) consists in including a regularization term in the least squares cost function for the estimation of the model parameters. The main result is that, for sufficiently large values of the regularization parameter, the identified model is stable and passive. More recently subspace identification with prior knowledge was considered in (Alenany et al, 2010; Trnka and Havlena, 2009). In (Alenany et al, 2010), prior knowledge about the steady-state gain of the system is taken into account by a modified PO-MOESP algorithm, where a constrained least squares problem is solved. The approach of Trnka and Havlena (2009) generalizes the regularization approach of Goethals et al (2003); Maciejowski (1995) to a Bayesian framework for including prior knowledge about the parameter vector. The method however involves a solution of a nonconvex optimization problem, which makes it comparable to the prediction error methods.

Missing data estimation

Most of the current literature on identification with missing data addresses special cases, such as particular patterns of occurrence of the missing data, or uses heuristics for estimation of the missing data, such as interpolation methods, followed by classical identification from the completed data. Three important special cases are:

- partial realization problem,
- missing input and output values in a single block, and

- missing values in the output only.

The partial realization problem, *i.e.*, finding a state space representation from the first few samples of the impulse response, can be viewed as a special missing data estimation problem. Indeed, finding a representation for the model is equivalent to completing the partial data of the impulse response. This solution based on realization theory, however, does not generalize to other patterns of missing data.

Another special identification problem with missing data considered in (Schoukens et al, 2012) is the problem when missing are in a block of $\ell(\mathscr{B})$ or more sequential values of all variables. In this case, the identification problem with missing data is equivalent to identification from two independent data sets—the samples before and the samples after the block of missing values. This result also does not generalize to other patterns of missing values.

The special case when the missing data is restricted to the output variables only can be handled by the classical prediction error identification methods (Ljung, 1999; Söderström and Stoica, 1989). The predictor is used to estimate the missing output values from the inputs, the current guess of the model, and the initial conditions.

The general identification problem with missing data can be approached by choosing a representation of the model and optimizing the complexity over the model parameters and the missing values, subject to the constraint that the completed data is a trajectory of the system. This leads to a nonconvex optimization problem. Three classes of methods that use this approach are:

- modification of the classical prediction error methods,
- methods developed in the structure low-rank approximation setting, and
- convex relaxation methods based on the nuclear norm heuristic.

All these methods are designed for estimation from noisy as well as missing data.

Exercises

3.1 (The most powerful unfalsified model in $\mathscr{L}_0$). Explain how to find $\mathscr{B}_{\mathrm{mpum}}(\mathscr{D})$ for the data $\mathscr{D} = \{d_1, \ldots, d_N\} \subset \mathbb{R}^{\mathrm{q}}$ in the model class of linear static models $\mathscr{L}_0^{\mathrm{q}}$.

3.2 (The most powerful unfalsified model in $\mathscr{L}_{0,\ell}^1$). Find conditions for existence of an exact autonomous linear time-invariant model with lag at most ℓ for a scalar given data sequence $y_{\mathrm{d}} = \big(y_{\mathrm{d}}(1), \ldots, y_{\mathrm{d}}(T)\big)$, $y_{\mathrm{d}}(t) \in \mathbb{R}$.

3.3 ($y_{\mathrm{d}} \mapsto \mathscr{B}_{\mathrm{mpum}}(y_{\mathrm{d}}) = \ker\big(p(\sigma)\big)$). Propose a method for computing the model parameter $p(z)$ of a kernel representation of the most powerful unfalsified model $\mathscr{B}_{\mathrm{mpum}}(y_{\mathrm{d}})$ in the model class $\mathscr{L}_{0,\ell}^1$. If the lag ℓ is not given, explain how to find it from the data y_{d}. Implement the methods and test it on the data

$$w_{\mathrm{d}} = (1, 2, 4, 7, 13, 24, 44, 81). \qquad (*)$$

3.4 ($y_\mathrm{d} \mapsto \mathscr{B}_\mathrm{mpum}(y_\mathrm{d}) = \mathscr{B}(A,c)$)). Propose a method for computing the model parameters (A,c) of a state space representation of the most powerful unfalsified model $\mathscr{B}_\mathrm{mpum}(y_\mathrm{d})$ in the model class $\mathscr{L}^1_{0,\ell}$. If the order $\mathtt{n} = \ell$ is not given, explain how to find it from the data y_d. Implement the method and test it on the data (∗).

3.5 (Data-driven step response estimation). Propose a method for computing the step response of $\mathscr{B}_\mathrm{mpum}(y_\mathrm{d})$ directly from the data y_d. Implement the method and test it on an example.

References

Alenany A, Shang H, Soliman M, Ziedan I (2010) Subspace identification with prior steady-state information. In: Proceedings of the International Conference on Computer Engineering and Systems, Cairo, Egypt

Alkire B, Vandenberghe L (2002) Convex optimization problems involving finite autocorrelation sequences. Mathematical Programming, Series A 93(3):331–359

Boyd S, Vandenberghe L (2001) Convex Optimization

Budin MA (1971) Minimal realization of discrete linear systems from input-output observations. IEEE Trans Automat Contr 16(5):395–401

Chiuso A (2006) Asymptotic variance of closed-loop subspace identification methods. IEEE Transactions on Automatic Control 51(8):1299–1314

Chou C, Verhaegen M (1997) Subspace algorithms for the identification of multivariate errors-in-variables models. Automatica 33(10):1857–1869

Fazel M (2002) Matrix rank minimization with applications. PhD thesis, Elec. Eng. Dept., Stanford University

Fazel M, Pong TK, Sun D, Tseng P (2013) Hankel matrix rank minimization with applications in system identification and realization. SIAM J Matrix Anal Appl 34(3):946–977

Gill PE, Murray M, Wright MH (1999) Practical Optimization. Academic Press

Goethals I, Van Gestel T, Suykens J, Van Dooren P, De Moor B (2003) Identification of positive real models in subspace identification by using regularization. IEEE Trans Automat Contr 48:1843–1847

Gopinath B (1969) On the identification of linear time-invariant systems from input-output data. The Bell System Technical J 48(5):1101–1113

Grant M, Boyd S (2017) CVX: Matlab software for disciplined convex programming. stanford.edu/~boyd/cvx

Heij C (1989) Deterministic identification of dynamical systems, Lecture notes in control and information sciences, vol 127. Springer

Liu Z, Vandenberghe L (2009) Interior-point method for nuclear norm approximation with application to system identification. SIAM J Matrix Anal Appl 31(3):1235–1256

Ljung L (1999) System Identification: Theory for the User. Prentice-Hall

Maciejowski J (1995) Guaranteed stability with subspace methods. Control Lett 26:153–156

Markovsky I, Mercère G (2017) Subspace identification with constraints on the impulse response. Int J Contr pp 1728–1735

Markovsky I, Willems JC, De Moor B (2006a) The module structure of ARMAX systems. In: Proc. of the 41st Conf. on Decision and Control, San Diego, USA, pp 811–816

Markovsky I, Willems JC, Van Huffel S, De Moor B (2006b) Exact and Approximate Modeling of Linear Systems: A Behavioral Approach. SIAM

Moonen M, De Moor B, Vandenberghe L, Vandewalle J (1989) On- and off-line identification of linear state-space models. Int J Control 49:219–232

Noël JP, Kerschen G (2013) Frequency-domain subspace identification for nonlinear mechanical systems. Mechanical Systems and Signal Processing 40(2):701–717

Rapisarda P, Trentelman H (2011) Identification and data-driven model reduction of state-space representations of lossless and dissipative systems from noise-free data. Automatica 47(8):1721–1728

Schoukens J, Vandersteen G, Rolain Y, Pintelon R (2012) Frequency response function measurements using concatenated subrecords with arbitrary length. IEEE Trans on Instr and Measurement 61(10):2682–2688

Söderström T, Stoica P (1989) System Identification. Prentice Hall

Stoica P, Moses R (2005) Spectral Analysis of Signals. Prentice Hall

Stoica P, McKelvey T, Mari J (2000) MA estimation in polynomial time. IEEE Trans Signal Proc 48(7):1999–2012

Trnka P, Havlena V (2009) Subspace like identification incorporating prior information. Automatica 45:1086–1091

Usevich K, Comon P (2015) Quasi-Hankel low-rank matrix completion: a convex relaxation. ArXiv: 1505.07766

Van Overschee P, De Moor B (1996a) Subspace Identification for Linear Systems: Theory, Implementation, Applications. Kluwer, Boston

Van Overschee P, De Moor B (1996b) Subspace identification for linear systems: Theory, implementation, applications. Kluwer, Boston

Verhaegen M, Dewilde P (1992) Subspace model identification, Part 1: The output-error state-space model identification class of algorithms. Int J Control 56:1187–1210

Viberg M (1995) Subspace-based methods for the identification of linear time-invariant systems. Automatica 31(12):1835–1851

Willems JC (1986a) From time series to linear system—Part I. Finite dimensional linear time invariant systems. Automatica 22(5):561–580

Willems JC (1986b) From time series to linear system—Part II. Exact modelling. Automatica 22(6):675–694

Willems JC (1987) From time series to linear system—Part III. Approximate modelling. Automatica 23(1):87–115

Willems JC, Rapisarda P, Markovsky I, De Moor B (2005) A note on persistency of excitation. Systems & Control Lett 54(4):325–329

Chapter 4
Approximate modeling

> *In general no simple relationships are satisfied exactly by the data. This discrepancy between observed data and simple relationships is often modelled by introducing stochastics. However, instead of stochastic uncertainty it is in our opinion primarily the complexity of reality which often prevents existence of simple exact models. In this case model errors do not reflect chance, but arise because a simple model can only give an approximate representation of complex systems.*
>
> *Heij (1989)*

Data modeling using a linear static model class leads to an unstructured low-rank approximation problem. When the approximation criterion is the Frobenius norm, the problem can be solved by the singular value decomposition. In general, however, unstructured low-rank approximation is a difficult nonconvex optimization problem. Section 4.1 presents a local optimization algorithm, based on alternating projections. An advantage of the alternating projections algorithm is that it is easily generalizable to missing data, nonnegativity, and other constraints on the data.

In the case of nonlinear or dynamical model classes, the optimal approximate modeling problem leads to structured low-rank approximation. Section 4.2 presents a variable projection algorithm for affine structured low-rank approximation. This method is well suited for problems where one dimension of the matrix is small and the other one is large. This is the case in system identification, where the small dimension is determined by the model's complexity and the large dimension is determined by the number of samples.

The alternating projections and variable projection algorithms perform local optimization. An alternative approach for structured low-rank approximation is to solve a convex relaxation of the problem. A convex relaxation based on replacement of the rank constraint by a constraint on the nuclear norm is shown in Section 4.3. The nuclear norm heuristic is also applicable for matrix completion (missing data estimation). Section 4.4 generalize the variable projection method to solve low-rank matrix completion and approximation problems.

4.1 Unstructured low-rank approximation

As shown in Section 1.1, line fitting leads to rank-1 approximation. Line fitting is a special case of data modeling with a linear static model. This section shows that in the general case of optimal approximate data modeling with a linear static model class the equivalent computational problem is unstructured low-rank approximation.

Problem 4.1 (Unstructured low-rank approximation). Given a matrix $D \in \mathbb{R}^{\mathtt{q}\times N}$, with $\mathtt{q} \leq N$, a matrix norm $\|\cdot\|$, and an integer $\mathtt{m}$, $0 < \mathtt{m} < \mathtt{q}$, find a matrix

$$\widehat{D}^* := \arg\min_{\widehat{D}} \|D - \widehat{D}\| \quad \text{subject to} \quad \operatorname{rank}(\widehat{D}) \leq \mathtt{m}. \tag{LRA}$$

Section 4.1.1 makes the link between data modeling and low-rank approximation rigorous by introducing a *data generating model*. When the data is generated in the errors-in-variables setting and the approximation criterion is "properly" chosen with respect to the distribution of the measurement noise, (LRA) yields a statistically optimal (maximum likelihood) estimate for the true data generating system.

In the case of zero mean measurement noise with Gaussian distribution, the "proper" choice of the approximation criterion is the weighted 2-norm

$$\|\Delta D\|_W := \sqrt{\operatorname{vec}^\top(\Delta D) W \operatorname{vec}(\Delta D)}, \quad \text{for all } \Delta D, \tag{$\|\cdot\|_W$}$$

specified by an $\mathtt{q}N \times \mathtt{q}N$ positive semidefinite matrix W. In the errors-in-variables setting, W should be chosen as the inverse of the noise covariance matrix.

Section 4.1.2 presents a hierarchical classification of low-rank approximation problems (LRA) with approximation criterion ($\|\cdot\|_W$) (*weighted low-rank approximation problems*) according to the structure of the weight matrix. In a special case, presented in Section 4.1.3, the problem admits analytic solution in terms of the singular value decomposition. In general, however, analytic solution is not known and the problem is solved by local optimization methods. Section 4.1.4 presents an algorithm based on the alternating projections, which has the additional advantage of being able to handle missing data (or, equivalently, singular weight matrix W).

4.1.1 Linear static data modeling

Consider the approximate modeling problem (AM) for the model class of linear static models, *i.e.*, $\mathscr{M}_{\mathtt{c}_{\max}} = \mathscr{L}_{\mathtt{m},0}$, and the orthogonal distance approximation criterion, *i.e.*, $\operatorname{error}(\mathscr{D}, \mathscr{B}) = \operatorname{dist}(\mathscr{D}, \mathscr{B})$.

Problem 4.2 (Static data modeling). Given N, $\mathtt{q}$-variable observations

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

a matrix norm $\|\cdot\|$, and model complexity $\mathtt{m}$, $0 < \mathtt{m} < \mathtt{q}$,

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{D} \quad \|D - \widehat{D}\| \\ \text{subject to} & \text{image}(\widehat{D}) \subseteq \widehat{\mathscr{B}} \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}_{\mathtt{m},0}, \end{array} \qquad \text{(AM } \mathscr{L}_{\mathtt{m},0}\text{)}$$

where $D \in \mathbb{R}^{\mathtt{q}\times N}$ is the matrix $D := \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$, constructed from the data.

Proposition 4.3. *The approximate linear static data modeling problem (AM $\mathscr{L}_{\mathtt{m},0}$) is equivalent to the unstructured low-rank approximation problem (LRA).*

A solution $\widehat{\mathscr{B}}^*$ to (AM $\mathscr{L}_{\mathtt{m},0}$) is an optimal approximate model for the data D with complexity bounded by $\mathtt{m}$. Of course, $\widehat{\mathscr{B}}^*$ depends on the approximation criterion, specified by the given norm $\|\cdot\|$. A justification for the choice of the norm $\|\cdot\|$ is provided in the errors-in-variables setting (see Example 2.14 on page 64), *i.e.*, the data matrix D is assumed to be a noisy measurement of a true matrix $\bar{D}$

$$D = \bar{D} + \widetilde{D}, \quad \text{where } \text{image}(\bar{D}) \subseteq \bar{\mathscr{B}} \in \mathscr{L}_{\mathtt{m},0} \quad \text{and} \quad \text{vec}(\widetilde{D}) \sim \text{N}(0, s^2 V), \text{ with } V \succ 0. \qquad \text{(EIV}_0\text{)}$$

In (EIV$_0$), $\widetilde{D}$ is the measurement error. It is assumed to be a random matrix with zero mean and normal distribution. The true data matrix $\bar{D}$ is generated by a true model $\bar{\mathscr{B}}$ in the model class $\mathscr{L}_{\mathtt{m},0}$. The model $\bar{\mathscr{B}}$ is the object to be estimated in the errors-in-variables setting. The estimation is possible due to the prior knowledge of the matrix V and the true model's complexity $\mathtt{m}$. Note that the true noise distribution is not fully specified because s^2 is unknown.

The following proposition shows that the problem of computing the maximum likelihood estimator in the errors-in-variables setting (EIV$_0$) is equivalent to unstructured low-rank approximation (Problem 4.1) with the weighted norm $\|\cdot\|_{V^{-1}}$.

Proposition 4.4 (Maximum likelihood property of optimal static model $\widehat{\mathscr{B}}^*$). *Assuming that the data is generated in the errors-in-variables setting (EIV$_0$), with known matrix V, a solution $\widehat{\mathscr{B}}^*$ to Problem 4.2 with approximation criterion ($\|\cdot\|_W$) and weight $W = V^{-1}$ is a maximum likelihood estimator for the true model $\bar{\mathscr{B}}$.*

4.1.2 A hierarchy of weighted low-rank approximation problems

In ($\|\cdot\|_W$), we vectorized the ΔD matrix column-wise. In addition to, the "column-wise weight matrix" W we use in this section the "row-wise weight matrix" $\bar{W}$,

$$\|\Delta D\|_W = \|\Delta D^\top\|_{\bar{W}}, \quad \text{for all } \Delta D \in \mathbb{R}^{\mathtt{q}\times N}.$$

i.e., $\bar{W}$ defines the approximation criterion by the row-wise vectorization of ΔD.

Special cases of the weighted low-rank approximation problem are obtained when the weight matrices W and $\bar{W}$ have block diagonal structure with the same or repeated blocks on the diagonal.

- *Column-wise weighting:* $W = \text{diag}(W_1, \ldots, W_N)$, where $W_j \in \mathbb{R}^{\mathtt{q}\times\mathtt{q}}$, $W_j \geq 0$.
- *Column-wise weighting with equal weight matrix for all columns:*

$$W = \text{diag}(\underbrace{W_1, \ldots, W_1}_{N}), \quad \text{where } W_1 \in \mathbb{R}^{\mathtt{q}\times\mathtt{q}},\ W_1 \geq 0.$$

- *Row-wise weighting:* $\bar{W} = \text{diag}(W_1, \ldots, W_\mathtt{q})$, where $W_i \in \mathbb{R}^{N\times N}$, $W_i \geq 0$.
- *Row-wise weighting with equal weight matrix for all rows:*

$$\bar{W} = \text{diag}(\underbrace{W_\text{r}, \ldots, W_\text{r}}_{\mathtt{q}}), \quad \text{where } W_\text{r} \in \mathbb{R}^{N\times N},\ W_\text{r} \geq 0.$$

- *Element-wise weighting:* $W = \text{diag}(w_1, \ldots, w_{\mathtt{q}N})$, where $\underbrace{\text{col}(w_1, \ldots, w_{\mathtt{q}N})}_{=:\, w} \geq 0$.

In the case of element-wise weights, the cost function can be written using the Hadamard product $\odot$ as

$$\|D - \widehat{D}\|_\Sigma := \|\Sigma \odot (D - \widehat{D})\|_\text{F} = \sqrt{\sum_{i=1}^{\mathtt{q}} \sum_{j=1}^{N} \sigma_{ij}(d_{ij} - \widehat{d}_{ij})}, \qquad (\|\cdot\|_\Sigma)$$

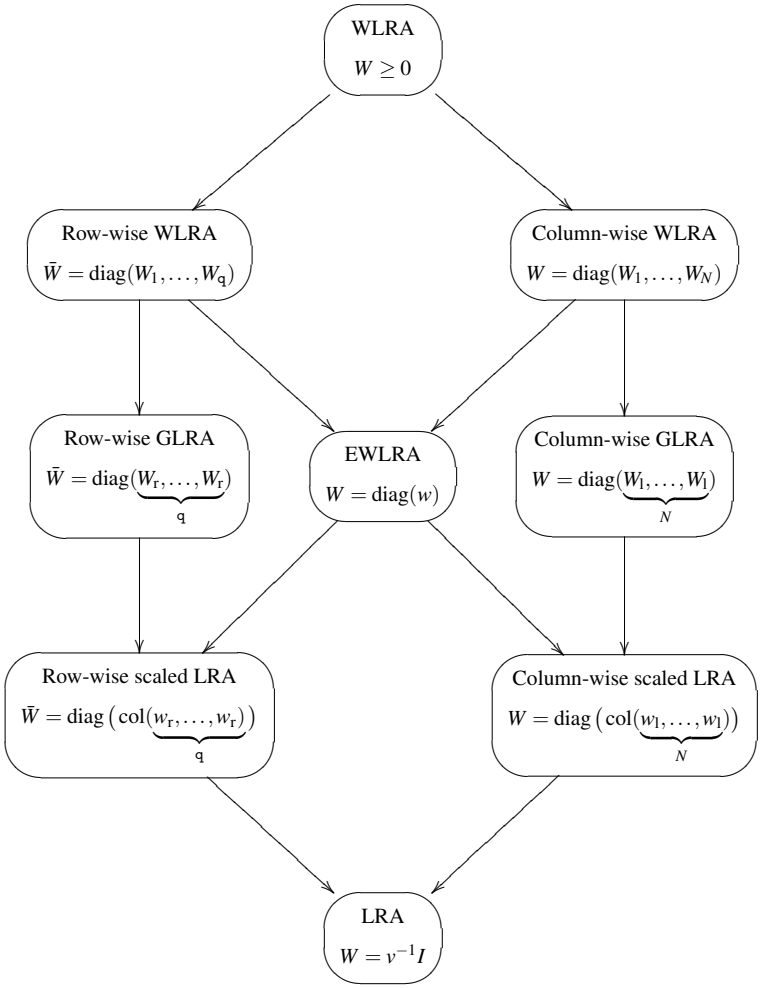
where $\Sigma \in \mathbb{R}^{\mathtt{q}\times N}$ is such that $\text{vec}(\Sigma) = \sqrt{w}$. Note that for a zero weight, *e.g.*, $\sigma_{ij} = 0$, the corresponding element d_{ij} of D is not taken into account in the approximation criterion and it is therefore treated as a missing value.

As shown in the next section, column/row-wise weighting with equal matrix for all columns/rows corresponds to approximation criteria $\|\sqrt{W_1}\Delta D\|_\text{F}$ and $\|\Delta D\sqrt{W_\text{r}}\|_\text{F}$. The (AM $\mathscr{L}_{\mathtt{m},0}$) problem with criterion $\|\sqrt{W_1}\Delta D\sqrt{W_\text{r}}\|_\text{F}$ is called *two-sided weighted*, also known as the *generalized low-rank approximation* problem. This latter problem allows analytic solution in terms of the singular value decomposition.

Figure 4.1 shows the hierarchy of weighted low-rank approximation problems.

4.1.3 Special cases with known analytic solutions

An extreme special case of the weighted low-rank approximation problem is the "unweighted" case, *i.e.*, weight matrix a multiple of the identity. Then, $\|\cdot\|_W$ is proportional to the Frobenius norm $\|\cdot\|_\text{F}$ and the low-rank approximation problem has an analytic solution in terms of the singular value decomposition of D. The results is known as the Eckart–Young–Mirsky theorem or the matrix approximation lemma. In view of its importance, we refer to this case as the *basic low-rank approximation*.



LRA — low-rank approximation
WLRA — weighted low-rank approximation
GLRA — generalized low-rank approximation
EWLRA — element-wise weighted LRA

Fig. 4.1: Weighted low-rank approximation problems can be classified according to the structure of the weight matrix W in a hierarchy, starting on the top with the most general problem of positive semi-definite weight matrix and coming at the bottom with the most restrictive problem (unweighted low-rank approximation) of weight matrix which is a multiple of the identity. On the left-hand side are weighted low-rank approximation problems with row-wise weighting and on the right-hand side are weighted low-rank approximation problems with column-wise weighting.

Theorem 4.5 (Eckart–Young–Mirsky). *Let $D = U\Sigma V^\top$ be the singular value decomposition of D and block-partition U, $\Sigma =: \mathrm{diag}(\sigma_1, \ldots, \sigma_{\mathtt{q}})$, and V as follows:*

$$U =: \overset{\mathtt{m}\quad \mathtt{q}-\mathtt{m}}{\begin{bmatrix} U_1 & U_2 \end{bmatrix}}\ \begin{matrix}\\ \mathtt{q}\end{matrix}, \quad \Sigma =: \overset{\mathtt{m}\quad \mathtt{q}-\mathtt{m}}{\begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}}\ \begin{matrix}\mathtt{m}\\ \mathtt{q}-\mathtt{m}\end{matrix} \quad \text{and} \quad V =: \overset{\mathtt{m}\quad \mathtt{q}-\mathtt{m}}{\begin{bmatrix} V_1 & V_2 \end{bmatrix}}\ \begin{matrix}\\ N\end{matrix},$$

Then the rank-$\mathtt{m}$ matrix, obtained from the truncated singular value decomposition

$$\widehat{D}^* = U_1 \Sigma_1 V_1^\top,$$

is a solution to (LRA) with the Frobenius norm. The approximation error is

$$\|D - \widehat{D}^*\|_{\mathrm{F}} = \min_{\mathrm{rank}(\widehat{D}) \leq \mathtt{m}} \|D - \widehat{D}\|_{\mathrm{F}} = \sqrt{\sigma_{\mathtt{m}+1}^2 + \cdots + \sigma_{\mathtt{q}}^2}.$$

The minimizer $\widehat{D}^*$ is unique if and only if $\sigma_{\mathtt{m}+1} \neq \sigma_{\mathtt{m}}$.

Note 4.6 (Unitarily invariant norms). Theorem 4.5 holds for any norm $\|\cdot\|$ that is invariant under orthogonal transformations, *i.e.*, satisfying the relation

$$\|U\Delta DV\| = \|\Delta D\|, \quad \text{for any } \Delta D \text{ and for any orthogonal matrices } U \text{ and } V.$$

Note 4.7 (Approximation in the spectral norm). For a matrix ΔD, let $\|\Delta D\|_2$ be the spectral (2-norm induced) matrix norm $\|\Delta D\|_2 = \sigma_{\max}(\Delta D)$. Then

$$\min_{\mathrm{rank}(\widehat{D}) = \mathtt{m}} \|D - \widehat{D}\|_2 = \sigma_{\mathtt{m}+1},$$

i.e., the optimal rank-$\mathtt{m}$ spectral norm approximation error is equal to the first neglected singular value. The truncated singular value decomposition yields an optimal approximation with respect to the spectral norm, however, in this case a minimizer is not unique even when the singular values $\sigma_{\mathtt{m}}$ and $\sigma_{\mathtt{m}+1}$ are different.

As defined, the low-rank approximation problem aims at a matrix $\widehat{D}$ that is a solution to the optimization problem (LRA). In data modeling problems, however, of primary interest is the optimal model $\widehat{D}$, *i.e.*, the most powerful unfalsified model for $\widehat{D}^*$. Theorem 4.5 gives the optimal approximating matrix $\widehat{D}^*$ in terms of the singular value decomposition of the data matrix D. Minimal parameters of kernel and image representations of the corresponding optimal model are directly available from the factors of the singular value decomposition of D.

Corollary 4.8 (Optimal data model). *An optimal in the Frobenius norm approximate model in the model class $\mathscr{L}_{\mathtt{m},0}$,* i.e., *$\widehat{\mathscr{B}}^* := \mathscr{B}_{\mathrm{mpum}}(\widehat{D}^*)$ is unique if and only if the singular values $\sigma_{\mathtt{m}}$ and $\sigma_{\mathtt{m}+1}$ of D are different, in which case*

$$\widehat{\mathscr{B}}^* = \ker(U_2^\top) = \mathrm{image}(U_1). \qquad (R^*, P^*)$$

105a ⟨*low-rank approximation* 105a⟩≡
```
function [R, P, dh] = lra(d, r)
[u, s, v] = svd(d); R = u(:, (r + 1):end)'; P = u(:, 1:r);
if nargout > 2, dh = u(:, 1:r) * s(1:r, 1:r) * v(:, 1:r)'; end
```
Defines:
lra, used in chunks 116b, 118c, 124b, 185d, 187a, 227a, 245, and 253c.

Corollary 4.9 (Nested approximations). *The optimal in the Frobenius norm approximate models $\widehat{\mathscr{B}}^*_{\mathtt{m}}$ in the model classes $\mathscr{L}_{\mathtt{m},0}$, $\mathtt{m} = 1,\ldots,\mathtt{q}$ are nested,* i.e.*,*

$$\widehat{\mathscr{B}}_{\mathtt{q}} \subseteq \widehat{\mathscr{B}}_{\mathtt{q}-1} \subseteq \cdots \subseteq \widehat{\mathscr{B}}_1.$$

Note 4.10 (Efficient computation using QR factorization when $N \gg \mathtt{q}$). Note from (R^*,P^*) that an optimal model $\widehat{\mathscr{B}}^*$ for D depends only on the left singular vectors of D. Since post multiplication of D by an orthogonal matrix Q does not change the left singular vectors, $\widehat{\mathscr{B}}^*$ is an optimal model for the data matrix DQ. For $N \gg \mathtt{q}$, computing the QR factorization $D^\top = Q^\top \left[\begin{smallmatrix} R_1 \\ 0 \end{smallmatrix}\right]$, followed by the singular value decomposition of R_1 is an efficient way for finding $\widehat{\mathscr{B}}$.

105b ⟨*data compression* 105b⟩≡ (107a)
```
d = triu(qr(d'))'; d = d(:, 1:q);
```

Note 4.11 (Efficient computation avoiding the full singular value decomposition). The fact that the parameters R^* and P^* of a kernel and image representations of the optimal model $\mathscr{B}^*$ depend only on some of the left singular vectors (see (R^*,P^*)) shows that the full singular value decomposition of the data matrix D is in fact not needed. Indeed, a representation of the optimal model can be obtained only from the first $\mathtt{m}$ or the last $\mathtt{p} := \mathtt{q} - \mathtt{m}$ left singular vectors. There are methods that allow efficient computation of a few leading or trailing singular values and corresponding singular vectors. Depending on the complexity of the model, it is computationally cheaper to obtain R^* ($\mathtt{p} < \mathtt{m}$) or P^* ($\mathtt{m} < \mathtt{p}$).

Similar analytic solution to the one in Theorem 4.5 is not known for the general weighted low-rank approximation problem (LRA) with ($\|\cdot\|_W$). Presently, the largest class of weighted low-rank approximation problems with analytic solution are those with a weight matrix W of the form

$$W = W_{\mathrm{r}} \otimes W_{\mathrm{l}}, \quad \text{where } W_{\mathrm{l}} \in \mathbb{R}^{\mathtt{q}\times\mathtt{q}} \text{ and } W_{\mathrm{r}} \in \mathbb{R}^{N\times N} \qquad (W_{\mathrm{r}} \otimes W_{\mathrm{l}})$$

are positive definite matrices and $\otimes$ is the Kronecker product.

Using the identities

$$\operatorname{vec}(AXB) = (B^\top \otimes A)\operatorname{vec}(X)$$

and

$$(A_1 \otimes B_1)(A_2 \otimes B_2) = (A_1A_2) \otimes (B_1B_2),$$

we have

$$\begin{aligned}\|D - \widehat{D}\|_{W_{\mathrm{r}}\otimes W_{\mathrm{l}}} &= \sqrt{\operatorname{vec}^\top(\Delta D)\,(W_{\mathrm{r}} \otimes W_{\mathrm{l}})\operatorname{vec}(\Delta D)} \\ &= \big\|(\sqrt{W_{\mathrm{r}}} \otimes \sqrt{W_{\mathrm{l}}})\operatorname{vec}(\Delta D)\big\|_2 \\ &= \big\|\operatorname{vec}(\sqrt{W_{\mathrm{l}}}\Delta D\sqrt{W_{\mathrm{r}}})\big\|_2 \\ &= \|\sqrt{W_{\mathrm{l}}}\Delta D\sqrt{W_{\mathrm{r}}}\|_{\mathrm{F}}.\end{aligned}$$

Therefore, the low-rank approximation problem (LRA) with norm ($\|\cdot\|_W$) and weight matrix ($W_{\mathrm{r}} \otimes W_{\mathrm{l}}$) is equivalent to the two-sided weighted problem

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{D} \quad \|\sqrt{W_{\mathrm{l}}}(D - \widehat{D})\sqrt{W_{\mathrm{r}}}\|_{\mathrm{F}} \\ &\text{subject to} && \operatorname{rank}(\widehat{D}) \leq \mathtt{m}. \end{aligned} \qquad \text{(WLRA2)}$$

As shown next, (WLRA2) has an analytic solution.

Theorem 4.12 (Two-sided weighted low-rank approximation). *Define the modified data matrix*

$$D_{\mathtt{m}} := \sqrt{W_{\mathrm{l}}}D\sqrt{W_{\mathrm{r}}},$$

and let $\widehat{D}^*_{\mathtt{m}}$ be the optimal (unweighted) low-rank approximation of $D_{\mathtt{m}}$. Then

$$\widehat{D}^* := \big(\sqrt{W_{\mathrm{l}}}\big)^{-1}\widehat{D}^*_{\mathtt{m}}\big(\sqrt{W_{\mathrm{r}}}\big)^{-1},$$

is a solution of the following two-sided weighted low-rank approximation problem (WLRA2). A solution always exists. It is unique if and only if $\widehat{D}^*_{\mathtt{m}}$ is unique.

The proof is left as an exercise (Exercise 4.3).

For a diagonal W, *i.e.*, element-wise weighting($\|\cdot\|_\Sigma$), the special case ($W_{\mathrm{r}} \otimes W_{\mathrm{l}}$) with analytical solution corresponds to a rank-1 matrix Σ of element-wise weights.

4.1.4 Alternating projections algorithm

In this section, we consider the low-rank approximation problem (LRA) with an element-wise weighted cost function ($\|\cdot\|_\Sigma$). The solution method is based on local optimization and uses the alternating projections algorithm. The first step in the derivation of the method is to use the image representation of the rank constraint

$$\operatorname{rank}(\widehat{D}) \leq \mathtt{m} \iff \text{there are } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \text{ and } L \in \mathbb{R}^{\mathtt{m}\times N}, \text{ such that } \widehat{D} = PL \quad \text{(IM)}$$

in order to obtain an equivalent parameter optimization problem

$$\text{minimize} \quad \text{over } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \text{ and } L \in \mathbb{R}^{\mathtt{m}\times N} \quad \|D - PL\|_\Sigma. \qquad \text{(LRA-IM)}$$

The alternating projections method exploits the fact that problem (LRA-IM) is bilinear, *i.e.*, it is a linear least squares problem in either P or L. The algorithm alternates between solving the two linear least squares problems:

$$\text{minimize} \quad \text{over } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \quad \|D-PL\|_\Sigma, \qquad \text{(LRA-IM}_P\text{)}$$

and

$$\text{minimize} \quad \text{over } L \in \mathbb{R}^{\mathtt{m}\times N} \quad \|D-PL\|_\Sigma. \qquad \text{(LRA-IM}_L\text{)}$$

Problems (LRA-IM$_P$) and (LRA-IM$_L$) are separable. Indeed, (LRA-IM$_L$) decouples into N independent least squares problems for the columns $l_1,\ldots,l_N$ of L,

$$\text{minimize} \quad \text{over } l_i \in \mathbb{R}^{\mathtt{m}} \quad \|d_i-Pl_i\|_{\Sigma_{:,i}}, \quad \text{for } i=1,\ldots,N. \qquad \text{(LRA-IM}_{l_i}\text{)}$$

Similarly, (LRA-IM$_P$) decouples into $\mathtt{q}$ independent least squares problems. The separability property allows more efficient implementation.

With a small modification, the alternating projections method can deal with missing values in D. Let $\mathscr{I}_\mathrm{g}$ be the set of indeces of the given elements in d_i. Using the matrix/vector indexing notation, introduced in Section 3.4.1, (LRA-IM$_{l_i}$) becomes

$$\text{minimize} \quad \text{over } l_i \in \mathbb{R}^{\mathtt{m}} \quad \|D_{\mathscr{I}_\mathrm{g},i}-P_{\mathscr{I}_\mathrm{g},:}l_i\|_{\Sigma_{\mathscr{I}_\mathrm{g},i}}.$$

The resulting alternating projections method for element-wise weighted low-rank approximation with missing values is summarized in Algorithm 6.

As a function of the iteration step, the cost function value is non-increasing. It can be shown that the iteration converges to a locally optimal solution of (LRA-IM) and that the local convergence rate is linear. The quantity $e^{(k)}$, computed on step 9 of the algorithm is the squared approximation error $e^{(k)}=\|D-\widehat{D}^{(k)}\|_\Sigma^2$ on the kth iteration step. Convergence of the iteration is judged on the basis of the relative decrease of the error $e^{(k)}$ after an update step. This corresponds to choosing a tolerance ε on the relative decrease of the cost function value. More expensive alternatives are to check the convergence of the approximation $\widehat{D}^{(k)}$ or the size of the gradient of the cost function with respect to the model parameters.

Initial approximation

The initial approximation for the iterative optimization method, see step 1 of Algorithm 6, is obtained by solving unweighted low-rank approximation problem where all missing elements (encoded as `NaN`'s) are filled in with zeros.

107a ⟨*low-rank approximation with missing data* 107a⟩≡

```
function [p, l] = lra_md(d, m)
d(isnan(d)) = 0; [q, N] = size(d);
if nargout == 1, ⟨data compression 105b⟩, end
⟨matrix approximation 107b⟩
```

The problem is solved using the singular value decomposition, however, in view of the large scale of the data the function `svds` which computes selected singular values and corresponding singular vectors is used instead of the function `svd`.

107b ⟨*matrix approximation* 107b⟩≡ (107a)

```
[u, s, v] = svds(d, m); p = u(:, 1:m); l = p' * d;
```


Algorithm 6 Alternating projections for weighted low-rank approximation.

Input: Data matrix $D\in\mathbb{R}^{\mathtt{q}\times N}$, rank constraint $\mathtt{m}$, elementwise nonnegative weight matrix $\Sigma\in\mathbb{R}^{\mathtt{q}\times N}$, and relative convergence tolerance ε.

1: Initial approximation: $P^{(0)}:=\mathtt{lra\_md}(D,\mathtt{m})$ {Frobenius norm low-rank approximation}
2: Let $k:=0$.
3: **repeat**
4: Let $e^{(k)}:=0$.
5: **for** $i=1,\ldots,N$ **do**
6: Let $\mathscr{I}_\mathrm{g}$ be the set of indexes of the given elements in $D_{:,i}$.
7: Define

$$c:=\operatorname{diag}(\Sigma_{\mathscr{I}_\mathrm{g},i})D_{\mathscr{I}_\mathrm{g},i}=\Sigma_{\mathscr{I}_\mathrm{g},i}\odot D_{\mathscr{I}_\mathrm{g},i}$$
$$P:=\operatorname{diag}(\Sigma_{\mathscr{I}_\mathrm{g},i})P^{(k)}_{\mathscr{I}_\mathrm{g},:}=(\Sigma_{\mathscr{I}_\mathrm{g},i}\mathbf{1}_\mathtt{m}^\top)\odot P^{(k)}_{\mathscr{I}_\mathrm{g},:}$$

8: Compute

$$\ell_i^{(k)}:=\left(P^\top P\right)^{-1}P^\top c.$$

9: Let

$$e^{(k)}:=e^{(k)}+\|c-P\ell_i^{(k)}\|^2.$$

10: **end for**
11: Define

$$L^{(k)}=\begin{bmatrix}\ell_1^{(k)} & \cdots & \ell_N^{(k)}\end{bmatrix}.$$

12: Let $e^{(k+1)}:=0$.
13: **for** $i=1,\ldots,\mathtt{q}$ **do**
14: Let $\mathscr{I}_\mathrm{g}$ be the set of indexes of the given elements in the ith row $D_{i,:}$.
15: Define

$$r:=D_{i,\mathscr{I}_\mathrm{g}}\operatorname{diag}(\Sigma_{i,\mathscr{I}_\mathrm{g}})=D_{i,\mathscr{I}_\mathrm{g}}\odot\Sigma_{i,\mathscr{I}_\mathrm{g}}$$
$$L:=L^{(k)}_{:,\mathscr{I}_\mathrm{g}}\operatorname{diag}(\Sigma_{i,\mathscr{I}_\mathrm{g}})=L^{(k)}_{:,\mathscr{I}_\mathrm{g}}\odot(\mathbf{1}_\mathtt{m}\Sigma_{i,\mathscr{I}_\mathrm{g}}).$$

16: Compute

$$p_i^{(k+1)}:=rL^\top\left(LL^\top\right)^{-1}.$$

17: Let

$$e^{(k+1)}:=e^{(k+1)}+\|r-p_i^{(k+1)}L\|^2.$$

18: **end for**
19: Define

$$P^{(k+1)}=\begin{bmatrix}p_1^{(k+1)}\\ \vdots\\ p_\mathtt{q}^{(k+1)}\end{bmatrix}.$$

20: $k=k+1$.
21: **until** $|e^{(k)}-e^{(k-1)}|/e^{(k)}<\varepsilon$.

Output: Locally optimal solution $\widehat{D}=\widehat{D}^{(k)}:=P^{(k)}L^{(k)}$ of (LRA-IM).

Note 4.13 (Large scale, sparse data). In an application of (LRA-IM) to building recommender systems, the data matrix D is large but only a small fraction of the elements are given. Such problems can be handled efficiently by encoding D and Σ as sparse matrices. The convention in this case is that missing d_{ij}'s are zeros.

4.2 Structured low-rank approximation

In this section, we generalize the low-rank approximation Problem 4.1 to problems that involves matrix structure, *i.e.*, the data matrix is structured (depends on parameters) and this structure should be preserved by the approximation. First, we show the relevance of Hankel structured low-rank approximation for modeling of autonomous linear time-invariant systems. Then, we present a special case of circular structure that has an analytic solution. Finally, we present a variable projection method for affine structured low-rank approximation. The method is specialized for the application of system identification, in which case the matrix structure is Hankel.

4.2.1 Autonomous linear time-invariant data modeling

As a motivation for structured low-rank approximation consider an example from system theory: the data matrix D being low-rank and Hankel structured is equivalent to the data being generated by a bounded complexity linear time-invariant system. To show this, consider first the special case of a scalar Hankel structure

$$\mathscr{H}_{\ell+1}(p) := \begin{bmatrix} p_1 & p_2 & \cdots & p_{n_p-\ell} \\ p_2 & p_3 & \cdots & p_{n_p-\ell+1} \\ \vdots & \vdots & & \vdots \\ p_{\ell+1} & p_{\ell+2} & \cdots & p_{n_p} \end{bmatrix}.$$

The approximation matrix $\widehat{D} = \mathscr{H}_{\ell+1}(\widehat{p})$ being rank deficient implies that there is a nonzero vector $R = \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix}$, such that

$$R\mathscr{H}_{\ell+1}(\widehat{p}) = 0.$$

Due to the Hankel structure, this system of equations can be written as

$$R_0\widehat{p}_t + R_1\widehat{p}_{t+1} + \cdots + R_\ell\widehat{p}_{t+\ell} = 0, \quad \text{for } t = 1,\ldots,n_p-\ell,$$

i.e., a homogeneous constant coefficients difference equation. Therefore, $\widehat{p}$ is a trajectory of an autonomous linear time-invariant system $\mathscr{B}$, defined by (KER). More generally, recall that for a multivariable autonomous system $\mathscr{B}$,

$$\dim(\mathscr{B}|_T) = \mathtt{n}(\mathscr{B}), \quad \text{for } T \geq \ell(\mathscr{B}),$$

where $\mathtt{n}(\mathscr{B})$ and $\ell(\mathscr{B})$ are, respectively, the order and the lag of the system $\mathscr{B}$.

The Hankel low-rank approximation problem is equivalent to the following modeling problem: Given signal $y_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{p}})^T$, norm $\|\cdot\|$, and a model complexity ℓ,

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{y} \quad \|y_{\mathrm{d}} - \widehat{y}\| \\ &\text{subject to} && \widehat{y} \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}_{0,\ell}. \end{aligned} \tag{AM $\mathscr{L}_{0,\ell}$}$$

A solution $\widehat{\mathscr{B}}^*$ is an optimal approximate model for w_{d} in the model class $\mathscr{L}_{0,\ell}$.

In the general case when the model has inputs, the parameters R_i of a kernel representation are $\mathtt{p} \times \mathtt{q}$ matrices. The Hankel structured low-rank approximation problem is equivalent then to the approximate linear time-invariant modeling problem (AM) with model class $\mathscr{M}_{\mathtt{c}_{\max}} = \mathscr{L}_{\mathtt{m},\ell}$ and orthogonal distance fitting criterion.

Problem 4.14 (Linear time-invariant dynamic modeling). Given T–samples, $\mathtt{q}$–variate, vector signal $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{q}})^T$, signal norm $\|\cdot\|$, and complexity bound $(\mathtt{m},\ell)$,

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{w} \quad \|w_{\mathrm{d}} - \widehat{w}\| \\ &\text{subject to} && \widehat{w} \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}. \end{aligned} \tag{AM $\mathscr{L}_{\mathtt{m},\ell}$}$$

The solution $\widehat{\mathscr{B}}^*$ of (AM $\mathscr{L}_{\mathtt{m},\ell}$) is an optimal approximate model for the signal w_{d} with complexity bounded by $(\mathtt{m},\ell)$. Note that problem (AM $\mathscr{L}_{\mathtt{m},\ell}$) reduces to

- (AM $\mathscr{L}_{0,\ell}$) when $\mathtt{m} = 0$, *i.e.*, when the model is autonomous, and
- (AM $\mathscr{L}_{\mathtt{m},0}$) when $\ell = 0$, *i.e.*, when the model is static.

Therefore, (AM $\mathscr{L}_{\mathtt{m},\ell}$) is a proper generalization of linear static and dynamic autonomous data modeling problems.

As shown in Section 5.2, the approximate linear time-invariant modeling problem (AM $\mathscr{L}_{\mathtt{m},\ell}$) is equivalent to a structured low-rank approximation problem (SLRA).

Problem SLRA (Structured low-rank approximation). Given a structure specification $\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m\times n}$, with $m \leq n$, a parameter vector $p \in \mathbb{R}^{n_p}$, a vector norm $\|\cdot\|$, and an integer r, $0 < r < \min(m,n)$,

$$\text{minimize} \quad \text{over } \widehat{p} \quad \|p - \widehat{p}\| \quad \text{subject to} \quad \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r. \tag{SLRA}$$

The matrix $\widehat{D}^* := \mathscr{S}(\widehat{p}^*)$ is an optimal approximation of the matrix $D := \mathscr{S}(p)$ with rank at most r within the class of matrices with structure $\mathscr{S}$.

Computing the optimal approximate model $\widehat{\mathscr{B}}^*$ from the solution $\widehat{p}^*$ of Problem SLRA is an *exact* identification problem.

Similarly to the static modeling problem, the dynamic modeling problem has a maximum likelihood interpretation in the errors-in-variables setting.

Proposition 4.15 (Maximum likelihood property of an optimal dynamic model). *Assume that the data w_{d} is generated in the errors-in-variables setting*

$$w_{\mathrm{d}} = \overline{w} + \widetilde{w}, \quad \text{where } \overline{w} \in \bar{\mathscr{B}}|_T \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell} \text{ and } \widetilde{w} \sim \mathrm{N}(0, s^2 I). \tag{EIV}$$

Then an optimal approximate model $\widehat{\mathscr{B}}^*$, solving (AM $\mathscr{L}_{\mathtt{m},\ell}$) with $\|\cdot\| = \|\cdot\|_2$, is a maximum likelihood estimator for the true model $\bar{\mathscr{B}}$.

4.2.2 Special cases with known analytic solutions

We showed in Section 4.1.3 that weighted unstructured low-rank approximation problems with rank-1 weight matrix have analytic solution in terms of the singular value decomposition of a modified data matrix. Similar result exists for circulant structured low-rank approximation. If the approximation criterion is a unitarily invariant matrix norm, the unstructured low-rank approximation (obtained for example from the truncated singular value decomposition) is unique. In the case of a circulant structure, it turns out that this unique minimizer also has circulant structure, so that the structure constraint is satisfied without explicitly enforcing it in the approximation problem (Beck and Ben-Tal, 2006; Chu and Plemmons, 2003).

An efficient computational method for obtaining the circulant structured low-rank approximation is the fast Fourier transform. Consider the scalar case and let

$$P_k := \textstyle\sum_{j=1}^{n_p} p_j e^{-\mathbf{i}\frac{2\pi}{n_p}kj}$$

be the discrete Fourier transform of p. Denote with $\mathscr{K}$ the subset of $\{1,\ldots,n_p\}$ consisting of the indexes of the $\mathtt{m}$ largest elements of $\{|P_1|,\ldots,|P_{n_p}|\}$. Assuming that $\mathscr{K}$ is uniquely defined by the above condition, *i.e.*, assuming that

$$k \in \mathscr{K} \text{ and } k' \notin \mathscr{K} \quad \Longrightarrow \quad |P_k| > |P_{k'}|,$$

the solution $\widehat{p}^*$ of the structured low-rank approximation problem (SLRA) with $\mathscr{S}$ a circulant matrix is unique and is given by $\widehat{p}^* = \frac{1}{n_p}\sum_{k\in\mathscr{K}} P_k e^{\mathbf{i}\frac{2\pi}{n_p}kj}$.

4.2.3 Variable projection algorithm

Next, we consider local optimization methods for affine structured low-rank approximation. First, we convert the rank constraint into an equivalent bilinear equality constraint using the kernel representation. Then, we describe in a literate programming style a method based on the variable projection. Finally, we specialize the method for single-input single-output linear time-invariant system identification, in which case the matrix is Hankel structured and efficient algorithms exist.

Parameterization of rank constraint via kernel representation

Consider the structured low-rank approximation problem (SLRA) with a weighted 2-norm $\|\Delta p\|_W := \Delta p^\top W \Delta p$. Different methods for solving the problem can be obtained by choosing different combinations of rank representation and optimization method. In this section, we choose the kernel representation of the rank constraint

$$\operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \le r \quad \Longleftrightarrow \quad \text{there is } R \in \mathbb{R}^{(m-r)\times m}, \text{ such that } R\mathscr{S}(\widehat{p}) = 0 \text{ and } RR^\top = I_{m-r}, \tag{rank$_R$}$$

and the variable projection approach (in combination with standard nonlinear optimization methods) for solving the resulting parameter optimization problem.

> The developed method is applicable for the general affinely structured and weighted low-rank approximation problem (SLRA). The price paid for the generality, however, is lack of efficiency compared to specialized methods exploiting the structure of the data matrix $\mathscr{S}(p)$ and the weight matrix W.

Representing the constraint of (SLRA) in the kernel form (rank$_R$), leads to the double minimization problem

$$\text{minimize} \quad \text{over } R \in \mathbb{R}^{(m-r)\times m} \quad M(R) \quad \text{subject to} \quad RR^\top = I_{m-r}, \tag{SLRA$_R$}$$

where

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

The computation of $M(R)$, called "inner" minimization, is over the estimate $\widehat{p}$ of p. The minimization over the kernel parameter $R \in \mathbb{R}^{(m-r)\times m}$ is called "outer". The inner minimization problem is a projection of the columns of $\mathscr{S}(p)$ onto the model $\mathscr{B} := \ker(R)$. Note that, the projection depends on the parameter R, which is the variable in the outer minimization problem. Thus, the name "variable projection".

For affine structures $\mathscr{S}$, the constraint $R\mathscr{S}(\widehat{p}) = 0$ is bilinear in the variables R and $\widehat{p}$. Then, the evaluation of the cost function M for the outer minimization problem is a linear least norm problem. Direct solution has cubic computational complexity in the number of structure parameters. Exploiting the structure of the problem (inherited from $\mathscr{S}$), results in computational methods with quadratic or linear complexity, depending on the type of structure. For a class of structures, called mosaic Hankel, which includes Hankel, Toeplitz, and Sylvester, the complexity for the cost function and Jacobian evaluation is linear (Usevich and Markovsky, 2014).

A variable projection method for affinely structured problems

Next, we present a literate program for solution of problem (SLRA$_R$), using the variable projection principle. First, we explain how the matrix structure $\mathscr{S}$ is represented in the code. Then, we consider the subproblem of minimization with respect to the parameter correction $\widehat{p}$. This problem is a linear least norm problem and has a closed form solution. Finally, we present the solution of the remaining problem of minimization with respect to the parameter R. This is a nonlinear least squares problem and is solved by standard local optimization methods.

Structure specification

The general affine structure

$$\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m\times n}, \quad \mathscr{S}(\widehat{p}) = S_0 + \sum_{k=1}^{n_p} S_k \widehat{p}_k \qquad (\mathscr{S}(\widehat{p}))$$

is specified by the n_p matrices $S_0, S_1, \ldots, S_{n_p} \in \mathbb{R}^{m\times n}$. This data is represented in the MATLAB implementation by an $m \times n$ matrix variable `s0`, corresponding to the matrix S_0, and an $mn \times n_p$ matrix variable `bfs`, corresponding to the matrix

$$\mathbf{S} := \begin{bmatrix} \operatorname{vec}(S_1) & \cdots & \operatorname{vec}(S_{n_p}) \end{bmatrix} \in \mathbb{R}^{mn\times n_p}.$$

The evaluation of $(\mathscr{S}(\widehat{p}))$ is then implemented as a matrix–vector product:

$$\operatorname{vec}\big(\mathscr{S}(\widehat{p})\big) = \operatorname{vec}(S_0) + \mathbf{S}\widehat{p}, \quad \text{or} \quad \mathscr{S}(\widehat{p}) = S_0 + \operatorname{vec}^{-1}(\mathbf{S}\widehat{p}). \qquad (\mathbf{S})$$

113 ⟨$(S_0, \mathbf{S}, \widehat{p}) \mapsto \widehat{D} = \mathscr{S}(\widehat{p})$ 113⟩≡
```
dh = s0 + reshape(bfs * ph, m, n);
```

Note 4.16 (Sparsity of S_0 and $\mathbf{S}$). In many applications the matrices S_k are sparse, so that, for efficiency, they can be stored and manipulated as sparse matrices.

A commonly encountered special case of an affine structure is when each element of the structured matrix $\mathscr{S}(p)$ depends on one structure parameter.

$$\big[\mathscr{S}(\widehat{p})\big]_{ij} = \begin{cases} S_{0,ij}, & \text{if } \mathtt{S}_{ij} = 0 \\ S_{0,ij} + \widehat{p}_{\mathtt{S}_{ij}} & \text{otherwise} \end{cases} \qquad \text{where } \mathtt{S}_{ij} \in \{0, 1, \ldots, n_p\}^{m\times n}, \qquad (\mathtt{S})$$

or, defining the extended structure parameter vector $\widehat{p}_{\text{ext}} := \begin{bmatrix} 0 \\ \widehat{p} \end{bmatrix}$

$$\big[\mathscr{S}(\widehat{p})\big]_{ij} = S_{0,ij} + \widehat{p}_{\text{ext},\mathtt{S}_{ij}}.$$

In (S), the structure is specified by the matrices S_0 and `S`. Although (S) is a special case of the affine structure $(\mathscr{S}(\widehat{p}))$, it covers all linear modeling problems considered in the book and will be used in the implementation of the solution method.

In the implementation of the algorithm, the matrix `S` corresponds to a variable `tts` and the extended parameter vector p_{ext} corresponds to a variable `pext`. Since in MATLAB indeces are positive integers (zero index is not allowed), in all indexing operations of `pext`, the index is incremented by one. Given the matrices S_0 and `S` and a structure parameter vector $\widehat{p}$, the matrix $\mathscr{S}(\widehat{p})$ is constructed by

114a ⟨$(S_0, \mathtt{S}, \widehat{p}) \mapsto \widehat{D} = \mathscr{S}(\widehat{p})$ 114a⟩≡ (116b 121b)
```
phext = [0; ph(:)]; dh = s0 + phext(tts + 1);
```

The matrix dimensions `m`, `n`, and the number of parameters `np` are obtained from `S`

114b ⟨$\mathtt{S} \mapsto (m, n, n_p)$ 114b⟩≡ (115c 116c 121)
```
[m, n] = size(tts); np = max(max(tts));
```

The transition from the specification of (S) to the specification $(\mathscr{S}(\widehat{p}))$ is

114c ⟨$\mathtt{S} \mapsto \mathbf{S}$ 114c⟩≡ (115c 116c 121d)
```
vec_tts = tts(:); NP = 1:np;
bfs = vec_tts(:, ones(1, np)) == NP(ones(m * n, 1), :);
```

Conversely, for affine structure of the type (S), defined by $\mathbf{S}$, m, and n, the matrix `S` is constructed by

114d ⟨$\mathbf{S} \mapsto \mathtt{S}$ 114d⟩≡
```
tts = reshape(bfs * (1:np)', m, n);
```

In most applications that we consider, the structure $\mathscr{S}$ is linear, so that `s0` is an optional input argument to the solvers with default value being the zero matrix.

114e ⟨*default* `s0` 114e⟩≡ (115c 116c 121)
```
if ~exist('s0', 'var') | isempty(s0), s0 = zeros(m, n); end
```

The default weight matrix W in the approximation criterion is the identity matrix.

114f ⟨*default weight matrix* 114f⟩≡ (115c 116c 121d)
```
if ~exist('w', 'var') | isempty(w), w = eye(np); end
```

Minimization over $\widehat{p}$

In order to solve the optimization problem (SLRA), we change variables

$$\widehat{p} \;\mapsto\; \Delta p = p - \widehat{p}.$$

Then, the constraint is written as a system of linear equations with unknown Δp:

$$\begin{aligned}
R\mathscr{S}(\widehat{p}) = 0 \quad &\Longleftrightarrow \quad R\mathscr{S}(p - \Delta p) = 0 \\
&\Longleftrightarrow \quad R\mathscr{S}(p) - R\mathscr{S}(\Delta p) + RS_0 = 0 \\
&\Longleftrightarrow \quad \operatorname{vec}\big(R\mathscr{S}(\Delta p)\big) = \operatorname{vec}\big(R\mathscr{S}(p)\big) + \operatorname{vec}(RS_0) \\
&\Longleftrightarrow \quad \underbrace{\begin{bmatrix} \operatorname{vec}(RS_1) & \cdots & \operatorname{vec}(RS_{n_p}) \end{bmatrix}}_{G(R)} \Delta p = \underbrace{G(R)p + \operatorname{vec}(RS_0)}_{h(R)} \\
&\Longleftrightarrow \quad G(R)\Delta p = h(R).
\end{aligned}$$

115a ⟨*form $G(R)$ and $h(R)$* 115a⟩≡ (115c)
```
g = reshape(R * reshape(bfs, m, n * np), size(R, 1) * n, np);
h = g * p + vec(R * s0);
```

Assuming that

$$n_p \leq (m-r)n \tag{A}$$

the inner minimimization in (SLRA$_R$) with respect to the new variable Δp is a linear least norm problem

$$M(R) = \min_{\Delta p} \quad \|\Delta p\|_W \quad \text{subject to} \quad G(R)\Delta p = h(R) \tag{LNP}$$

and has the analytic solution

$$\Delta p^*(R) = W^{-1}G^\top(R)\big(G(R)W^{-1}G^\top(R)\big)^{-1}h(R),$$
$$M(R) = \|\Delta p^*(R)\|_W = \sqrt{\Delta p^{*\top}(R)W\Delta p^*(R)}.$$

115b ⟨*solve the least norm problem* 115b⟩≡ (115c)
```
dp = inv_w * g' * (pinv(g * inv_w * g') * h);
```

The function M corresponds to the data–model misfit function in data modeling problems and will be refered to as the structured low-rank approximation misfit.

115c ⟨*Structured low-rank approximation misfit* 115c⟩≡
```
function [M, ph] = misfit_slra(R, tts, p, w, s0, bfs, inv_w)
⟨S ↦ (m,n,np) 114b⟩
⟨default s0 114e⟩, ⟨default weight matrix 114f⟩
if ~exist('bfs') | isempty(bfs), ⟨S ↦ S 114c⟩, end
⟨form G(R) and h(R) 115a⟩
if ~exist('inv_w') | isempty(inv_w), inv_w = inv(w); end
⟨solve the least norm problem 115b⟩
M = sqrt(dp' * w * dp); ph = p - dp;
```
Defines:
misfit_slra, used in chunk 116.

Minimization over R

General purpose constrained optimization methods are used for the outer minimization problem in (SLRA$_R$), *i.e.*, the minimization of M over R, subject to the constraint that R is full row rank. This is a nonconvex optimization problem, so that there is no guarantee that a globally optimal solution is found.

115d ⟨*set optimization solver and options* 115d⟩≡ (116a 185f)
```
prob = optimset();
prob.solver = 'fmincon';
prob.options = optimset('disp', 'off');
```

115e ⟨*call optimization solver* 115e⟩≡ (116a 185f)
```
[x, fval, flag, info] = fmincon(prob); info.M = fval;
```


116a ⟨*nonlinear optimization over R* 116a⟩≡ (116c)
```
⟨set optimization solver and options 115d⟩
prob.x0 = Rini; inv_w = inv(w);
prob.objective = ...
          @(R) misfit_slra(R, tts, p, w, s0, bfs, inv_w);
prob.nonlcon = @(R) deal([], [R * R' - eye(size(R, 1))]);
⟨call optimization solver 115e⟩, R = x;
```
Uses misfit_slra 115c.

If not specified, the initial approximation is computed from a heuristic that ignores the structure and replaces the weighted norm by the Frobenius norm, so that the resulting problem can be solved by the singular value decomposition (function lra).

116b ⟨*default initial approximation* 116b⟩≡ (116c)
```
if ~exist('Rini') | isempty(Rini)
  ph = p; ⟨(S0,S,p) ↦ D = S(p) 114a⟩, Rini = lra(dh, r);
end
```
Uses lra 105a.

The resulting function for affine structured low-rank approximation is:

116c ⟨*Structured low-rank approximation* 116c⟩≡
```
function [R, ph, info] = slra(tts, p, r, w, s0, Rini)
⟨S ↦ (m,n,np) 114b⟩, ⟨S ↦ S 114c⟩
⟨default s0 114e⟩, ⟨default weight matrix 114f⟩
⟨default initial approximation 116b⟩
⟨nonlinear optimization over R 116a⟩
if nargout > 1,
  [M, ph] = misfit_slra(R, tts, p, w, s0, bfs, inv_w);
end
```
Defines:
slra, used in chunks 124a, 137, 247b, and 255.
Uses misfit_slra 115c.

Algorithms for linear time-invariant system identification

Approximate linear time-invariant system identification problems lead to equivalent Hankel structured low-rank approximation problems. Therefore, the function slra, implemented in the previous section can be used in this case. The function slra, however, is designed for general affine structured problems and does not exploit the Hankel structure in the case of linear time-invariant system identification. This section shows an efficient implementation of the variable projection method for structured low-rank approximation in the case of single-input single-output system identification. The structure is exploited by fast methods for factorization of structured matrices. An alternative approach is to use methods for Kalman filtering.

Misfit computation

Consider the misfit between the data w_{d} and a model $\mathscr{B}$, defined as

$$\operatorname{dist}(w_{\mathrm{d}}, \mathscr{B}) := \min_{\widehat{w}} \ \|w_{\mathrm{d}} - \widehat{w}\|_2 \quad \text{subject to} \quad \widehat{w} \in \mathscr{B}. \qquad \text{(misfit } \mathscr{L}_{\mathtt{m},\ell}\text{)}$$

Geometrically, $\operatorname{dist}(w_{\mathrm{d}}, \mathscr{B})$ is the orthogonal projection of w_{d} on $\mathscr{B}$. Assuming that $\mathscr{B}$ is controllable, $\mathscr{B}$ has an image representation $\mathscr{B} = \operatorname{image}\big(P(\sigma)\big)$. Then, the constraint $\widehat{w} \in \mathscr{B}$ becomes $\widehat{w} = P(\sigma)v$, for some latent variable v. Using a matrix representation of the polynomial operator $P(\sigma)$, we have $\widehat{w} = \mathscr{T}_T(P)v$, where $\mathscr{T}_T(P)$ is the block banded Toeplitz matrix (see Exercise 5.1)

$$\mathscr{T}_T(P) := \begin{bmatrix} P_0 & P_1 & \cdots & P_\ell & & \\ & P_0 & P_1 & \cdots & P_\ell & \\ & & \ddots & \ddots & & \ddots \\ & & & P_0 & P_1 & \cdots & P_\ell \end{bmatrix} \in \mathbb{R}^{\mathtt{q}T \times (T+\ell)}. \qquad (\mathscr{T})$$

117a ⟨*Toeplitz matrix constructor* 117a⟩≡

```
function TP = blktoep(P, T)
[q, l1] = size(P); l = l1 - 1; TP = zeros(T * q, T + l);
ind = 1 + (0:T - 1) * q * (T + 1);
for i = 1:q
  for j = 1:l1
    TP(ind + (i - 1) + (j - 1) * (T * q)) = P(i, j);
  end
end
```

Defines:
`blktoep`, used in chunks 117b, 151, 247d, and 255.

The misfit computation problem (misfit $\mathscr{L}_{\mathtt{m},\ell}$) is then equivalent to the standard linear least squares problem

$$M(P) := \operatorname{dist}\big(w_{\mathrm{d}}, \operatorname{image}\big(P(\sigma)\big)\big) = \min_{v} \ \|w_{\mathrm{d}} - \mathscr{T}_T(P)v\|_2. \qquad \text{(misfit}_P\text{)}$$

The solution of (misfit$_P$), implemented in the functions `misfit_siso`, is

$$\widehat{w} = \mathscr{T}_T(P)\big(\mathscr{T}_T^\top(P)\mathscr{T}_T(P)\big)^{-1}\mathscr{T}_T^\top(P)w_{\mathrm{d}}.$$

117b ⟨dist($w_{\mathrm{d}}, \mathscr{B}$) 117b⟩≡

```
function [M, wh] = misfit_siso(w, P)
try, [M, wh] = misfit_siso_efficient(w, P);
catch
  ⟨reshape w and define q, T 27a⟩
  TP = blktoep(P, T);
  wh = reshape(TP * (TP \ w(:)), 2, T);
  M  = norm(w - wh, 'fro');
end
```

Defines:
`misfit_siso`, used in chunks 118b, 119, 146a, and 147b.

Uses `blktoep` 117a.

Solving the least squares problem (misfit$_P$) by the QR or Cholesky factorization without taking into account the structure of $\mathscr{T}_T(P)$ results in a cubic in T computational cost. It turns out that by properly exploiting the structure, algorithms with computational cost that is linear in T can be achieved. One approach is based on of structured linear algebra computational methods (Kailath and Sayed, 1995, 1999), implemented in the SLICOT library (Benner et al, 1999). The function `misfit_siso_efficient` is based on a SLICOT subroutine for Cholesky factorization of positive definite banded Toeplitz matrix. An alternative approach, which also results in methods with linear in T cost are based on the system theoretic interpretation of the problem: equivalence between misfit computation and Kalman smoothing. In this approach, the computation is done by a Riccati type recursion.

Misfit minimization

Consider now the misfit minimization problem

$$\widehat{\mathscr{B}}^* := \arg\min_{\widehat{\mathscr{B}}} \quad \operatorname{dist}(w_{\mathrm{d}}, \widehat{\mathscr{B}}) \quad \text{subject to} \quad \widehat{\mathscr{B}} \in \mathscr{L}_{1,\ell}^2. \qquad \text{(SISO-SYSID)}$$

Using the representation $\mathscr{B} = \operatorname{image}\big(P(\sigma)\big)$, (SISO-SYSID) is equivalent to

$$\text{minimize} \quad \text{over } P \in \mathbb{R}^{\mathtt{q}(\ell+1)\times 1} \quad \operatorname{dist}\big(w_{\mathrm{d}}, \operatorname{image}\big(P(\sigma)\big)\big) \qquad \text{(SYSID}_P\text{)}$$

which is a constrained nonlinear least squares problem.

118a ⟨*Single input single output system identification* 118a⟩≡

```
function [sysh, wh, info] = ident_siso(w, n, sys)
if ~exist('sys', 'var')
  ⟨suboptimal approximate single input single output system identification 118c⟩
else
  ⟨(TF) ↦ P(z) 240c⟩
end
⟨misfit minimization 118b⟩
```

The Optimization Toolbox is used for performing the misfit minimization.

118b ⟨*misfit minimization* 118b⟩≡ (118a) 119▷

```
prob = optimset();
prob.solver = 'fminunc'; prob.x0 = P;
prob.options = optimset('disp', 'off');
prob.objective = @(P) misfit_siso(w, P);
[x, fval, flag, info] = fminunc(prob); info.M = fval; P = x;
```

Uses `misfit_siso` 117b.

The initial approximation is computed from a relaxation ignoring the structure:

118c ⟨*suboptimal approximate single input single output system identification* 118c⟩≡ (118a)

```
R = lra(blkhank(w, n + 1), 2 * n + 1); ⟨R(z) ↦ P(z) 240e⟩
```

Uses `blkhank` 26a and `lra` 105a.

The solution obtained by the optimization solver is an image representation of a (locally) optimal approximate model $\widehat{\mathscr{B}}^*$. The image representation is converted to a transfer function representation in order to make the obtained model compatible with other software packages for linear time-invariant systems identification, analysis, and design that accept transfer function representation.

119 ⟨*misfit minimization* 118b⟩+≡ (118a) ◁118b

```
⟨P(z) ↦ (TF) 240d⟩ sysh = sys;
if nargout > 1, [M, wh] = misfit_siso(w, P); end
```

Uses misfit_siso 117b.

4.3 Nuclear norm heuristic

Replacing the rank constraint by a constraint on the nuclear norm leads to a convex optimization problem—a semidefinite program. A semidefinite program, in turn, can be solved by existing algorithms with provable convergence properties and readily available high quality software implementation. Apart from theoretical justification and easy implementation in practice, formulating the problem as a semidefinite program has the advantage of flexibility. For example, adding affine inequality constraints in the data modeling problem preserves the convexity.

A disadvantage of using the nuclear norm heuristic is the fact that the number of optimization variables in the semidefinite optimization problem depends quadratically on the number of data points in the data modeling problem. This makes methods based on the nuclear norm heuristic impractical for problems with more than a few hundreds of data points, which are "small size" data modeling problems.

This section describes a convex relaxation method for affine structured low-rank approximation, based on the nuclear norm. An implementation of the method in a literate programming style is presented. The nuclear norm relaxation method is compared with alternative methods: the singular value decomposition in case of an unstructured problem and Kung's method in case of a Hankel structured problem.

4.3.1 Nuclear norm heuristics for structured low-rank approximation

First, a semidefinite program equivalent to regularized nuclear norm minimization is shown. Then, the result is applied to structured low-rank approximation problem, by relaxing the rank constraint to a constraint on the nuclear norm.

Regularized nuclear norm minimization

The nuclear norm of a matrix is the sum of the matrix's singular values

$$\|M\|_* = \text{sum of the singular values of } M. \tag{NN}$$

Consider the mapping $\mathscr{S}$ from a structure parameter space $\mathbb{R}^{n_p}$ to the set of matrices $\mathbb{R}^{m\times n}$ (see ($\mathscr{S}(\widehat{p})$), on page 113). Regularized nuclear norm minimization

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{p} \quad \|\mathscr{S}(\widehat{p})\|_* + \gamma\|p-\widehat{p}\|_2 \\ \text{subject to} & G\widehat{p} \leq h \end{array} \tag{NNM}$$

is a convex optimization problem and can be solved globally and efficiently. Since,

$$\|\widehat{D}\|_* < \mu \quad \Longleftrightarrow \quad \frac{1}{2}\big(\operatorname{trace}(U)+\operatorname{trace}(V)\big) < \mu \quad \text{and} \quad \begin{bmatrix} U & \widehat{D}^\top \\ \widehat{D} & V \end{bmatrix} \succeq 0,$$

we obtain an equivalent semidefinite programming problem

$$\begin{array}{lll} \text{minimize} & \text{over } \widehat{p}, U, V, \text{ and } \nu & \frac{1}{2}\big(\operatorname{trace}(U)+\operatorname{trace}(V)\big) + \gamma\nu \\ \text{subject to} & \begin{bmatrix} U & \mathscr{S}^\top(\widehat{p}) \\ \mathscr{S}(\widehat{p}) & V \end{bmatrix} \succeq 0, & \|p-\widehat{p}\|_2 < \nu, \quad \text{and} \quad G\widehat{p} \leq h. \end{array} \tag{NNM'}$$

Structured low-rank approximation

Consider the affine structured low-rank approximation problem (SLRA). Due to the rank constraint, this problem is non-convex. Replacing the rank constraint by a constraint on the nuclear norm results in a convex relaxation of (SLRA)

$$\text{minimize} \quad \text{over } \widehat{p} \quad \|p-\widehat{p}\|_2 \quad \text{subject to} \quad \|\mathscr{S}(\widehat{p})\|_* \leq \mu. \tag{RSLRA}$$

The motivation for using the nuclear norm heuristic in solving (SLRA) is that choosing an appropriate value for the bound on the nuclear norm μ results in a solution $\mathscr{S}(\widehat{p})$ that satisfies the constraint $\operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r$. Moreover, the nuclear norm has the theoretical justification as being the tightest relaxation of the rank.

Problem (RSLRA) can also be written in the equivalent unconstrained form

$$\text{minimize} \quad \text{over } \widehat{p} \quad \|\mathscr{S}(\widehat{p})\|_* + \gamma\|p-\widehat{p}\|_2, \tag{RSLRA'}$$

where γ is a regularization parameter. It is related to the parameter μ in (RSLRA). The formulation (RSLRA') of the relaxed affine structured low-rank approximation problem (RSLRA) is a regularized nuclear norm minimization problem (NNM').

4.3.2 *Literate programs*

This section presents an implementation of the nuclear norm minimization method for affine structured low-rank approximation. First, the implementation of the regularized nuclear norm minimization method is presented. Then, the method is used for solving the Problem SLRA. A nontrivial subproblem is the automatic selection of the regularization parameter γ. For this purpose we use a bisection method.

Regularized nuclear norm minimization

The CVX package is used in order to automatically translate problem (NNM') into a standard convex optimization problem and solve it by existing optimization solvers.

121a ⟨*Regularized nuclear norm minimization* 121a⟩≡ 121b▷
```
function [ph, info] = nucnrm(tts, p, gamma, nrm, w, s0, g, h)
⟨S ↦ (m,n,n_p) 114b⟩, ⟨default s0 114e⟩
```
Defines:
nucnrm, used in chunk 122a.

The code consists of definition of the optimization variables:

121b ⟨*Regularized nuclear norm minimization* 121a⟩+≡ ◁121a 121c▷
```
cvx_begin sdp; cvx_quiet(true);
  variable U(n, n) symmetric;
  variable V(m, m) symmetric;
  variables ph(np) nu;
  ⟨(S_0,S,p̂) ↦ D̂ = 𝒮(p̂) 114a⟩
```

and direct rewriting of the cost function and constraints of (NNM') in CVX syntax:

121c ⟨*Regularized nuclear norm minimization* 121a⟩+≡ ◁121b
```
  minimize( trace(U) / 2 + trace(V) / 2 + gamma * nu );
  subject to
    [U dh'; dh V] > 0;
    norm(w * (p - ph), nrm) < nu;
    if (nargin > 6) & ~isempty(g), g * ph < h; end
cvx_end
```

The `w` argument specifies the norm ($\|\cdot\|_v$) and is equal to 1, 2, `inf`, or (in the case of a weighted 2-norm) a $n_p \times n_p$ positive semidefinite matrix. The `info` output variable of the function `nucnrm` is a structure with fields `optval` (the optimal value) and `status` (a string indicating the convergence status).

Structured low-rank approximation

The following function finds suboptimal solution of the structured low-rank approximation problem by solving the relaxation problem (RSLRA'). Affine structures of the type (S) are considered.

121d ⟨*Structured low-rank approximation using the nuclear norm* 121d⟩≡ 122b▷
```
function [ph, gamma] = slra_nn(tts, p, r, gamma, nrm, w, s0)
```



```
⟨S ↦ (m,n,n_p) 114b⟩, ⟨S ↦ S 114c⟩, ⟨default s0 114e⟩, ⟨default weight matrix 114f⟩
if ~exist('gamma', 'var'), gamma = []; end % default gamma
if ~exist('nrm', 'var'),   nrm = 2;    end % default norm
```
Defines:
slra_nn, used in chunk 123e.

If a parameter γ is supplied, the convex relaxation (RSLRA') is completely specified and can be solved by a call to `nucnrm`.

122a ⟨*solve the convex relaxation (RSLRA') for given γ parameter* 122a⟩≡ (122)
```
ph = nucnrm(tts, p, gamma, nrm, w, s0);
```
Uses nucnrm 121a.

Large values of γ lead to solutions $\widehat{p}$ with small approximation error $\|p-\widehat{p}\|_W$, but high rank. Vice verse, small values of γ lead to solutions $\widehat{p}$ with low-rank, but high approximation error $\|p-\widehat{p}\|_W$. If not given as an input argument, a value of γ, which gives an approximation matrix $\mathscr{S}(\widehat{p})$ with numerical rank r is computed by bisection on an a priori given interval $[\gamma_{\min}, \gamma_{\max}]$.

122b ⟨*Structured low-rank approximation using the nuclear norm* 121d⟩+≡ ◁121d
```
if ~isempty(gamma) & isscalar(gamma)
  ⟨solve the convex relaxation (RSLRA') for given γ parameter 122a⟩
else
  if ~isempty(gamma)
    gamma_min = gamma(1); gamma_max = gamma(2);
  else
    gamma_min = 0; gamma_max = 100;
  end
  ⟨parameters of the bisection algorithm 123a⟩
  ⟨bisection on γ 122c⟩
end
```

On each iteration of the bisection algorithm, the convex relaxation (RSLRA') is solved for γ equal to the mid point $(\gamma_{\min}+\gamma_{\max})/2$ of the interval and the numerical rank of the approximation $\mathscr{S}(\widehat{p})$ is checked by computing the singular value decomposition. If the numerical rank is higher than r, $\gamma_{\max}$ is redefined to the mid point, so that the search continuous on a smaller value of γ (which has the potential of decreasing the rank). Otherwise, $\gamma_{\min}$ is redefined to be the mid point, so that the search continuous on a larger value of γ (which has the potential of increasing the rank). The search continuous till the interval $[\gamma_{\min}, \gamma_{\max}]$ is sufficiently small or a predefined maximum number of iterations is exceeded.

122c ⟨*bisection on γ* 122c⟩≡ (122b)
```
iter = 0;
while ((gamma_max - gamma_min) / gamma_max > rel_gamma_tol) ...
      & (iter < maxiter)
  gamma = (gamma_min + gamma_max) / 2;
  ⟨solve the convex relaxation (RSLRA') for given γ parameter 122a⟩
  sv = svd(ph(tts));
  if (sv(r + 1) / sv(1) > rel_rank_tol) ...
    & (sv(1) > abs_rank_tol)
    gamma_max = gamma;
  else
    gamma_min = gamma;
```

```
    end
    iter = iter + 1;
  end
```

The rank test and the interval width test involve a priori set tolerances.

123a ⟨*parameters of the bisection algorithm* 123a⟩≡ (122b)
```
rel_rank_tol  = 1e-6; abs_rank_tol = 1e-6;
rel_gamma_tol = 1e-5; maxiter      = 20;
```

Examples

In this section, we test the approximation accuracy of the nuclear norm heuristic, implemented in the function `slra_nn`, for solving randomly generated unstructured and Hankel structured low-rank approximation problems. A rank deficient "true" data matrix is constructed, where the rank $\bar{r}$ is a simulation parameter. In the case of a Hankel structure, the "true" structure parameter vector $\bar{p}$ is generated as the impulse response (skipping the first sample) of a discrete-time linear time-invariant system of order $\bar{r}$. This ensures that the "true" Hankel structured data matrix $\mathscr{S}(\bar{p}) = \mathscr{H}_L(\bar{p})$, $L > \bar{r}$ has the desired rank $\bar{r}$.

123b ⟨*Test* `slra_nn` 123b⟩≡ 123c▷
```
⟨initialize the random number generator 25⟩
if strcmp(structure, 'hankel')
  np = m + n - 1; tts = hankel(1:m, m:np);
  p0 = impulse(drss(r0), np + 1); p0 = p0(2:end);
```
Defines:
`test_slra_nn`, used in chunk 124.

In the unstructured case, the data matrix is generated by multiplication of random $m \times \bar{r}$ and $\bar{r} \times n$ factors of a rank revealing factorization of the data matrix $\mathscr{S}(\bar{p})$.

123c ⟨*Test* `slra_nn` 123b⟩+≡ ◁123b 123d▷
```
else % unstructured
  np = m * n; tts = reshape(1:np, m, n);
  p0 = rand(m, r0) * rand(r0, n); p0 = p0(:);
end
```

The data parameter p, passed to the low-rank approximation function, is a noisy corrupted version of the true data parameter $\bar{p}$, where the additive noise is zero mean Gaussian. The noise standard deviation is a simulation parameter.

123d ⟨*Test* `slra_nn` 123b⟩+≡ ◁123c 123e▷
```
e = randn(np, 1); p = p0 + nl * e / norm(e) * norm(p0);
```

The results obtained by `slra_nn`

123e ⟨*Test* `slra_nn` 123b⟩+≡ ◁123d 124a▷
```
[ph, gamma] = slra_nn(tts, p, r0);
```
Uses `slra_nn` 121d.

are compared with the ones of alternative methods by checking the singular values of $\mathscr{S}(\hat{p})$, indicating the numerical rank, and the fitting error $\|p - \hat{p}\|_2$.

In the case of a Hankel structure, the alternative methods, being used, is Kung's method (implemented in the function `h2ss`) and the method based on local optimization (implemented in the function `slra`).

124a ⟨*Test* `slra_nn` 123b⟩+≡ ◁123e 124b▷
```
if strcmp(structure, 'hankel')
  sysh = h2ss([0; p], r0);
  ph2 = impulse(sysh, np + 1); ph2 = ph2(2:end);
  tts_ = hankel(1:(r0 + 1), (r0 + 1):np);
  [Rh, ph3] = slra(tts_, p, r0);
  sv = [svd(p(tts)) svd(ph(tts)) svd(ph2(tts)) svd(ph3(tts))]
  cost = [norm(p - p)   norm(p - ph) ...
          norm(p - ph2) norm(p - ph3)]
```
Uses `h2ss` 74c and `slra` 116c.

In the unstructured case, the alternative method is basic low-rank approximation (`lra`), which gives globally optimal result in this setup.

124b ⟨*Test* `slra_nn` 123b⟩+≡ ◁124a
```
else % unstructured
  [Rh, Ph, dh] = lra(p(tts)', r0); dh = dh';
  sv = [svd(p(tts)) svd(ph(tts)) svd(dh)]
  cost = [norm(p - p) norm(p - ph) norm(p - dh(:))]
end
```
Uses `lra` 105a.

The first test example is a 5×5, unstructured matrix, whose true value has rank 3.

124c ⟨*Test* `slra_nn` *on unstructured problem* 124c⟩≡
```
m = 5; n = 5; r0 = 3; nl = 1; structure = 'unstructured';
test_slra_nn
```
Uses `test_slra_nn` 123b.

The result shows that the numerical rank (with tolerance 10^{-5}) of both approximations is equal to the specified rank but the approximation error achieved by the nuclear norm heuristic is about two times bigger than the approximation error of the optimal approximation. The corresponding trade-off curve is shown in Figure 4.2.

The second test example is a 5×5, Hankel structured matrix of rank 3. In this case, the subspace method `h2ss` and the local optimization based method `slra` are also heuristics for solving the Hankel structured low-rank approximation problem and give, respectively, suboptimal and locally optimal results.

124d ⟨*Test* `slra_nn` *on Hankel structured problem* 124d⟩≡
```
m = 5; n = 5; r0 = 3; nl = 0.05; structure = 'hankel';
test_slra_nn
```
Uses `test_slra_nn` 123b.

The result shows that the approximations have numerical rank matching the specification but `slra_nn` gives about two times bigger approximation error than Kung's method. The corresponding trade-off curve is shown in Figure 4.2.

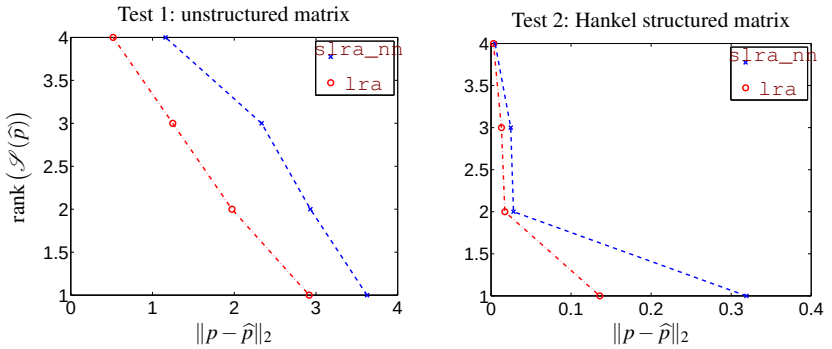


Fig. 4.2: In experiments with unstructured (left plot) and Hankel structured (right plot) matrices, the low-rank approximation, computed by the singular value decomposition, outperforms the nuclear norm heuristic. The set of feasible points in the accuracy vs complexity plane for the nuclear norm heuristic is included into the set of feasible points for the singular value decomposition. For unstructured matrix, the singular value decomposition yields a globally optimal result. In case of a Hankel structured matrix, however, the singular value decomposition is like the nuclear norm a heuristic for solving the structured low-rank approximation problem.

4.4 Missing data estimation

This section generalizes the variable projection method, presented in Section 4.2.3, to low-rank approximation with missing data. The inner minimization is a singular linear least norm problem and admits an analytic solution. The outer problem is an optimization on a manifold. Two approaches are: 1) minimization subject to quadratic equality constraints and 2) unconstrained minimization of a regularized cost function. The method is furthermore generalized to weighted cost functions.

4.4.1 Problem formulation

First, we generalize the structured low-rank approximation problem (SLRA) to structured low-rank matrix approximation and *completion*. Recall from Section 3.4 the notation `NaN` for a missing value, $\mathbb{R}_{\mathrm{e}}$ for the extended set $\mathbb{R} \cup \texttt{NaN}$, $\mathscr{I}_{\mathrm{g}}$ for the indices of the given data elements, and $\mathscr{I}_{\mathrm{m}}$ for the indices of the missing data elements. Let also n_{m} and n_{g} be the number of missing and given elements.

The considered low-rank approximation problem is: Given a data vector $p \in \mathbb{R}_{\mathrm{e}}^{n_p}$,

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} \quad \|p_{\mathscr{I}_{\mathrm{g}}} - \widehat{p}_{\mathscr{I}_{\mathrm{g}}}\|_2^2 \\ \text{subject to} & \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r, \end{array} \tag{SLRAC}$$

where $\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m \times n}$ is an affine matrix structure ($\mathscr{S}(\widehat{p})$). Without loss of generality, assume $r < m \leq n$. Using the kernel representation of the rank constraint (rank_R), the following equivalent problem to (SLRAC) is obtained

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} \text{ and } R \in \mathbb{R}^{(m-r)\times m} \quad \|p_{\mathscr{I}_{\mathrm{g}}} - \widehat{p}_{\mathscr{I}_{\mathrm{g}}}\|_2^2 \\ \text{subject to} & R\mathscr{S}(\widehat{p}) = 0 \quad \text{and} \quad R \text{ has full row rank}, \end{array}$$

which is a double minimization over the parameters R and $\widehat{p}$

$$\begin{array}{ll} \text{minimize} & \text{over } R \in \mathbb{R}^{(m-r)\times m} \quad M(R) \\ \text{subject to} & R \text{ has full row rank}, \end{array} \tag{SLRAC$_R$}$$

where

$$M(R) := \min_{\widehat{p}} \|p_{\mathscr{I}_{\mathrm{g}}} - \widehat{p}_{\mathscr{I}_{\mathrm{g}}}\|_2^2 \quad \text{subject to} \quad R\mathscr{S}(\widehat{p}) = 0. \tag{M}$$

The evaluation of the cost function M, *i.e.*, solving (M) for a given value of R, is refered to as the *inner minimization problem*. This problem is solved analytically. The remaining problem of minimizing M over R is refered to as the *outer minimization problem*, which is optimization on a manifold and has no analytic solution.

4.4.2 Analytical solution of the inner minimization problem

In this section, we consider the inner minimization problem (M).

Problem 4.17. Given affine structure $\mathscr{S}$, structure parameter vector with missing values $p \in \mathbb{R}_{\mathrm{e}}^{n_p}$, and a kernel parameter $R \in \mathbb{R}^{(m-r)\times m}$, evaluate the cost function $M(R)$, defined in (M), and find a vector $\widehat{p} \in \mathbb{R}^{n_p}$ that attains the minimum.

For a given structure $\mathscr{S}$ and $R \in \mathbb{R}^{(m-r)\times m}$, we define the matrix

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

Theorem 4.18 (Markovsky and Usevich (2013)). *Under the assumptions:*

1. $G_{:,\mathscr{I}_m}$ *is full column rank,*
2. $1 \leq (m-r)n - n_m \leq n_{\mathrm{g}}$, *and*
3. $\bar{G} := G_{:,\mathscr{I}_m}^{\perp} G_{:,\mathscr{I}_{\mathrm{g}}}$ *is full row rank,*

Problem 4.17 has a unique global minimum

$$\begin{aligned} \widehat{p}_{\mathscr{I}_{\mathrm{g}}} &= p_{\mathscr{I}_{\mathrm{g}}} - \bar{G}^{\top}\big(\bar{G}\bar{G}^{\top}\big)^{-1} s \\ \widehat{p}_{\mathscr{I}_m} &= -G_{:,\mathscr{I}_m}^{+}\big(\operatorname{vec}(RS_0) + G_{:,\mathscr{I}_{\mathrm{g}}}\widehat{p}_{\mathscr{I}_{\mathrm{g}}}\big), \end{aligned} \tag{$\widehat{p}$}$$

with objective function value

$$M(R) = s^\top \left(\bar{G}\bar{G}^\top\right)^{-1} s, \quad \text{where } s := \bar{G} p_{\mathscr{I}_g} + G^\perp_{:,\mathscr{I}_m} \operatorname{vec}(RS_0). \tag{M}$$

Note that Assumption 1 of Theorem 4.18 is stronger (implies) assumption (A), used on page 115 for the derivation of the variable projection methods in the case of an affine structured low-rank approximation without missing data.

Lemma 4.19 (Generalized least norm problem). *Consider the problem*

$$f = \min_{x,y} \quad \|x\|_2^2 \quad \text{subject to} \quad Ax + By = c, \tag{GLN}$$

with $A \in \mathbb{R}^{m\times n_x}$, $B \in \mathbb{R}^{m\times n_y}$, and $c \in \mathbb{R}^m$. Under the following assumptions:

1. *B is full column rank,*
2. *$1 \leq m - n_y \leq n_x$, and*
3. *$\bar{A} := B^\perp A$ is full row rank,*

problem (GLN) has a unique solution

$$\begin{aligned} f &= c^\top (B^\perp)^\top \left(\bar{A}\bar{A}^\top\right)^{-1} B^\perp c, \\ x &= \bar{A}^\top \left(\bar{A}\bar{A}^\top\right)^{-1} B^\perp c \quad \text{and} \quad y = B^+(c - Ax). \end{aligned} \tag{SOL}$$

Assumption 1 of Lemma 4.19 is a necessary condition for uniqueness of the solution. Relaxing assumption 1 implies that any vector in the affine space

$$\mathscr{Y} = B^+(c - Ax) + \operatorname{null}(B)$$

is also a solution to the generalized least norm problem (GLN). Assumption 2 of Lemma 4.19 ensures that (GLN) is a least norm problem and has a nontrivial solution. In the case $m = n_y$, the problem has a trivial solution $f = 0$. In the case $m - n_y > n_x$, the problem generically has no solution. Assumption 3 is also required for uniqueness of the solution. It can also be relaxed, making y nonunique. In the case of unstructured matrix, assumption 2 of Theorem 4.18 reduces to $n_\mathrm{m} < (m-r)n$, *cf.*, assumption (A) on page 115.

The generalized least norm problem (GLN) is related to the following weighted least norm problem with a singular positive semidefinite weight matrix W

$$\min_z \; z^\top W z \quad \text{subject to} \quad Dz = c,$$

In order to show this, consider the change of variables $\bar{z} = T^{-1}z$, where T is a nonsingular matrix. We obtain the equivalent problem

$$\min_{\bar{z}} \; \bar{z}^\top T^\top W T \bar{z} \quad \text{subject to} \quad DT\bar{z} = c.$$

There exists a nonsingular matrix T, such that $T^\top W T = \left[\begin{smallmatrix} I_{n_x} & \\ & 0 \end{smallmatrix}\right]$. Partitioning $\bar{z}$ and $\bar{D} := DT^{-1}$ conformably as $\bar{z} = \left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]$ and $\bar{D} = \begin{bmatrix} A & B \end{bmatrix}$, we obtain problem (GLN).

4.4.3 Outer minimization problem

The outer minimization problem (SLRAC_R) is a nonlinear least squares problem. In order to apply standard optimization methods, however, we need to replace first the constraint "R full row rank" with an equivalent equality constraint, *e.g.*, $RR^\top = I_{m-r}$, *cf.*, (SLRA_R) on page 112. This leads to the nonlinear least squares problem with quadratic equality constraint:

$$\text{minimize} \quad \text{over } R \in \mathbb{R}^{(m-r)\times m} \quad M(R) \quad \text{subject to} \quad RR^\top = I_{m-r}. \tag{SLRAC'_R}$$

(SLRAC'_R) is optimization on a Stiefel manifold, see (Absil et al, 2008), and can be solved by the methods implemented in the Manopt package (Boumal et al, 2014).

Next, we consider an alternative penalty method. We reformulation (SLRAC'_R) as a regularized unconstrained nonlinear least squares problem by adding the regularization term $\gamma\|RR^\top - I_{m-r}\|_\mathrm{F}^2$ to the cost function

$$\text{minimize} \quad \text{over } R \in \mathbb{R}^{(m-r)\times m} \quad M(R) + \gamma\|RR^\top - I_{m-r}\|_\mathrm{F}^2. \tag{SLRAC''_R}$$

The parameter γ should be chosen "large enough" in order to enforce the constraint $RR^\top = I_{m-r}$. As shown in the following theorem $\gamma = \|p_{\mathscr{I}_\mathrm{g}}\|_2^2$ is a "sufficiently large" value for linearly structured problems.

Theorem 4.20 (Markovsky and Usevich (2013)). *Let $M : \mathbb{R}^{(m-r)\times m} \to \mathbb{R}_+$ be a homogeneous function,* i.e., *$M(R) = M(TR)$, for any R and a nonsingular $m \times m$ matrix T. Assume that γ satisfies $\gamma > \min_{\{R \mid \operatorname{rank}(R) = m-r\}} M(R)$. Then, the solutions of problem (SLRAC''_R) coincide with the solutions of (SLRAC'_R).*

The value $\gamma = \max_{R\in\mathscr{R}_f} M(R)$ always satisfies the assumption of Theorem 4.20. In particular, for a linear structure $\mathscr{S}$, it is sufficient to take $\gamma = \|p_{\mathscr{I}_\mathrm{g}}\|_2^2$.

Solving (SLRAC_R) by local minimization requires an initial approximation for the parameter R, *i.e.*, a suboptimal solution. Such a solution can be computed from a heuristic that ignores the data matrix structure $\mathscr{S}$ and fills in the missing values with initial estimates. Rigorous analysis of the missing values imputation question is done in (Keshavan et al, 2010). Theorem 1.1 of Keshavan et al (2010) gives theoretical justification for the zero imputation in the case of unstructured $\mathscr{S}$. The resulting unstructured low-rank approximation problem can then be solved analytically in terms of the singular value decomposition (`lra_md` on page 107).

An efficient method for evaluation of the cost function and its derivatives in the case of mosaic-Hankel matrix structure is presented in (Usevich and Markovsky, 2014). The method for general affine structure (Markovsky and Usevich, 2013) and the efficient methods of Usevich and Markovsky (2014) are implemented in MATLAB (using Optimization Toolbox) and in C++ (using the Levenberg-Marquardt algorithm (Marquardt, 1963) from the GNU Scientific Library (Galassi et al, 2017)). Description of the software and overview of its applications is given in (Markovsky and Usevich, 2014).

4.4.4 Weighted approximation

The method for structured low-rank matrix approximation and completion, presented in Sections 4.4.2 and 4.4.3, is generalized in this section to the following weighted structured low-rank matrix approximation and completion problem:

$$\begin{array}{lll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} & (p_{\mathscr{I}_\text{g}} - \widehat{p}_{\mathscr{I}_\text{g}})^\top W_\text{g} (p_{\mathscr{I}_\text{g}} - \widehat{p}_{\mathscr{I}_\text{g}}) \\ \text{subject to} & \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r, & \end{array} \tag{WSLRAC}$$

where W_g is a positive definite matrix. To see this, consider the change of variables

$$p'_{\mathscr{I}_\text{g}} = W_\text{g}^{1/2} p_{\mathscr{I}_\text{g}} \quad \text{and} \quad \widehat{p}'_{\mathscr{I}_\text{g}} = W_\text{g}^{1/2} \widehat{p}_{\mathscr{I}_\text{g}}. \tag{$p \mapsto p'$}$$

It reduces (WSLRAC) to an equivalent unweighted problem (SLRAC). We have

$$\mathscr{S}(\widehat{p}) = S_0 + \operatorname{vec}^{-1}(\mathbf{S}\widehat{p}), \quad \text{where } \mathbf{S} := \begin{bmatrix} \operatorname{vec}(S_1) & \cdots & \operatorname{vec}(S_{n_p}) \end{bmatrix} \in \mathbb{R}^{mn \times n_p}. \tag{$\mathbf{S}$}$$

The structure $\mathscr{S}'$ of the equivalent problem is defined by the matrices S_0 and

$$\mathbf{S}' = \begin{bmatrix} \operatorname{vec}(S'_1) & \cdots & \operatorname{vec}(S'_{n_p}) \end{bmatrix}, \quad \text{where}$$
$$\mathbf{S}'_{:,\mathscr{I}_\text{g}} = \mathbf{S}_{:,\mathscr{I}_\text{g}} W_\text{g}^{-1/2} \text{ and } \mathbf{S}'_{:,\mathscr{I}_\text{m}} = \mathbf{S}_{:,\mathscr{I}_\text{m}}. \tag{$\mathscr{S} \mapsto \mathscr{S}'$}$$

Therefore, problem (WSLRAC) is solved by:

1. preprocessing the data p and the structure $\mathscr{S}$, as in ($p \mapsto p'$) and ($\mathscr{S} \mapsto \mathscr{S}'$),
2. solving the equivalent unweighted problem with structure parameter vector p', structure specification $\mathscr{S}'$, and rank specification r, and
3. postprocessing the solution $\widehat{p}'$, obtained in step 2, in order to obtain the solution $\widehat{p}_{\mathscr{I}_\text{g}} = W_\text{g}^{-1/2} \widehat{p}'_{\mathscr{I}_\text{g}}$ of the original problem.

Using the transformation ($p \mapsto p'$), ($\mathscr{S} \mapsto \mathscr{S}'$) and the solution (M) of (SLRAC), we obtain the following explicit expression for the cost function of (WSLRAC)

$$M(R) = \big(\bar{G} p_{\mathscr{I}_\text{g}} - G^\perp_{:,\mathscr{I}_\text{m}} \operatorname{vec}(RS_0)\big)^\top W_\text{g}^{-1} \bar{G}^\top \big(\bar{G} W_\text{g}^{-1} \bar{G}^\top\big)^{-1}$$
$$\bar{G} W_\text{g}^{-1} \big(\bar{G} p_{\mathscr{I}_\text{g}} - G^\perp_{:,\mathscr{I}_\text{m}} \operatorname{vec}(RS_0)\big), \tag{M_W}$$

where $\bar{G} = G^\perp_{:,\mathscr{I}_\text{m}} G_{:,\mathscr{I}_\text{g}}$ and G is defined in (G).

In the case of a diagonal weight matrix, $W_\text{g} = \operatorname{diag}(w_1, \ldots, w_{n_\text{g}})$, with $w_i > 0$, an infinite weight $w_j = \infty$ specifies a fixed parameter value $\widehat{p}_j = p_j$. A problem with infinite weights is equivalent to structured low-rank approximation with fixed parameters assigned to the constant term S_0 of the structure specification. Let $\mathscr{I}_\text{f}$ be the set of indices of the fixed structure parameters and $\overline{\mathscr{I}}_\text{f}$ its complement

$$\mathscr{I}_\text{f} = \{\, j \in \{1, \ldots, n_p\} \mid \widehat{p}_j = p_j \,\} \quad \text{and} \quad \overline{\mathscr{I}}_\text{f} = \{\, j \in \{1, \ldots, n_p\} \mid j \notin \mathscr{I}_\text{f} \,\}.$$

The equivalent problem has structure, defined by

$$\mathscr{S}'(\widehat{p}') = S_0 + \sum_{i \in \mathscr{I}_\text{f}} S_i p_i + \sum_{i \in \overline{\mathscr{I}}_\text{f}} S_i \widehat{p}_i, \quad \text{where } \widehat{p}' := \widehat{p}|_{\overline{\mathscr{I}}_\text{f}}.$$

The estimate $\widehat{p}$ is recovered from $\widehat{p}'$ by $\widehat{p}|_{\overline{\mathscr{I}}_\text{f}} = \widehat{p}'$ and $\widehat{p}|_{\mathscr{I}_\text{f}} = p|_{\mathscr{I}_\text{f}}$.

An interesting observation is that the structured low-rank approximation problems (SLRAC) can be solved as an equivalent weighted unstructured problem. Consider an instance of problem (SLRAC), refered to as problem P1, with structure $\mathscr{S} = \mathscr{S}_1$ and an instance of problem (WSLRAC), refer to as problem P2, with unstructured correction ($\mathscr{S}_2 = \operatorname{vec}^{-1}$, $n_{p_2} = mn$) and weight matrix $W_2^{-1} = \mathbf{S_1} \mathbf{S_1}^\top$.

It can be verified by inspection that the cost functions (M) and (M_W) of problems P1 and P2, respectively, coincide. However, the $mn \times mn$ weight matrix W_2 is singular ($\operatorname{rank}(W_2)$ is equal to the number of structure parameters of problem P1, which is less than mn). In the derivation of the cost function (M_W), it is assumed that W_g is positive definite, so that minimization of (M_W) is not equivalent to problem P2.

4.5 Notes and references

Equivalence of low-rank approximation and principal component analysis

The principal component analysis method for dimensionality reduction is usually introduced in a stochastic setting as maximization of the variance of the projected data on a subspace. Computationally, however, the problem of finding the principal components and the corresponding principal vectors is an eigenvalue/eigenvector decomposition problem for the sample covariance matrix

$$\Psi(\mathscr{D}) := \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix} \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}^\top.$$

From this algorithmic point of view, the equivalence of principal component analysis and low-rank approximation problem is a basic linear algebra fact: the space spanned by the first $\mathtt{m}$ principal vectors of the data matrix $\mathscr{D}$ coincides with the model $\widehat{\mathscr{B}} = \operatorname{span}(\widehat{\mathscr{D}})$, where $\widehat{\mathscr{D}}$ is a solution of a low-rank approximation problem.

Weighted low-rank approximation

Methods for solving weighted low-rank approximation problems with nonsingular weight matrix have been considered in the literature under different names:

- Riemannian singular value decomposition (De Moor, 1993),
- maximum likelihood principal component analysis (Wentzell et al, 1997),
- weighted low-rank approximation (Manton et al, 2003), and
- weighted total least squares (Markovsky et al, 2005).

The Riemannian singular value decomposition method of De Moor (1993) resembling the inverse power iteration algorithm. It has no proven convergence properties. The maximum likelihood principal component analysis methods of Schuermans et al (2005); Wentzell et al (1997) are developed for applications in chemometrics. They are alternating projections algorithms and can solve general weighted low-rank approximation problems. They are globally convergent with linear convergence rate. The methods may be slow when the $r+1$st and the rth singular values of the data matrix D are close to each other. In the unweighted case, this situation corresponds to lack of uniqueness of the solution, *cf.*, Theorem 4.5. The convergence properties of alternating projections algorithms are studied in (Kiers, 2002; Krijnen, 2006). Manton et al (2003) treat the problem as an optimization over a Grassman manifold and propose steepest decent and Newton type algorithms. The least squares nature of the problem is not exploited in this work and the proposed algorithms are not globally convergent.

Software for mosaic-Hankel structured low-rank approximation

The mosaic-Hankel matrix structure is a generalization of the block-Hankel structure. It is a block matrix, which blocks are Hankel matrices. In the context of linear time-invariant system identification, mosaic-Hankel matrices appear when the data consists of multiple trajectories. If all trajectories have the same number of samples the problem can be solved via block-Hankel structured low-rank approximation, however, the general case requires mosaic-Hankel structure.

An efficient local optimization based software package for mosaic Hankel structure is presented in (Markovsky and Usevich, 2014). In addition, the software allows specification of a *weighted 2-norm* approximation criterion, *fixed elements* in the approximating matrix, *missing elements* in the data matrix, and *linear constraints* on an approximating matrix's left kernel basis. The computational method is based on the variable projection principle and is presented in (Usevich and Markovsky, 2014).

The software is available from: `http://slra.github.io/`.

Nuclear norm heuristic

The nuclear norm relaxation for solving rank minimization problems (RM) was proposed in (Fazel, 2002). It is a generalization of the ℓ_1-norm heuristic from sparse vector approximation problems to low-rank matrix approximation problems. The CVX package is developed and maintained by Grant and Boyd (2017). A Python version is also available (Dahl and Vandenberghe, 2010). The computational engines of CVX are SDPT3 and SeDuMi. These solvers can deal with a medium-size structured low-rank approximation problems ($n_p < 100$). An efficient interior point method for solving (NNM), which can deal with up to 500 parameters, is presented in (Liu and Vandenberghe, 2009). The method is implemented in Python.

System identification with missing data

There are three main approaches for identification with missing data:

- modification of the classical prediction error methods,
- methods developed in the structure low-rank approximation setting, and
- convex relaxation methods based on the nuclear norm heuristic.

The approach using the prediction error setting (Wallin and Hansson, 2014) employs standard nonlinear local optimization methods. These methods require initial values for the optimization variables (model parameters and missing values) and the results depend on their closeness to a "good" locally optimal solution. Similar in spirit but different in implementation details are the methods developed in the structure low-rank approximation setting (Markovsky, 2013; Markovsky and Usevich, 2013).

The approach based on relaxation of the problem to a convex one by using the nuclear norm in lieu of the rank is proposed in (Liu and Vandenberghe, 2009). System identification with missing data is handled by 1) completion of the missing data using the nuclear norm heuristic (this step requires solution of a convex optimization problem), and 2) identification of a model parameter from the completed sequence using classical subspace identification methods. The optimization problem on the first step involves a trade-off between the model complexity and the model accuracy. This trade-off is set by a user defined hyper-parameter.

Exercises

4.1 (Distance from a data point to a linear model). The 2-norm distance from a point $d \in \mathbb{R}^{\mathtt{q}}$ to a linear static model $\mathscr{B} \subset \mathbb{R}^{\mathtt{q}}$ is defined as

$$\operatorname{dist}(d, \mathscr{B}) := \min_{\widehat{d} \in \mathscr{B}} \|d - \widehat{d}\|_2. \tag{dist}$$

1. Let $\operatorname{image}(P)$ be a minimal representation of $\mathscr{B}$. Explain how to find $\operatorname{dist}(d, \mathscr{B})$. Find $\operatorname{dist}\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \operatorname{image}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)\right)$.
2. Let $\ker(R)$ be a minimal representation of $\mathscr{B}$. Explain how to find $\operatorname{dist}(d, \mathscr{B})$.
3. Prove that in the linear static case, a solution $\widehat{d}^*$ of (dist) is always unique.
4. Prove that in the linear static case, the approximation error $\Delta d^* := d - \widehat{d}^*$ is orthogonal to $\mathscr{B}$. Is the converse true, *i.e.*, is it true that if for some $\widehat{d}$, $d - \widehat{d}$ is orthogonal to $\mathscr{B}$, then $\widehat{d} = \widehat{d}^*$?

4.2 (Distance from a data point to an affine model). Consider again the distance $\operatorname{dist}(d, \mathscr{B})$ defined in (dist). In this problem, $\mathscr{B}$ is an affine static model, *i.e.*, $\mathscr{B} = \mathscr{B}' + c$, where $\mathscr{B}'$ is a linear static model and c is a fixed vector.

1. Explain how to reduce the problem of computing the distance from a point to an affine static model to an equivalent problem of computing the distance from a point to a linear static model (Problem 4.1).
2. Find $\operatorname{dist}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \ker\left(\begin{bmatrix} 1 & 1 \end{bmatrix}\right) + \begin{bmatrix} 1 \\ 2 \end{bmatrix}\right)$.

4.3 (Two-sided weighted low rank approximation).
Prove Theorem 4.12 on page 106. Using the result, write a MATLAB function that solves the two-sided weighted low-rank approximation problem (WLRA2).

4.4 (A simple method for approximate system identification). Modify the algorithm developed in Problem 3.3, so that it can be used as an approximate linear time-invariant identification method. Assume that the lag ℓ of the system is given.

4.5 (Misfit computation using state space representation).
Given a finite sequence $w_{\mathrm{d}} = \big(w_{\mathrm{d}}(1), \ldots, w_{\mathrm{d}}(T)\big)$ and an input/state/output representation $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$ of a linear time-invariant system $\mathscr{B}$, find the misfit $\mathrm{dist}\big(w_{\mathrm{d}}, \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)\big)\big)$ between w_{d} and $\mathscr{B}$, defined in (misfit $\mathscr{L}_{\mathtt{m},\ell}$).

References

Absil PA, Mahony R, Sepulchre R (2008) Optimization Algorithms on Matrix Manifolds. Princeton University Press, Princeton, NJ

Beck A, Ben-Tal A (2006) A global solution for the structured total least squares problem with block circulant matrices. SIAM J Matrix Anal Appl 27(1):238–255

Benner P, Mehrmann V, Sima V, Van Huffel S, Varga A (1999) SLICOT—a subroutine library in systems and control theory. In: Applied and Computational Control, Signal and Circuits, vol 1, Birkhauser, chap 10, pp 499–539

Boumal N, Mishra B, Absil PA, Sepulchre R (2014) Manopt, a Matlab toolbox for optimization on manifolds. Journal of Machine Learning Research 15:1455–1459, URL http://www.manopt.org

Chu M, Plemmons R (2003) Real-valued, low rank, circulant approximation. SIAM J Matrix Anal Appl 24(3):645–659

Dahl J, Vandenberghe L (2010) CVXOPT: Python software for convex optimization. URL abel.ee.ucla.edu/cvxopt

De Moor B (1993) Structured total least squares and L_2 approximation problems. Linear Algebra Appl 188–189:163–207

Fazel M (2002) Matrix rank minimization with applications. PhD thesis, Elec. Eng. Dept., Stanford University

Fazel M, Pong TK, Sun D, Tseng P (2013) Hankel matrix rank minimization with applications in system identification and realization. SIAM J Matrix Anal Appl 34(3):946–977

Galassi M, et al (2017) GNU scientific library reference manual. http://www.gnu.org/software/gsl/

Grant M, Boyd S (2008) Graph implementations for nonsmooth convex programs. In: Blondel V, Boyd S, Kimura H (eds) Recent Advances in Learning and Control, Springer, stanford.edu/~boyd/graph_dcp.html, pp 95–110

Grant M, Boyd S (2017) CVX: Matlab software for disciplined convex programming. stanford.edu/~boyd/cvx

Heij C (1989) Deterministic identification of dynamical systems, Lecture notes in control and information sciences, vol 127. Springer

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

Kailath T, Sayed A (1999) Fast reliable algorithms for matrices with structure. SIAM, Philadelphia

Keshavan R, Montanari A, Oh S (2010) Matrix completion from noisy entries. J Mach Learn Res 11:2057–2078

Kiers H (2002) Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems. Comput Stat Data Anal 41:157–170

Krijnen W (2006) Convergence of the sequence of parameters generated by alternating least squares algorithms. Comput Stat Data Anal 51:481–489

Liu Z, Vandenberghe L (2009) Interior-point method for nuclear norm approximation with application to system identification. SIAM J Matrix Anal Appl 31(3):1235–1256

Liu Z, Hansson A, Vandenberghe L (2013) Nuclear norm system identification with missing inputs and outputs. Control Lett 62:605–612

Manton J, Mahony R, Hua Y (2003) The geometry of weighted low-rank approximations. IEEE Trans Signal Proc 51(2):500–514

Markovsky I (2012) How effective is the nuclear norm heuristic in solving data approximation problems? In: Proc. of the 16th IFAC Symposium on System Identification, Brussels, pp 316–321

Markovsky I (2013) A software package for system identification in the behavioral setting. Control Eng Practice 21(10):1422–1436

Markovsky I, Usevich K (2013) Structured low-rank approximation with missing data. SIAM J Matrix Anal Appl 34(2):814–830

Markovsky I, Usevich K (2014) Software for weighted structured low-rank approximation. J Comput Appl Math 256:278–292

Markovsky I, Rastello M, Premoli A, Kukush A, Van Huffel S (2005) The element-wise weighted total least squares problem. Comput Statist Data Anal 50(1):181–209

Marquardt D (1963) An algorithm for least-squares estimation of nonlinear parameters. SIAM J Appl Math 11:431–441

Schuermans M, Markovsky I, Wentzell P, Van Huffel S (2005) On the equivalence between total least squares and maximum likelihood PCA. Analytica Chimica Acta 544(1–2):254–267

Usevich K, Markovsky I (2014) Variable projection for affinely structured low-rank approximation in weighted 2-norms. J Comput Appl Math 272:430–448

Wallin R, Hansson A (2014) Maximum likelihood estimation of linear SISO models subject to missing output data and missing input data. Int J Control 87(11):2354–2364

Wentzell P, Andrews D, Hamilton D, Faber K, Kowalski B (1997) Maximum likelihood principal component analysis. J Chemometrics 11:339–366

Part II
Applications and generalizations

Chapter 5
Applications

The difficult and interesting problems are defined by the applications.

S. Wold

For applicable *control engineering research, three things need to be present: a real and pressing set of problems , intuitively graspable theoretical approaches to design, which can be underpinned by sound mathematics, and good interactive software which can be used to turn designs into practical applications.*

MacFarlane (2013)

This chapter reviews applications of structured low-rank approximation for

1. model reduction,
2. approximate realization,
3. output only identification (sum-of-damped-exponentials modeling),
4. harmonic retrieval,
5. errors-in-variables system identification,
6. output error system identification,
7. finite impulse response system identification (deconvolution),
8. approximate common factor, controllability radius computation, and
9. pole placement by a low-order controller.

The problems occurring in these applications are special cases of the SLRA problem, obtained by choosing specific structure $\mathscr{S}$ and rank constraint r. Moreover, the structure is of the type (S), so that the algorithm and the software, developed in Section 4.2.3, can be used to solve the problems.

137 ⟨*solve Problem SLRA* 137⟩≡ (138a 142 145a 146c 148b)
```
[R, ph, info] = slra(tts, par, r, [], s0);
```
Uses slra 116c.

In the applications reviewed, the approximation norm $\|\cdot\|$ is the 2-norm $\|\cdot\|_2$.

5.1 Model reduction

The model reduction problem considered in this section takes as given data an impulse response of a high order linear time-invariant system and aims at a reduced order linear time-invariant system that approximates as well as possible the data. This *data-driven* model reduction problem can be viewed also as an approximate version of the realization problem or identification of an autonomous linear time-invariant system. In the special case of a model without repeated poles, the problem considered is the *sum-of-damped-exponentials modeling* problem in signal processing. When the exponents are undamped, *i.e.*, the model is marginally stable, the problem is called *harmonic retrieval*.

5.1.1 Approximate realization

Define the 2-norm $\|\Delta H\|_2$ of a matrix-valued signal $\Delta H \in (\mathbb{R}^{\mathtt{p}\times\mathtt{m}})^{T+1}$ as

$$\|\Delta H\|_2 := \sqrt{\textstyle\sum_{t=0}^{T} \|\Delta H(t)\|_{\mathrm{F}}^2}.$$

Acting on a finite sequence $\big(H(0),\ldots,H(T)\big)$, the shift operator σ removes $H(0)$.

Problem 5.1 (Approximate realization). Given a matrix valued finite time series $H_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{p}\times\mathtt{m}})^T$ and a complexity specification ℓ, find an optimal approximate model for H_{d} of a bounded complexity $(\mathtt{m},\ell)$, such that

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{H} \text{ and } \widehat{\mathscr{B}} && \|H_{\mathrm{d}} - \widehat{H}\|_2 \\ &\text{subject to} && \widehat{H} \text{ is the impulse response of } \widehat{\mathscr{B}} \in \mathscr{L}_{\mathtt{m},\ell}^{\mathtt{m}+\mathtt{p}}. \end{aligned}$$

138a ⟨*2-norm optimal approximate realization* 138a⟩≡
```
function [sysh, hh, info] = h2ss_opt(h, ell)
⟨reshape H and define m, p, T 75⟩
⟨approximate realization structure 138b⟩
⟨solve Problem SLRA 137⟩
⟨p̂ ↦ Ĥ 139a⟩, ⟨Ĥ ↦ B̂ 139b⟩
```
Defines:
h2ss_opt, used in chunks 140 and 141b.

Problem 5.1 is equivalent to Problem SLRA, with

- Hankel structured data matrix $\mathscr{S}(p) = \mathscr{H}_{\ell+1}(\sigma H_{\mathrm{d}})$ and
- rank reduction by the number of outputs p.

138b ⟨*approximate realization structure* 138b⟩≡ (138a)
```
par = vec(h(:, :, 2:end)); s0 = []; r = p * ell;
tts = blkhank(reshape(1:length(par), p, m, T - 1), ell + 1);
```
Uses blkhank 26a.

The statement follows from the basic fact of realization theory (see Theorem 2.5)

$$\widehat{H} \text{ is the impulse response of } \widehat{\mathscr{B}} \in \mathscr{L}_{\mathtt{m},\ell} \iff \operatorname{rank}\big(\mathscr{H}_{\ell+1}(\sigma\widehat{H})\big) \leq \mathtt{p}\ell.$$

The optimal approximate model $\widehat{\mathscr{B}}^*$ does not depend on the shape of the Hankel matrix as long as the number of rows and the number of columns are sufficiently large: at least $\mathtt{p}(\ell+1)$ rows and at least $\mathtt{m}(\ell+1)$ columns. The variable projection method, however, is not applicable for solving low-rank approximation problem with $\mathscr{H}_L(\sigma H_{\mathrm{d}})$, with $L > \ell+1$ due to violation of assumption A (see page 115).

The mapping $\widehat{p} \mapsto \widehat{H}$ from the solution $\widehat{p}$ of the structured low-rank approximation problem to the optimal approximation $\widehat{H}$ of the noisy impulse response H is reshaping the vector $\widehat{p}$ as a $\mathtt{m} \times \mathtt{p} \times T$ tensor `hh`, representing the sequence $\widehat{H}(1), \ldots, \widehat{H}(T-1)$ and setting $\widehat{H}(0) = H(0)$ (since $\widehat{D} = H(0)$).

139a ⟨$\widehat{p} \mapsto \widehat{H}$ 139a⟩≡ (138a)

```
hh = zeros(p, m, T); hh(:, :, 1) = h(:, :, 1);
hh(:, :, 2:end) = reshape(ph(:), p, m, T - 1);
```

The mapping $\widehat{H} \mapsto \widehat{\mathscr{B}}$ to the optimal model is the realization problem, *i.e.*, the 2-norm (locally) optimal realization $\widehat{\mathscr{B}}^*$ is obtained by exact realization of the approximation $\widehat{H}$, computed by the structured low-rank approximation method.

139b ⟨$\widehat{H} \mapsto \widehat{\mathscr{B}}$ 139b⟩≡ (138a)

```
sysh = h2ss(hh, l * p);
```

Uses h2ss 74c.

Due to the rank constraint of the Hankel matrix $\mathscr{H}_{\ell+1}(\sigma\widehat{H})$ in the structured low-rank approximation problem, by construction, the approximation $\widehat{H}$ of H has an exact realization in the model class $\mathscr{L}^{\mathtt{m}+\mathtt{p}}_{\mathtt{m},\ell}$.

Note 5.2 (Using the $\widehat{R}$ parameter of the structured low-rank approximation solver to obtain the model $\widehat{\mathscr{B}}$). In the numerical solution of the structured low-rank approximation problem, the kernel representation (rank_R) of the rank constraint is used (see page 112). The parameter R, computed by the solver, gives a kernel representation of the optimal approximate model $\widehat{\mathscr{B}}^*$. The kernel representation can subsequently be converted into a state space representation. This gives an alternative (more efficient) way of implementing the function `h2ss_opt` to the one using system realization ($\widehat{H} \mapsto \widehat{\mathscr{B}}$). The same note applies to the other problems, reviewed in the chapter.

Example 5.3. The following script verifies that the local optimization based method `h2ss_opt` improves the suboptimal approximation computed by Kung's method `h2ss`. The data is a noisy impulse response of a random stable linear time-invariant system. The number of inputs `m` and outputs `p`, the order `n` of the system, the number of data points `T`, and the noise standard deviation `s` are simulation parameters.

139c ⟨*Test* h2ss_opt 139c⟩≡ 140a▷

```
⟨initialize the random number generator 25⟩
n = p * l; sys0 = drss(n, p, m);
h0 = reshape(shiftdim(impulse(sys0, T), 1), p, m, T);
h = h0 + s * randn(size(h0));
```


Defines:
test_h2ss_opt, used in chunk 140b.

The solutions, obtained by the unstructured and Hankel structured low-rank approximation methods, are computed and the relative approximation errors are printed.

140a ⟨*Test* h2ss_opt 139c⟩+≡ ◁139c

```
[sysh, hh] = h2ss(h, n); norm(h(:) - hh(:)) / norm(h(:))
[sysh_, hh_] = h2ss_opt(h, l); norm(h(:) - hh_(:)) / norm(h(:))
```

Uses h2ss 74c and h2ss_opt 138a.

The optimization based method improves the suboptimal results of the singular value decomposition based method at the price of extra computation, a process referred to as *iterative refinement* of the solution.

140b ⟨*Compare* h2ss *and* h2ss_opt 140b⟩≡

```
m = 2; p = 3; l = 1; T = 25; s = 0.2; test_h2ss_opt
```

Uses test_h2ss_opt 139c.

⇝ 0.6231 for `h2ss` and 0.5796 for `h2ss_opt`

5.1.2 Model reduction

The finite time H_2 norm $\|\Delta\mathscr{B}\|_T$ of a linear time-invariant system $\Delta\mathscr{B}$ is defined as the 2-norm of the sequence of its first T Markov parameters, *i.e.*, if ΔH is the impulse response of $\Delta\mathscr{B}$, $\|\Delta\mathscr{B}\|_T := \|\Delta H\|_2$.

Problem 5.4 (Finite time H_2 model reduction). Given a linear time-invariant system $\mathscr{B}_{\mathrm{d}} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}$ and a complexity specification $\ell_{\mathrm{red}} < \ell$, find an optimal approximation of $\mathscr{B}_{\mathrm{d}}$ with bounded complexity $(\mathtt{m}, \ell_{\mathrm{red}})$, such that

$$\text{minimize} \quad \text{over } \widehat{\mathscr{B}} \quad \|\mathscr{B}_{\mathrm{d}} - \widehat{\mathscr{B}}\|_T \quad \text{subject to} \quad \widehat{\mathscr{B}} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell_{\mathrm{red}}}.$$

Problem 5.4 is equivalent to Problem SLRA with

- Hankel structured data matrix $\mathscr{S}(p) = \mathscr{H}_{\ell+1}(\sigma H_{\mathrm{d}})$, where H_{d} is the impulse response of $\mathscr{B}_{\mathrm{d}}$ and
- rank reduction by the number of outputs $\mathtt{p} := \mathtt{q} - \mathtt{m}$,

Therefore, finite time H_2 model reduction is equivalent to the approximate realization problem with H_{d} being the impulse response of $\mathscr{B}_{\mathrm{d}}$. In practice, $\mathscr{B}_{\mathrm{d}}$ need not be linear time-invariant system since in the model reduction problem only the knowledge of its impulse response H_{d} is used.

140c ⟨*Finite time H_2 model reduction* 140c⟩≡

```
function [sysh, hh, info] = mod_red(sys, T, lred)
[sysh, hh, info] = h2ss_opt(shiftdim(impulse(sys, T), 1), lred);
```

Uses h2ss_opt 138a.

5.1.3 Output only identification / autonomous system identification

Realization of an impulse response is closely related to exact identification of an autonomous system. To show this, let $\mathscr{B}_{\text{i/s/o}}(A,b,C,d)$ be a realization of y. Then the response of the autonomous system $\mathscr{B}_{\text{ss}}(A,C)$ to initial condition $x(0)=b$ is y.

141a ⟨*impulse response realization ↦ autonomous system realization* 141a⟩≡ (141b 142)

```
xinih = sysh.b; sysh = ss(sysh.a, [], sysh.c, [], -1);
```

This link gives yet another problem that is equivalent to the approximate realization.

Problem 5.5 (Output only identification / autonomous system identification). Given a signal $y_{\text{d}} \in (\mathbb{R}^{\text{p}})^T$ and a complexity specification ℓ, find an optimal approximate model for y_{d} of bounded complexity $(0,\ell)$, such that

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{y} \quad \|y_{\text{d}} - \widehat{y}\|_2 \\ &\text{subject to} && \widehat{y} \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}^{\text{p}}_{0,\ell}. \end{aligned}$$

Problem 5.5 is equivalent to Problem SLRA with

- Hankel structured data matrix $\mathscr{S}(p) = \mathscr{H}_{\ell+1}(y_{\text{d}})$ and
- rank reduction by the number of outputs p.

Note 5.6 (Sum-of-damped-exponetials modeling). Excluding the cases of multiple poles, the model class of autonomous linear time-invariant systems $\mathscr{L}^{\text{p}}_{0,\ell}$ is equivalent to the *sum-of-damped-exponentials model* class, *i.e.*, signals y

$$y(t) = \textstyle\sum_{k=1}^{\ell} \alpha_k e^{\beta_k t} e^{\mathbf{i}(\omega_k t + \phi_k)}, \quad (\mathbf{i} = \sqrt{-1}).$$

The parameters $\{\, \alpha_k, \beta_k, \omega_k, \phi_k \,\}_{j=1}^{\ell}$ of the sum-of-damped-exponentials model have the following meaning: α_k are amplitudes, β_k damping factors, ω_k frequencies, and ϕ_k initial phases of the k-th exponential signal.

141b ⟨*Output only identification* 141b⟩≡

```
function [sysh, yh, xinih, info] = ident_aut(y, l)
[sysh, yh, info] = h2ss_opt(y, l);
⟨impulse response realization ↦ autonomous system realization 141a⟩
```

Uses h2ss_opt 138a.

5.1.4 Harmonic retrieval

The aim of the harmonic retrieval problem is to approximate the data by a sum of sines. From a system identification point of view, harmonic retrieval aims to model the data by a marginally stable linear time-invariant autonomous system.

Problem 5.7 (Harmonic retrieval). Given a signal $y_{\text{d}} \in (\mathbb{R}^{\text{p}})^T$ and a complexity specification ℓ, find an optimal approximate model for y_{d} that is in the model class $\mathscr{L}^{\text{p}}_{0,\ell}$ and is marginally stable, *i.e.*,

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{y} \quad \|y_{\text{d}} - \widehat{y}\|_2 \\ &\text{subject to} && \widehat{y} \in \widehat{\mathscr{B}}|_T,\ \widehat{\mathscr{B}} \in \mathscr{L}^{\text{p}}_{0,\ell}, \text{ and } \widehat{\mathscr{B}} \text{ is marginally stable.} \end{aligned}$$

Due to the stability constraint, Problem 5.7 is not a special case of problem SLRA. In the univariate case $\text{p} = 1$, however, a necessary condition for an autonomous model $\mathscr{B}$ to be marginally stable is that the parameter R of a kernel representation $\ker\big(R(\sigma)\big)$ of $\mathscr{B}$ is either palindromic,

$$R(z) := \sum_{i=0}^{\ell} z^i R_i \text{ is palindromic} \quad :\iff \quad R_{\ell-i} = R_i, \text{ for } i = 0,1,\ldots,\ell$$

or antipalindromic ($R_{\ell-i} = -R_i$). The antipalindromic case is nongeneric in the space of the marginally stable systems, so as relaxation of the stability constraint, we can use the constraint that the kernel representation is palindromic (Markovsky and Rao, 2008).

Problem 5.8 (Harmonic retrieval, relaxed version, scalar case). Given a signal $y_{\text{d}} \in (\mathbb{R})^T$ and a complexity specification ℓ, find an optimal approximate model for y_{d} that is in the model class $\mathscr{L}^{1}_{0,\ell}$ and has a palindromic kernel representation,

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{y} \quad \|y_{\text{d}} - \widehat{y}\|_2 \\ &\text{subject to} && \widehat{y} \in \widehat{\mathscr{B}}|_T,\ \widehat{\mathscr{B}} \in \mathscr{L}^{1}_{0,\ell} \text{ and } \ker(\widehat{R}) = \widehat{\mathscr{B}}, \text{with } R \text{ palindromic.} \end{aligned}$$

142 ⟨*Harmonic retrieval* 142⟩≡

```
function [sysh, yh, xinih, info] = harmonic_retrieval(y, l)
⟨harmonic retrieval structure 143a⟩
⟨solve Problem SLRA 137⟩, yh = ph; sysh = h2ss(yh, n);
⟨impulse response realization ↦ autonomous system realization 141a⟩
```

Defines:
harmonic_retrieval, used in chunk 143b.
Uses h2ss 74c.

The constraint "R palindromic" can be expressed as a structural constraint on the data matrix, which reduces the relaxed harmonic retrieval problem to the structured low-rank approximation problem. Problem 5.8 is equivalent to Problem SLRA with

- structured data matrix composed of a Hankel matrix next to a Toeplitz matrix:

$$\mathscr{S}(p) = \begin{bmatrix} \mathscr{H}_{\ell+1}(y) & \updownarrow \mathscr{H}_{\ell+1}(y) \end{bmatrix},$$

where

$$\updownarrow\mathscr{H}_{\ell+1}(y) := \begin{bmatrix} y_{\ell+1} & y_{\ell+2} & \cdots & y_T \\ \vdots & \vdots & & \vdots \\ y_2 & y_3 & \cdots & y_{T-\ell+1} \\ y_1 & y_2 & \cdots & y_{T-\ell} \end{bmatrix},$$

- rank reduction by one.

143a ⟨*harmonic retrieval structure* 143a⟩≡ (142)

```
par = y(:); np = length(par); n = l * 1; r = n; s0 = [];
tts = [blkhank(1:np, n + 1) flipud(blkhank(1:np, n + 1))];
```

Uses blkhank 26a.

The statement follows from the equivalence

$$\widehat{y} \in \widehat{\mathscr{B}}|_T,\ \widehat{\mathscr{B}} \in \mathscr{L}^1_{0,\ell} \text{ and } \ker(\widehat{R}) = \widehat{\mathscr{B}} \text{ is palindromic}$$
$$\iff \quad \operatorname{rank}\left(\begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{y}) & \updownarrow\mathscr{H}_{\ell+1}(\widehat{y}) \end{bmatrix}\right) \leq \ell.$$

In order to show it, let $\ker(R)$, with $R(z) = \sum_{i=0}^{\ell} z^i R_i$ full row rank, be a kernel representation of $\mathscr{B} \in \mathscr{L}^1_{0,\ell}$. Then $\widehat{y} \in \widehat{\mathscr{B}}|_T$ is equivalent to

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \mathscr{H}_{\ell+1}(\widehat{y}) = 0.$$

If, in addition, R is palindromic, then

$$\begin{bmatrix} R_\ell & \cdots & R_1 & R_0 \end{bmatrix} \mathscr{H}_{\ell+1}(\widehat{y}) = 0 \quad \iff \quad \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \updownarrow\mathscr{H}_{\ell+1}(\widehat{y}) = 0.$$

We have that

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{y}) & \updownarrow\mathscr{H}_{\ell+1}(\widehat{y}) \end{bmatrix} = 0. \tag{$*$}$$

which is equivalent to

$$\operatorname{rank}\left(\begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{y}) & \updownarrow\mathscr{H}_{\ell+1}(\widehat{y}) \end{bmatrix}\right) \leq \ell.$$

Conversely, ($*$) implies $\widehat{y} \in \widehat{\mathscr{B}}|_T$ and R palindromic.

Example 5.9. The data is generated as a sum of random sinusoids with additive noise. The number hn of sinusoids, the number of samples T, and the noise standard deviation s are simulation parameters.

143b ⟨*Test* harmonic_retrieval 143b⟩≡

```
⟨initialize the random number generator 25⟩
t  = 1:T; f = 1 * pi * rand(hn, 1); phi = 2 * pi * rand(hn, 1);
y0 = sum(sin(f * t + phi(:, ones(1, T))));
yt = randn(size(y0)); y = y0 + s * norm(y0) * yt / norm(yt);
[sysh, yh, xinih, info] = harmonic_retrieval(y, hn * 2);
```

Defines:
test_harmonic_retrieval, used in chunk 144.
Uses harmonic_retrieval 142.

Figure 5.1 shows the true signal and the estimate obtained with harmonic_retrieval in the following simulation example:

144 ⟨*Example of harmonic retrieval* 144⟩≡

```
clear all, T = 50; hn = 2; s = 0.015; test_harmonic_retrieval
```

Uses test_harmonic_retrieval 143b.

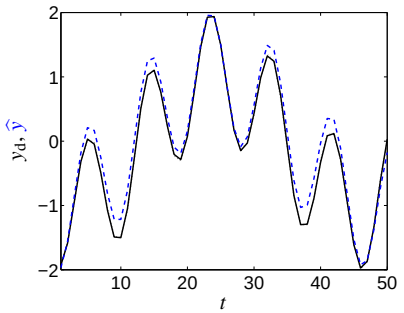


Fig. 5.1: In a simulation example, imposing only the palindromic constraint results in a marginally stable model that gives a good approximation of the data (solid black line — true system's trajectory y_0 and dashed blue line — best approximation $\widehat{y}$).

5.2 System identification

In Section 5.1, the given data is an impulse or a free response and the link between the applications and the structured low-rank approximation problem is given by realization theory. In this section, the data is a general trajectory of the system and the link to Problem SLRA is given by the following lemma.

Lemma 5.10. *The signal w is a trajectory of a linear time-invariant system of complexity bounded by $(\mathtt{m}, \ell)$,* i.e.,

$$w|_{T-\ell} \in \mathscr{B}|_{T-\ell} \quad \textit{and} \quad \mathscr{B} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}$$

if and only if

$$\operatorname{rank}\left(\mathscr{H}_{\ell+1}(w)\right) \leq \mathtt{m}(\ell+1) + (\mathtt{q}-\mathtt{m})\ell.$$

The problems considered are errors-in-variables system identification, output error identification, and identification of a finite impulse response system.

5.2.1 Errors-in-variables identification

In errors-in-variables data modeling problems, the observed variables are a priori known (or assumed) to be noisy. This prior knowledge is used to correct the data, so that the corrected data is consistent with a model in the model class and the correction is as small as possible. The resulting problem is misfit minimization.

Problem 5.11 (Errors-in-variables identification). Given T samples, $\mathtt{q}$ variables, vector signal $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{q}})^T$ and a model complexity $(\mathtt{m}, \ell)$,

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{w} \quad \|w_{\mathrm{d}} - \widehat{w}\|_2 \\ &\text{subject to} && \widehat{w} \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}. \end{aligned}$$

145a ⟨*Errors-in-variables identification* 145a⟩≡
```
function [sysh, wh, info] = ident_eiv(w, m, l)
⟨reshape w and define q, T 27a⟩
⟨errors-in-variables identification structure 145b⟩
⟨solve Problem SLRA 137⟩, wh = reshape(ph(:), q, T);
⟨exact identification: ŵ ↦ B̂ 145c⟩
```
Defines:
`ident_eiv`, used in chunk 146a.

Problem 5.11 is equivalent to Problem SLRA with

- Hankel structured data matrix $\mathscr{S}(p) = \mathscr{H}_{\ell+1}(w_{\mathrm{d}})$ and
- rank reduction with the number of outputs $\mathtt{p}$.

145b ⟨*errors-in-variables identification structure* 145b⟩≡ (145a)
```
par = w(:); np = length(par); n = l * 1;
p = q - m; r = m * (l + 1) + n;
tts = blkhank(reshape(1:np, q, T), l + 1); s0 = [];
```
Uses `blkhank` 26a.

The identified system can be recovered from the optimal approximating trajectory $\widehat{w}$ by exact identification.

145c ⟨*exact identification:* $\widehat{w} \mapsto \widehat{\mathscr{B}}$ 145c⟩≡ (145a 146c)
```
sysh = w2h2ss(wh, m, n);
```
Uses `w2h2ss` 78c.

Note 5.12 (Using the $\widehat{R}$ parameter of the structured low-rank approximation solver to obtain the model $\widehat{\mathscr{B}}$). The parameter $\widehat{R}$, computed by the solver, gives a kernel representation of the optimal approximate model. The kernel representation can subsequently be converted to a state space representation. This gives an alternative way of implementing the function `ident_eiv` to the one using exact identification.

Example 5.13. In this example, the approximate model computed by the function `ident_eiv` is compared with the model obtained by the function `w2h2ss`. Although `w2h2ss` is an exact identification method, it can be used as a heuristic for

approximate identification. The data is generated in the errors-in-variables setup. The true system is a random single-input single-output system.

146a ⟨*Test* `ident_eiv` 146a⟩≡
```
⟨initialize the random number generator 25⟩
m = 1; p = 1; n = p * l; sys0 = drss(n, p, m);
xini0 = rand(n, 1); u0 = rand(T, m);
y0 = lsim(sys0, u0, 1:T, xini0);
w = [u0'; y0'] + s * randn(m + p, T);
sys = w2h2ss(w, m, n); ⟨(TF) ↦ P(z) 240c⟩ misfit_siso(w, P)
[sysh, wh, info] = ident_eiv(w, m, l); info.M
```
Defines:
`test_ident_eiv`, used in chunk 146b.
Uses `ident_eiv` 145a, `misfit_siso` 117b, and `w2h2ss` 78c.

In a particular example

146b ⟨*Compare* `w2h2ss` *and* `ident_eiv` 146b⟩≡
```
l = 4; T = 30; s = 0.1; test_ident_eiv
```
Uses `test_ident_eiv` 146a.

the obtained results are misfit 1.2113 for `w2h2ss` and 0.2701 for `ident_eiv`.

5.2.2 Output error identification

In the errors-in-variables setting, using an input-output partitioning of the variables, both the input and the output are noisy. In some applications, however, the input is not measured; it is designed by the user. Then, it is natural to assume that the input is noise free. This leads to the output error identification problem.

Problem 5.14 (Output error identification). Given a signal $y_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{p}})^T$ with an input/output partitioning $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$, $\dim(u) = \mathtt{m}$, and a complexity specification ℓ, find an optimal approximate model for w_{d} of a bounded complexity $(\mathtt{m}, \ell)$, such that

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{y} \quad \|y_{\mathrm{d}} - \widehat{y}\|_2 \\ &\text{subject to} && (u_{\mathrm{d}}, \widehat{y}) \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}. \end{aligned}$$

146c ⟨*Output error identification* 146c⟩≡
```
function [sysh, wh, info] = ident_oe(w, m, l)
⟨reshape w and define q, T 27a⟩
⟨output error identification structure 147a⟩
⟨solve Problem SLRA 137⟩, wh = [w(1:m, :); reshape(ph(:), p, T)];
⟨exact identification: ŵ ↦ B̂ 145c⟩
```
Defines:
`ident_oe`, used in chunk 147b.

Output error identification is a limiting case of errors-in-variables identification when the noise variance tends to zero. Alternatively, output error identification is a special case of ARMAX system identification when the noise is not modeled (the stochastic part $H(z)$, see Section 2.3, is the identity).

As shown next, Problem 5.14 is equivalent to Problem SLRA with

- data matrix $\mathscr{S}(p) = \begin{bmatrix} \mathscr{H}_{\ell+1}(u_\mathrm{d}) \\ \mathscr{H}_{\ell+1}(y_\mathrm{d}) \end{bmatrix}$ has a fixed block and a Hankel block,
- rank reduction by the number of outputs p.

147a ⟨*output error identification structure* 147a⟩≡ (146c)
```
par = vec(w((m + 1):end, :)); np = length(par); p = q - m;
j = T - l; n = l * p; r = m * (l + 1) + n;
s0  = [blkhank(w(1:m, :), l + 1); zeros((l + 1) * p, j)];
tts = [zeros((l + 1) * m, j);
blkhank(reshape(1:np, p, T), l + 1)];
```
Uses blkhank 26a.

The statement is based on a corollary of Lemma 5.10:

$$\begin{bmatrix} u_\mathrm{d} \\ \widehat{y} \end{bmatrix} \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathscr{L}_{\mathtt{m},\ell} \quad \Longleftrightarrow \quad \operatorname{rank}\left(\begin{bmatrix} \mathscr{H}_{\ell+1}(u_\mathrm{d}) \\ \mathscr{H}_{\ell+1}(\widehat{y}) \end{bmatrix} \right) \leq \mathtt{q}\ell + \mathtt{m}.$$

Example 5.15. This example is analogous to the example of the errors-in-variables identification method. The approximate model obtained with the function w2h2ss is compared with the approximate model obtained with ident_oe. In this case, the data is generated in the output error setting, *i.e.*, the input is exact and the output is noisy. In this simulation setup we expect that the optimization based method ident_oe improves the result obtained with the subspace based method w2h2ss.

147b ⟨*Test* ident_oe 147b⟩≡
```
⟨initialize the random number generator 25⟩
m = 1; p = 1; n = l * p; sys0 = drss(n, p, m);
xini0 = rand(n, 1);
u0 = rand(T, m); y0 = lsim(sys0, u0, 1:T, xini0);
w = [u0'; y0'] + s * [zeros(m, T); randn(p, T)];
sys = w2h2ss(w, m, n); ⟨(TF) ↦ P(z) 240c⟩ misfit_siso(w, P)
[sysh, wh, info] = ident_oe(w, m, l); info.M
```
Defines:
test_ident_oe, used in chunk 147c.
Uses ident_oe 146c, misfit_siso 117b, and w2h2ss 78c.

In the following simulation example

147c ⟨*Example of output error identification* 147c⟩≡
```
l = 4; T = 30; s = 0.1; test_ident_oe
```
Uses test_ident_oe 147b.

w2h2ss achieves misfit 1.0175 and ident_oe achieves misfit 0.2331.

5.2.3 *Finite impulse response system identification*

Let $\mathrm{FIR}_{\mathtt{m},\ell}$ be the model class of finite impulse response linear time-invariant systems with m inputs and lag at most ℓ, *i.e.*,

$$\mathrm{FIR}_{\mathtt{m},\ell} := \{ \mathscr{B} \in \mathscr{L}_{\mathtt{m},\ell} \mid \mathscr{B} \text{ has a finite impulse response} \}.$$

Identification of a finite impulse response model in the output error setting leads to the ordinary linear least squares problem

$$\begin{bmatrix} \widehat{H}(0) & \widehat{H}(1) & \cdots & \widehat{H}(\ell) \end{bmatrix} \mathscr{H}_{\ell+1}(u_\mathrm{d}) = \begin{bmatrix} y_\mathrm{d}(1) & \cdots & y_\mathrm{d}(T-\ell) \end{bmatrix}.$$

148a ⟨*Output error finite impulse response identification* 148a⟩≡
```
function [hh, wh] = ident_fit_oe(w, m, l)
⟨reshape w and define q, T 27a⟩
⟨Finite impulse response identification structure 149⟩
D = par(tts);
hh_ = D(((m * (l + 1)) + 1):end, :) / D(1:(m * (l + 1)), :);
hh = reshape(hh_, p, m, l + 1); hh = hh(:, :, end:-1:1);
uh = w(1:m, :); yh = [hh_ * D(1:(m * (l + 1)), :) zeros(p, l)];
wh = [uh; yh];
```
Defines:
ident_fir_oe, never used.

Next, we define the finite impulse response identification problem in the errors-in-variables setting.

Problem 5.16 (Errors-in-variables finite impulse response identification). Given a signal $y_\mathrm{d} \in (\mathbb{R}^{\mathtt{p}})^T$ with an input/output partition $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$, with $\dim(u) = \mathtt{m}$, and a complexity specification ℓ, find an optimal approximate finite impulse response model for w_d of bounded complexity $(\mathtt{m}, \ell)$, such that

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{w} \quad \|w_\mathrm{d} - \widehat{w}\|_2 \\ &\text{subject to} && \widehat{w} \in \widehat{\mathscr{B}}|_T \text{ and } \widehat{\mathscr{B}} \in \mathrm{FIR}_{\mathtt{m},\ell}. \end{aligned}$$

148b ⟨*Errors-in-variables finite impulse response identification* 148b⟩≡
```
function [hh, wh, info] = ident_fir_eiv(w, m, l)
⟨reshape w and define q, T 27a⟩
⟨Finite impulse response identification structure 149⟩
⟨solve Problem SLRA 137⟩, hh = rio2x(R)';
hh = reshape(hh, p, m, l + 1);
hh = hh(:, :, end:-1:1);
uh = reshape(ph(1:(T * m)), m, T);
yh = reshape(ph(((T * m) + 1):end), p, T - l);
wh = [uh; [yh zeros(p, l)]];
```
Defines:
ident_fir_eiv, never used.
Uses rio2x 43b.

Problem 5.16 is equivalent to Problem SLRA with

- data matrix
$$\mathscr{S}(p) = \begin{bmatrix} \mathscr{H}_{\ell+1}(u_\mathrm{d}) \\ \begin{bmatrix} y_\mathrm{d}(1) & \cdots & y_\mathrm{d}(T-\ell) \end{bmatrix} \end{bmatrix},$$
composed of a fixed block and a Hankel structured block and
- rank reduction by the number of outputs p.

149 ⟨*Finite impulse response identification structure* 149⟩≡ (148)

```
p = q - m; r = (l + 1) * m;
par = vec([w(1:m, :), w((m + 1):end, 1:(T - l))]); s0 = [];
tts = [blkhank((1:(T * m)), l + 1);
       reshape(T * m + (1:((T - l) * p)), p, T - l)];
```

Uses blkhank 26a.

The statement follows from the equivalence

$$\widehat{w}|_{T-\ell} \in \mathscr{B}|_{T-\ell} \text{ and } \widehat{\mathscr{B}} \in \mathrm{FIR}_{\mathtt{m},\ell} \quad \Longleftrightarrow \quad \operatorname{rank}\left(\begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{u}) \\ \begin{bmatrix}\widehat{y}(1) & \cdots & \widehat{y}(T-\ell)\end{bmatrix}\end{bmatrix}\right) \leq \mathtt{m}(\ell+1).$$

In order to show it, let $H = \big(H(0), H(1), \ldots, H(\ell), 0, 0, \ldots\big)$. be the impulse response of $\widehat{\mathscr{B}} \in \mathrm{FIR}_{\mathtt{m},\ell}$. The signal $\widehat{w} = \begin{bmatrix}\widehat{u}\\ \widehat{y}\end{bmatrix}$ is a trajectory of $\mathscr{B}$ if and only if

$$\begin{bmatrix} H(\ell) & \cdots & H(1) & H(0)\end{bmatrix} \mathscr{H}_{\ell+1}(\widehat{u}) = \begin{bmatrix}\widehat{y}(1) & \cdots & \widehat{y}(T-\ell)\end{bmatrix}.$$

Equivalently, $\widehat{w} = \begin{bmatrix}\widehat{u}\\ \widehat{y}\end{bmatrix}$ is a trajectory of $\mathscr{B}$ if and only if

$$\begin{bmatrix} H(\ell) & \cdots & H(1) & H(0) & -I_{\mathtt{p}}\end{bmatrix} \begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{u}) \\ \begin{bmatrix}\widehat{y}(1) & \cdots & \widehat{y}(T-\ell)\end{bmatrix}\end{bmatrix} = 0,$$

which implies that,

$$\operatorname{rank}\left(\begin{bmatrix} \mathscr{H}_{\ell+1}(\widehat{u}) \\ \begin{bmatrix}\widehat{y}(1) & \cdots & \widehat{y}(T-\ell)\end{bmatrix}\end{bmatrix}\right) \leq \mathtt{m}(\ell+1).$$

For exact data, *i.e.*, assuming that

$$y_{\mathrm{d}}(t) = (H \star u_{\mathrm{d}})(t) := \sum_{\tau=0}^{\ell} H(\tau) u_{\mathrm{d}}(t-\tau)$$

the finite impulse response identification problem is equivalent to the deconvolution problem: Given the signals u_{d} and $y_{\mathrm{d}} := H \star u_{\mathrm{d}}$, find the signal H. For noisy data, the finite impulse response identification problem can be viewed as an *approximate deconvolution problem*. The approximation is in the sense of finding the nearest signals $\widehat{u}$ and $\widehat{y}$ to the given ones u_{d} and y_{d}, such that $\widehat{y} := \widehat{H} \star \widehat{u}$.

5.3 Approximate common factor of two polynomials

Section 5.3.1 defines the approximate version of the greatest common divisor problem, when the given polynomials are co-prime and an approximate common factor of a specified degree d is desired. Section 5.3.2 develops a solution method based

on low-rank approximation. An alternative method that optimizes directly over the common factor is developed in Section 5.3.3. Section 5.3.4 show an application of the approximate common factor computation problem for computing the distance of a given linear time-invariant system to the set of uncontrollable systems.

5.3.1 Problem formulation

Since $(\mathtt{n}+1)$-dimensional vectors correspond to degree-n polynomials

$$\operatorname{col}(p_0, p_1, \ldots, p_{\mathtt{n}}) \in \mathbb{R}^{\mathtt{n}+1} \quad \leftrightarrow \quad p(z) = p_0 + p_1 z + \cdots + p_{\mathtt{n}} z^{\mathtt{n}} \in \mathbb{R}[z],$$

with some abuse of notation, we refer to p as both the vector and the polynomial.

The polynomials p and $\widehat{p}$ of degree n are "close" to each other if the distance

$$\operatorname{dist}\left(p, \widehat{p}\right) := \|p - \widehat{p}\|_2 \tag{dist}$$

is "small", *i.e.*, if the norm of the error $\Delta p := p - \widehat{p}$ is small. (dist) may not be an appropriate distance measure in applications where the polynomial roots rather than coefficients are of primary interest. Polynomial roots might be sensitive (especially for high order polynomials) to perturbations in the coefficients, so that closeness of coefficients does not necessarily imply closeness of the roots. Using (dist), however, simplifies the solution of the approximate common factor problem defined next.

Problem 5.17 (Approximate common factor). Given polynomials $\mathscr{D} = \{p, q\}$, of degree n, and a natural number d, find polynomials $\widehat{\mathscr{D}} = \{\widehat{p}, \widehat{q}\}$ that have a common factor c of degree d and minimize the approximation error

$$\operatorname{dist}(\mathscr{D}, \widehat{\mathscr{D}}) := \sqrt{\operatorname{dist}^2(p, \widehat{p}) + \operatorname{dist}^2(q, \widehat{q})}.$$

The polynomial c is an optimal approximate common factor of p and q.

Note 5.18 (Connection to data modeling). As in the data modeling problems, we denote with $\mathscr{D}$ the given data (a pair of polynomials) and with $\widehat{\mathscr{D}}$ the approximation of the data. The equivalent of a "model" in Problem 5.17 is the common factor c. The equivalent of "model complexity", however, is not the degree d of the common factor but $\mathtt{n} - \mathtt{d}$. Indeed, the trivial case of co-prime polynomials ($\mathtt{d} = 0$) corresponds to the highest complexity "model".

Note 5.19 (Common factor certificate). The object of interest in solving Problem 5.17 is the approximate common factor c. The approximating polynomials $\widehat{p}$ and $\widehat{q}$ are auxiliary variables introduced for the purpose of defining c. They serve, however, as a certificate that the obtained polynomial c is a common factor of nearby polynomials to the given polynomials p and q. Indeed, with given $\widehat{p}$ and $\widehat{q}$ the cost function $\operatorname{dist}(\mathscr{D}, \widehat{\mathscr{D}})$ can be directly evaluated.

5.3.2 Low-rank approximation of the Sylvester matrix

As shown in Section 1.3.3, Problem 5.17 can be restated as an equivalent Sylvester structured low-rank approximation. Note, however, that the rank reduction is equal to the degree $\mathtt{d}$ of the common factor. Then, for $\mathtt{d} > 1$, the variable projection method developed in Section 4.2.3 can not be used to solve the Sylvester structured low-rank approximation problem because assumption (A) (see page 115) is not satisfied. This problem is circumvented in the following theorem by using a *reduced Sylvester* matrix, in which case the necessary rank reduction is one.

Proposition 5.20 *The polynomials p and q have a common factor of degree $\mathtt{d}$ if and only if the reduced Sylvester matrix*

$$\mathscr{R}_{\mathtt{d}}(p,q) = \begin{bmatrix} p_0 & & & q_0 & & & \\ p_1 & p_0 & & q_1 & q_0 & & \\ \vdots & p_1 & \ddots & \vdots & q_1 & \ddots & \\ p_{\mathtt{n}} & \vdots & \ddots & p_0 & q_{\mathtt{n}} & \vdots & \ddots & q_0 \\ & p_{\mathtt{n}} & & p_1 & & q_{\mathtt{n}} & & q_1 \\ & & \ddots & \vdots & & & \ddots & \vdots \\ & & & p_{\mathtt{n}} & & & & q_{\mathtt{n}} \end{bmatrix} \in \mathbb{R}^{(2\mathtt{n}-\mathtt{d}+1)\times(2\mathtt{n}-2\mathtt{d}+2)} \qquad (\mathscr{R}_{\mathtt{d}})$$

is rank deficient.

151 ⟨*Reduced Sylvester matrix constructor* 151⟩≡

```
function S = red_sylv(p, q, d)
n = length(p) - 1;
S = [blktoep(p, n - d + 1)' blktoep(q, n - d + 1)'];
```

Defines:
`red_sylv`, used in chunk 247a.
Uses `blktoep` 117a.

Proposition 5.20 shows that Problem 5.17 is equivalent to the structured low-rank approximation problem

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{p},\widehat{q} \in \mathbb{R}^{\mathtt{n}+1} && \left\| \begin{bmatrix} p \\ q \end{bmatrix} - \begin{bmatrix} \widehat{p} \\ \widehat{q} \end{bmatrix} \right\| \\ &\text{subject to} && \operatorname{rank}\left(\mathscr{R}_{\mathtt{d}}(\widehat{p},\widehat{q})\right) = 2\mathtt{n}-2\mathtt{d}+1 \end{aligned} \qquad (\text{ACF-}\mathscr{R}_{\mathtt{d}})$$

with rank reduction 1 (rather than $\mathtt{d}$ as in case of using the full Sylvester matrix $\mathscr{R}$).

Note 5.21. The approximate common factor c of p and q is not explicitly computed in (ACF-$\mathscr{R}_{\mathtt{d}}$), however, it can be found from $\widehat{p}$ and $\widehat{q}$ by finding the greatest common factor of $\widehat{p}$ and $\widehat{q}$. Note that this step does not involve an approximation.

Problem (ACF-$\mathscr{R}_{\mathtt{d}}$) is equivalent to Problem SLRA with

- reduced Sylvester data matrix $\mathscr{S}(p) = \mathscr{R}_{\mathtt{d}}(p,q)$ and

- rank reduction by one.

Using Theorem 5.20, in Exercise 5.2, you will develop a method for approximate common factor computation that is based on the `slra` function.

Ignoring the Sylvester structure constraint in (ACF-$\mathscr{R}_{\mathtt{d}}$) results in a suboptimal solution method, based on unstructured low-rank approximation

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{D} \text{ and } z && \left\| \mathscr{R}_{\mathtt{d}}(p,q) - \widehat{D} \right\|_{\mathrm{F}}^2 \\ &\text{subject to} && \widehat{D}z = 0 \quad \text{and} \quad z^\top z = 1. \end{aligned} \qquad (\text{ACF-LRA})$$

From the proof of Proposition 5.20 (see page 264), we have that

$$z := \begin{bmatrix} v \\ -u \end{bmatrix},$$

where u and v are such that $\widehat{p} = uc$ and $\widehat{q} = vc$.

Then, we find c by solving the system of equations (see Exercise 5.1)

$$\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} \mathscr{T}_{\mathtt{d}+1}^\top(u) \\ \mathscr{T}_{\mathtt{d}+1}^\top(v) \end{bmatrix} c, \qquad ((u,v) \mapsto c)$$

where $\mathscr{T}_{\mathtt{n}}(c)$ is the upper triangular Toeplitz matrix ($\mathscr{T}$), defined on page 117. Note, however, that $((u,v) \mapsto c)$ is an overdetermined system of equations, so that generically it has no solution. We solve it approximately in the least squares sense. This gives us the two-step Algorithm 7 for computing a suboptimal initial approximation. The first step is an unstructured low-rank approximation for the computation of z and the second step is a least squares approximation for the computation of c.

Algorithm 7 Suboptimal approximate common factor computation.

Input: Polynomials p and q and a positive integer $\mathtt{d}$.
1: Solve the unstructured low-rank approximation problem (ACF-LRA).
2: Let $\left[\begin{smallmatrix} v \\ -u \end{smallmatrix}\right] := z$, where $u, v \in \mathbb{R}^{\mathtt{n}-\mathtt{d}+1}$.
3: Solve the least squares problem $((u,v) \mapsto c)$.
Output: The solution c of the least squares problem.

5.3.3 Equivalent optimization problem

By definition, the polynomial c is a common factor of $\widehat{p}$ and $\widehat{q}$ if there are polynomials u and v, such that $\widehat{p} = uc$ and $\widehat{q} = vc$. Using the auxiliary variables u and v, Problem 5.17 becomes

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{p}, \widehat{q}, u, v, c \quad \left\| \begin{bmatrix} p \\ q \end{bmatrix} - \begin{bmatrix} \widehat{p} \\ \widehat{q} \end{bmatrix} \right\| \\ &\text{subject to} \quad \widehat{p} = uc \quad \text{and} \quad \widehat{q} = vc. \end{aligned} \tag{ACF}$$

As in problems related to data modeling, the constraint of (ACF) is bilinear in the optimization variables. Using the bilinear structure, we eliminate the optimization variables $\widehat{p}$, $\widehat{q}$, u, and v. This gives an equivalent optimization problem over c only.

Theorem 5.22. *The optimization problem (ACF) is equivalent to*

$$\text{minimize} \quad \text{over } c_0, \ldots, c_{\mathtt{d}-1} \in \mathbb{R} \quad M(c), \tag{ACF'}$$

where

$$M(c) := \sqrt{\text{trace}\left(\begin{bmatrix} p & q \end{bmatrix}^\top \left(I - \mathscr{T}_{\mathtt{n}+1}(c) \left(\mathscr{T}_{\mathtt{n}+1}^\top(c) \mathscr{T}_{\mathtt{n}+1}(c) \right)^{-1} \mathscr{T}_{\mathtt{n}+1}^\top(c) \right) \begin{bmatrix} p & q \end{bmatrix} \right)}.$$

The value of $M(c) = \text{dist}(\mathscr{D}, \widehat{\mathscr{D}})$ is the approximation errors in taking c as an approximate common factor of p and q. The equivalent problem (ACF') is a nonlinear least squares problem and is solved by standard local optimization methods, see Algorithm 8. Optionally, the "certificate" $\widehat{p}$ and $\widehat{q}$ for c being an approximate common factor of p and q with approximation error $M(c)$ is computed.

Algorithm 8 Optimal approximate common factor computation.

Input: Polynomials p and q and a positive integer $\mathtt{d}$.
1: Compute an initial approximation $c_{\text{ini}} \in \mathbb{R}^{\mathtt{d}+1}$ using Algorithm 7.
2: Solve (ACF') using a standard local optimization method.
3: **if** $\widehat{p}$ and $\widehat{q}$ are required **then**
4: Solve $\begin{bmatrix} p & q \end{bmatrix} = \mathscr{T}_{\mathtt{n}-\mathtt{d}+1}(c) \begin{bmatrix} u & v \end{bmatrix}$ for u and v. $\{c \mapsto (u, v)\}$
5: Define $\widehat{p} = \mathscr{T}_{\mathtt{d}+1}(u)c$ and $\widehat{q} = \mathscr{T}_{\mathtt{d}+1}(v)c$. $\{(u, v) \mapsto (\widehat{p}, \widehat{q})\}$
6: **end if**
Output: The approximate common factor $c \in \mathbb{R}^{\mathtt{d}+1}$ and the approximating polynomials $\widehat{p}$ and $\widehat{q}$.

5.3.4 Distance to uncontrollability

As an application of the approximate common factor computation in systems and control, next, we consider the distance to uncontrollability problem. A state space representation $\mathscr{B}_{\text{i/s/o}}(A, B, C, D)$ of a linear time-invariant system $\mathscr{B}$ is *state controllable* if and only if the controllability matrix $\mathscr{C}(A, B)$ is full rank. Therefore, the question of whether a given state space representation is state controllable is a rank test problem for the structured matrix $\mathscr{C}(A, B)$. Arbitrary small perturbations of the system's parameters, however, can switch the controllability property. This issue is addressed by the notion of distance to uncontrollability, which is quantitative rather

than qualitative measure of controllability. A quantitative measure for the distance of $\mathscr{C}(A, B)$ to rank deficiency is the size of the smallest $(\Delta A, \Delta B)$, such that

$$\mathscr{C}(\widehat{A}, \widehat{B}) := \mathscr{C}(A, B) + \mathscr{C}(\Delta A, \Delta B)$$

is rank deficient. Paige (1981) defined the distance to uncontrollability as

$$\begin{aligned} d'_{\text{ctrb}}(A, B) := \text{minimize} \quad &\text{over } \widehat{A}, \widehat{B} \quad \left\| \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \right\|_{\text{F}} \\ &\text{subject to} \quad (\widehat{A}, \widehat{B}) \text{ is uncontrollable.} \end{aligned}$$

The measure $d'_{\text{ctrb}}(A, B)$, however, is not invariant of the state space representation because it depends on the choice of basis. We resolve this issue in the behavioral setting, where controllability is defined as a property of the system rather then as a property of a particular representation.

Recall the definition of controllability in the behavioral setting, given on page 49. Checking the controllability property in practice is done by performing a numerical test on the parameters of the system's representation. In the case of a single-input single-output system, defined by an input/output representation $\mathscr{B} = \mathscr{B}_{\text{i/o}}(p, q)$, $\mathscr{B}$ is controllable if and only if p and q are co-prime.

Theorem 5.23 (Polderman and Willems (1998), Theorem 5.2.11). *Consider the polynomials p and q and let $\deg(p) \geq \deg(q)$. The system $\mathscr{B}_{\text{i/o}}(p, q)$ is controllable if and only if p and q are co-prime.*

Let $\overline{\mathscr{L}_{\text{ctrb}}}$ be the set of uncontrollable linear time-invariant systems and

$$\text{dist}\left(\mathscr{B}_{\text{i/o}}(p, q), \mathscr{B}_{\text{i/o}}(\widehat{p}, \widehat{q}) \right) := \left\| \begin{bmatrix} q \\ p \end{bmatrix} - \begin{bmatrix} \widehat{q} \\ \widehat{p} \end{bmatrix} \right\|. \tag{dist}$$

In order to avoid the nonuniqueness of the parameters (p, q) in $\mathscr{B}_{\text{i/o}}(p, q)$, without loss of generality, we assume in the definition of (dist) that p and $\widehat{p}$ are monic.

The distance to uncontrollability in the behavioral setting is defined as follows.

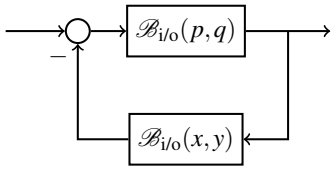
Problem 5.24. Given a controllable system $\mathscr{B}_{\text{i/o}}(p, q)$, find

$$d_{\text{ctrb}}(\mathscr{B}) := \min_{\widehat{\mathscr{B}} \in \overline{\mathscr{L}_{\text{ctrb}}}} \text{dist}(\mathscr{B}, \widehat{\mathscr{B}}).$$

We refer to $d_{\text{ctrb}}(\mathscr{B})$ as the *controllability radius*. The problem of computing $d_{\text{ctrb}}(\mathscr{B})$ is a special case of the approximate common factor problem 5.17, when $\mathtt{d} = 1$. Algorithm 8, then, solves a univariate optimization.

5.4 Pole placement by a low-order controller

Consider the single-input single-output feedback control system:



The plant $\mathscr{B}_{\text{i/o}}(p,q)$ is assumed to be causal and controllable. This implies that the polynomials p and q specifying the plant are co-prime and satisfy the constraint $\deg(q) \leq \deg(p) =: \ell_p$,. The polynomials x and y parameterize the controller and are unknowns. The design constraints are that the controller should be causal and have order bounded by a specified integer ℓ_x, *i.e.*,

$$\deg(y) \leq \deg(x) =: \ell_x < \ell_p. \tag{deg}$$

The pole placement problem is to determine x and y, so that the poles of the closed-loop system are as close as possible to desired locations, specified by the roots of a polynomial f, where $\deg(f) = \ell_x + \ell_p$. We consider a modification of the pole placement problem that aims to assign exactly the poles of a plant that is as close to the given plant as possible.

Problem 5.25 (Pole placement by low-order controller). Given

1. the plant $\mathscr{B}_{\text{i/o}}(p,q)$
2. a polynomial f, whose roots are the desired poles of the closed-loop system, and
3. a bound $\ell_x < \deg(p)$ on the order of the controller,

find a controller $\mathscr{B}_{\text{i/o}}(x,y)$, such that

1. the degree constraint (deg) is satisfied and
2. the controller assigns the poles of a system $\mathscr{B}_{\text{i/o}}(\widehat{p},\widehat{q})$, which is as close as possible to $\mathscr{B}_{\text{i/o}}(p,q)$ in the sense that (dist) is minimized.

Next, we write down explicitly the considered optimization problem. The closed-loop system is $\mathscr{B}_{\text{i/o}}(px+qy,qx)$, so that a solution to the pole placement problem is given by a solution to the Diophantine equation $px+qy=f$. Written in a matrix form, it is a Sylvester structured system of equations

$$\underbrace{\begin{bmatrix} p_0 & & & q_0 & & \\ p_1 & \ddots & & q_1 & \ddots & \\ \vdots & \ddots & p_0 & \vdots & \ddots & q_0 \\ p_{\ell_p} & & p_1 & q_{\ell_p} & & q_1 \\ & \ddots & \vdots & & \ddots & \vdots \\ & & p_{\ell_p} & & & q_{\ell_p} \end{bmatrix}}_{\mathscr{R}_{\ell_P-\ell_X}(p,q)} \begin{bmatrix} x_0 \\ \vdots \\ x_{\ell_x} \\ y_0 \\ \vdots \\ y_{\ell_x} \end{bmatrix} = \underbrace{\begin{bmatrix} f_0 \\ \vdots \\ f_{\ell_p} \\ f_{\ell_p+1} \\ \vdots \\ f_{\ell_p+\ell_x} \end{bmatrix}}_{f}.$$

The system is overdetermined due to the degree constraint (deg). Therefore, the pole placement by low-order controller problem 5.25 can be written as

$$\begin{aligned} \text{minimize} \quad & \text{over } \widehat{p},\, \widehat{q} \in \mathbb{R}^{\ell_p+1} \text{ and } x,\, y \in \mathbb{R}^{\ell_x+1} \quad \left\| \begin{bmatrix} p \\ q \end{bmatrix} - \begin{bmatrix} \widehat{p} \\ \widehat{q} \end{bmatrix} \right\| \\ \text{subject to} \quad & \mathscr{R}_{\ell_P-\ell_X}(\widehat{p},\widehat{q}) \begin{bmatrix} x \\ y \end{bmatrix} = f. \end{aligned}$$

Problem 5.25 is equivalent to Problem SLRA with

- data matrix
$$\mathscr{S}(p) = \begin{bmatrix} \begin{bmatrix} f_0 & f_1 & \cdots & f_{\ell_p+\ell_x} \end{bmatrix} \\ \mathscr{R}^{\top}_{\ell_P-\ell_X}(p,q) \end{bmatrix},$$
composed of a fixed block and a Sylvester structured block and
- rank reduction by one.

Indeed, $\mathscr{R}_{\ell_P-\ell_X}(\widehat{p},\widehat{q}) \begin{bmatrix} x \\ y \end{bmatrix} = f$ is equivalent to rank $\big(\mathscr{S}(\widehat{p})\big) \leq 2\ell_x + 1$.

5.5 Notes and references

System theory

A survey on applications of structured low-rank approximation in system identification and signal processing is given in (De Moor, 1993) and (Markovsky, 2008). Approximate realization is a special identification problem (the input is a pulse and the initial conditions are zeros). Nevertheless, the exact version of this problem is a well studied problem. Approximate realization methods, based on the singular value decomposition, are proposed in (Kung, 1978; Zeiger and McEwen, 1974).

Comprehensive treatment of model reduction methods is given in (Antoulas, 2005). The balanced model reduction method is proposed by Moore (1981) and error bounds are derived by Glover (1984). Proper orthogonal decomposition is a popular method for nonlinear model reduction. This method is unstructured low-rank approximation of a matrix composed of "snapshots" of the state vector of the system. The method is data-driven in the sense that the method operates on data of the full order system and a model of that system is not derived.

Errors-in-variables system identification methods are developed in (Aoki and Yue, 1970; Markovsky et al, 2005; Pintelon et al, 1998). Their consistency properties are studied in (Kukush et al, 2005; Markovsky and Pintelon, 2015; Pintelon and Schoukens, 2001). For a survey, see (Söderström, 2007). Most of the work on the subject is presented in the classical input/output setting, *i.e.*, the proposed methods are defined in terms of transfer function, matrix fraction description, or input/state/output representations.

Modeling by the orthogonal distance fitting criterion (misfit approach) is initiated in (Willems, 1987) and further developed in (Markovsky et al, 2005; Roorda, 1995a,b; Roorda and Heij, 1995), where algorithms for solving the problems are developed. A proposal for combination of misfit and latency for linear time-invariant system identification is made in (Lemmerling and De Moor, 2001).

Signal processing

Linear prediction methods based on optimization techniques are developed in (Bresler and Macovski, 1986; Cadzow, 1988). Cadzow (1988) proposed a method for Hankel structured low-rank approximation that alternates between unstructured low-rank approximation and structure approximation. This method however does not converge to a locally optimal solution (De Moor, 1994). Application of structured low-rank approximation methods for audio processing is described in (Lemmerling et al, 2003). The shape from moments problem (Golub et al, 1999; Milanfar et al, 1995) is equivalent to Hankel structured low-rank approximation.

Approximate polynomial common factor

There is a vast amount of literature on the problem of computing approximate common factors of polynomials, see (Usevich and Markovsky, 2017) for an overview. Some authors, *e.g.*, (Bini and Boito, 2010; Rupprecht, 1999), considered the what is called ε-GCD problem: Given polynomials p and q and a tolerance $\varepsilon > 0$,

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{p} \text{ and } \widehat{q} \quad \deg\left(\operatorname{GCD}(\widehat{p},\widehat{q})\right) \\ \text{subject to} & \left\| \begin{bmatrix} p \\ q \end{bmatrix} - \begin{bmatrix} \widehat{p} \\ \widehat{q} \end{bmatrix} \right\|_2 \leq \varepsilon. \end{array}$$

The ε-GCD problem is equivalent to Problem 5.17, considered in the book. Both problem formulations trace the same Pareto-optimal curve in the approximation error vs GCD degree space by varying the hyper parameters ε and $\mathtt{d}$.

The problem of Section 5.3.4 falls into a broader category of *distance problems* (Higham, 1989), such as distance to instability, distance to positive definiteness, *etc*. There is a big volume of literature devoted to the computation of the distance to controllability, see, *e.g.*, (Eising, 1984; Karow and Kressner, 2009; Khare et al, 2012). Other applications of the approximate common factor computation are:

- common dynamics computation (Papy et al, 2006),
- blind finite impulse response system identification (Xu et al, 1995), and
- computation of a minimal kernel representation (Polderman and Willems, 1998).

Exercises

5.1 (Matrix representation of polynomial multiplication). There is a natural correspondence among $(m+1)$-dimensional vectors $\begin{bmatrix} a_0 & a_1 & \cdots & a_m \end{bmatrix}^\top$, m-th order polynomials $a_0 + a_1 z + \cdots + a_m z^m$, and $(m+1)$-taps sequences $(a_0, a_1, \ldots, a_m)$. The same symbol a denotes the vector, the polynomial, and the sequence. For a given $a \in \mathbb{R}^{m+1}$ and a natural number n, $\mathscr{M}_n(a)$ is the $(m+n+1) \times (n+1)$ matrix

$$\mathscr{M}_n(a) := \begin{bmatrix} a_0 & & \\ a_1 & \ddots & \\ \vdots & \ddots & a_0 \\ a_m & & a_1 \\ & \ddots & \vdots \\ & & a_m \end{bmatrix} = \mathscr{T}_{n+1}^\top(a).$$

1. Show that the product of two polynomials $c(z) = a(z)b(z)$ corresponds to the matrix-vector product $c = \mathscr{M}_n(a)b$, where n is the degree of b.
2. Show that the convolution $c = a \star b$ of two sequences—a with $m+1$ taps and b with $n+1$ taps—defined as $c_i := \sum_{k=0}^{m} a_k b_{i-k}$, for $i = 0, \ldots, m+n$, where $b_k = 0$ for $k < 0$ and $k > n$, corresponds to the matrix-vector product $c = \mathscr{M}_n(a)b$.
3. Show that $c = a(\sigma)b$—the action of a difference operator defined by a polynomial a of degree m on a sequence b with $n+1$ taps—defined by

$$c_i = a_0 b_i + a_1 b_{i+1} + \cdots + a_m b_{i+m}, \quad \text{for } i = 0, 1, \ldots, n-m,$$

corresponds to the matrix-vector product $c = \mathscr{M}_{n-m}^\top(a)b = \mathscr{T}_{n-m+1}(a)b$.

5.2 (Computing approximate common factor with `slra`).
By Theorem 5.20, the approximate common factor problem 5.17 is equivalent to the reduced Sylvester structured low-rank approximation problem (ACF-$\mathscr{R}_\mathtt{d}$). Use the `slra` function in order to solve (ACF-$\mathscr{R}_\mathtt{d}$).

5.3 (Approximate common factor numerical example).
Apply the method for computing approximate common factor, based on the `slra` function (Exercise 5.2), on an example with $\mathtt{d} = 2$ and polynomials

$$\begin{aligned} p(z) &= (4 + 2z + z^2)(5 + 2z) + 0.05 + 0.03z + 0.04z^2 \\ q(z) &= (4 + 2z + z^2)(5 + \ z) + 0.04 + 0.02z + 0.01z^2 \end{aligned}$$

Compare the results obtained by `slra` with the ones obtained by Algorithm 8.

5.4 (SLRA package).

- Download and install the SLRA package from `http://slra.github.io/`.
- Study and execute the demo files `demo.m` and `test_m/demo.m`.

5.5 (IDENT and AGCD packages). The IDENT and AGCD packages provide rapper functions to the SLRA package for the purpose of linear time-invariant system identification and approximate polynomial common factor computation. These packages are installed together with the SLRA package (see the sub-directories `ident` and `agcd`). Study and redo the examples available at

`http://slra.github.io/software-ident.html`.

References

Antoulas A (2005) Approximation of Large-Scale Dynamical Systems. SIAM

Aoki M, Yue P (1970) On a priori error estimates of some identification methods. IEEE Trans Automat Contr 15(5):541–548

Bini D, Boito P (2010) A fast algorithm for approximate polynomial GCD based on structured matrix computations. In: Numerical Methods for Structured Matrices and Applications, Birkhäuser, pp 155–173

Bresler Y, Macovski A (1986) Exact maximum likelihood parameter estimation of superimposed exponential signals in noise. IEEE Trans Acust, Speech, Signal Process 34:1081–1089

Cadzow J (1988) Signal enhancement—A composite property mapping algorithm. IEEE Trans Signal Proc 36:49–62

De Moor B (1993) Structured total least squares and L_2 approximation problems. Linear Algebra Appl 188–189:163–207

De Moor B (1994) Total least squares for affinely structured matrices and the noisy realization problem. IEEE Trans Signal Proc 42(11):3104–3113

Eising R (1984) Distance between controllable and uncontrollable. Control Lett 4:263—264

Glover K (1984) All optimal Hankel-norm approximations of linear multivariable systems and their l^∞-error bounds. Int J Control 39(6):1115–1193

Golub G, Milanfar P, Varah J (1999) A stable numerical method for inverting shape from moments. SIAM J Sci Comput 21:1222–1243

Higham N (1989) Matrix nearness problems and applications. In: Gover M, Barnett S (eds) Applications of Matrix Theory, Oxford University Press, pp 1–27

Karow M, Kressner D (2009) On the structured distance to uncontrollability. Control Lett 58:128–132

Khare S, Pillai H, Belur M (2012) Computing the radius of controllability for state space systems. Control Lett 61:327–333

Kukush A, Markovsky I, Van Huffel S (2005) Consistency of the structured total least squares estimator in a multivariate errors-in-variables model. J Statist Plann Inference 133(2):315–358

Kung S (1978) A new identification method and model reduction algorithm via singular value decomposition. In: Proc. 12th Asilomar Conf. Circuits, Systems, Computers, Pacific Grove, pp 705–714

Lemmerling P, De Moor B (2001) Misfit versus latency. Automatica 37:2057–2067

Lemmerling P, Mastronardi N, Van Huffel S (2003) Efficient implementation of a structured total least squares based speech compression method. Linear Algebra Appl 366:295–315

MacFarlane A (2013) Multivariable feedback: a personal reminiscence. International Journal of Control 86(11):1903–1923

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

Markovsky I, Pintelon R (2015) Identification of linear time-invariant systems from multiple experiments. IEEE Trans Signal Process 63(13):3549–3554

Markovsky I, Rao S (2008) Palindromic polynomials, time-reversible systems, and conserved quantities. In: 16th Mediterranean Conf. on Control and Automation, Ajaccio, France, pp 125–130

Markovsky I, Willems JC, Van Huffel S, De Moor B, Pintelon R (2005) Application of structured total least squares for system identification and model reduction. IEEE Trans Automat Control 50(10):1490–1500

Milanfar P, Verghese G, Karl W, Willsky A (1995) Reconstructing polygons from moments with connections to array processing. IEEE Trans Signal Proc 43:432–443

Moore B (1981) Principal component analysis in linear systems: Controllability, observability and model reduction. IEEE Trans Automat Contr 26(1):17–31

Paige CC (1981) Properties of numerical algorithms related to computing controllability. IEEE Trans Automat Contr 26:130–138

Papy JM, Lathauwer LD, Huffel SV (2006) Common pole estimation in multichannel exponential data modeling. Signal Processing 86(4):846–858

Pintelon R, Schoukens J (2001) System Identification: A Frequency Domain Approach. IEEE Press, Piscataway, NJ

Pintelon R, Guillaume P, Vandersteen G, Rolain Y (1998) Analyses, development, and applications of TLS algorithms in frequency domain system identification. SIAM J Matrix Anal Appl 19(4):983–1004

Polderman J, Willems JC (1998) Introduction to Mathematical Systems Theory. Springer-Verlag, New York

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

Roorda B (1995b) Global total least squares—a method for the construction of open approximate models from vector time series. PhD thesis, Tinbergen Institute

Roorda B, Heij C (1995) Global total least squares modeling of multivariate time series. IEEE Trans Automat Contr 40(1):50–63

Rupprecht D (1999) An algorithm for computing certified approximate GCD of n univariate polynomials. J Pure Appl Algebra 139(1–3):255–284

Söderström T (2007) Errors-in-variables methods in system identification. Automatica 43:939–958

Usevich K, Markovsky I (2017) Variable projection methods for approximate (greatest) common divisor computations. Theoretical Computer Science

Willems JC (1987) From time series to linear system—Part III. Approximate modelling. Automatica 23(1):87–115

Xu G, Liu H, Tong L, Kailath T (1995) A least-squares approach to blind channel identification 43(12):2982–2993

Zeiger H, McEwen A (1974) Approximate linear realizations of given dimension via Ho's algorithm. IEEE Trans Automat Contr 19:153–153

Chapter 6
Data-driven filtering and control

If the model of a system is exact, it is optimal for all applications. However, if the model is only an approximation of the "true system", then the quality of the model should be dependent on the intended application.

Gevers (2004)

The bottleneck in solving real-life data processing problems, such as dynamic measurement in metrology, noise cancellation in acoustics, and ranking papers in scientometrics is obtaining an adequate model for the data generating process. Classical modeling methods ignore the subsequent usage of the model for design of a predictor, controller, or classifier. This issue is often addressed by trial-and-error human interaction. The approach presented in this chapter is to merge the data modeling and model-based design subproblems into one joint problem, called *data-driven design*. Its benefits are optimality of the overall design and automation of the design process, which reduce the cost and increase the overall design reliability.

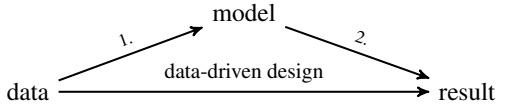
The chapter is organized as follows. Section 6.1 gives motivation and introduction to data-driven signal processing and control. The main idea—posing data-driven problems as missing data estimation—is informally presented in Section 6.2. Specific examples that illustrate the idea are shown in Section 6.3. Section 6.4 describes the solution approach. First, we establish the equivalence of the data-driven problem and a weighted structured low-rank matrix approximation and completion problem. For the solution of the latter problem we use the variable projection method of Section 4.4. A simulation example of data-driven impulse response simulation illustrate the theoretical properties and compares the method based on the variable projection with a subspace method as well as a classical model-based method.

6.1 Model-based vs data-driven paradigms

State-of-the-art signal processing and control methods are model-based. This means that, first, a model class is selected using prior knowledge and observed data. Then, model parameters are estimated using the data. Finally, the signal processing or control task is solved using the identified model and the problem specification.

The model-based approach splits the original problem into two steps:

1. *model identification* and
2. *model-based design*.



The identification step simplifies the design problem but does not always take into account the design objectives. This leads to *suboptimal performance of the overall design*. The suboptimality issue is addressed by repeating steps 1 and 2 with human interaction (*ad hoc* adjustment of the identification criterion). Although iterative adjustment of the identification method and redesign is common, it is inefficient, unreliable, or even impossible to use when optimal performance is desired and the data generating process is a complex multivariable dynamical system.

An alternative to the model-based approach is to solve the filtering or control problem directly without first identifying a model, see Figure 6.1. From applications' point of view, this data-driven approach is closer to the real-life problem than the model-based approach. Indeed, in practice model parameters are rarely given, but data may often be observed. From the theoretical point of view, a data-driven approach opens up the possibility for a new class of solution methods and algorithms not based on an explicit model representation of the data generating process.

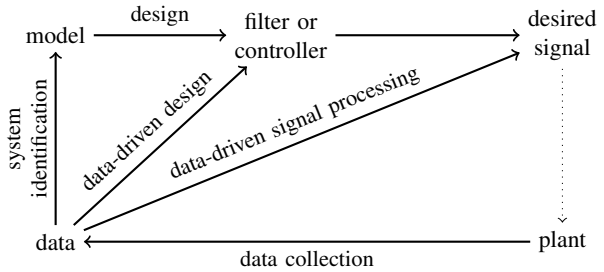


Fig. 6.1: Data-driven methods bypass the model identification step. Such method map plant data to filter/controller or directly to the desired filtered/control signal.

6.2 Missing data approach

The classical motivation for missing data in signal processing and control problems is sensor failures, where measurements are *accidentally* corrupted. More recently, missing data estimation is used for compressive sensing, where measurements are *intentionally* skipped. In this chapter, we use missing data estimation for solving data-driven estimation and control problems, *i.e.*, the missing data represents the signal that we *aim to find* on the first place. Examples are initial state state estimation, prediction/filtering/smoothing, partial realization, and optimal tracking control.

We pose the data-driven design problem as the problem of finding a missing part of a trajectory of a linear time-invariant system, where other parts of the trajectory are given and are exact or are approximated. Once the problem is formulated as a missing data estimation, it is shown to be equivalent to an element-wise weighted mosaic-Hankel structured low-rank matrix approximation and completion problem. We use the method based on the variable projection principle, see Section 4.4.

The methodology for design via missing data estimation is illustrated on the example of forecasting, see Figure 6.2. The data generating process is second order, autonomous, discrete-time, linear time-invariant. The data y is collected over a (past) period of time $[1,t]$ and the goal is to predict y over a (future) period of time, *e.g.*, $[t+1,2t]$. The classical model-based approach is: 1. using identification methods, find a state-space representation of the data generating system, 2. using the identified state-space model, estimate the initial condition, 3. using the model and the estimated initial condition, simulate the future response. The missing data estimation approach is a *Hankel structured rank constrained matrix completion* problem:

$$\text{find } y(t+1),\ldots,y(2t) \text{ such that } \operatorname{rank}\left(\begin{bmatrix} y(1) & y(2) & \cdots & y(t) \\ y(2) & & \iddots & y(t+1) \\ \vdots & \iddots & \iddots & \vdots \\ y(t) & y(t+1) & \cdots & y(2t) \end{bmatrix}\right) \leq 2.$$

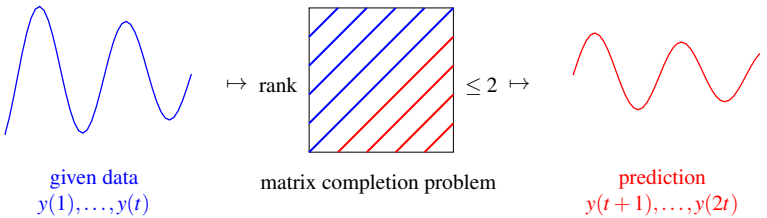


Fig. 6.2: The predicted signal is found by completing a Hankel matrix with the constraint that the matrix has a specified rank. The part of the matrix shown in red is to be computed, in such a way that the completed matrix has rank at most two.

6.3 Estimation and control examples

First, we state the classical model-based problems. Then, we state the corresponding data-driven problems, where the system $\mathscr{B}$ is implicitly specified by data w_{d}. In both cases the problem is formulated as missing data estimation in terms of a generic trajectory $w = \big(w(1),\ldots,w(T)\big)$ of the system $\mathscr{B}$, with an input/output partition $\Pi w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ and past/future partition $w = w_{\mathrm{p}} \wedge w_{\mathrm{f}}$. The "past" of the trajectory consists of the first T_{p} samples and is used for estimation or specification of the initial conditions for the "future", which consists of the remaining T_{f} samples (see Table 6.1). In data-driven filtering and control, each of the elements u_{p}, y_{p}, u_{f}, and y_{f} of the trajectory w is either exact, inexact, or missing, depending on the particular problem. (For the specification of initial conditions by a trajectory, see Exercise 2.4.)

Table 6.1: Partitioning of the trajectory w into input u and output y variables and past "p" and future "f" time horizons.

	past	future
input	u_{p}	u_{f}
output	y_{p}	y_{f}

6.3.1 Problem statement with a given model

The classical model-based state estimation problem is defined as follows: Given a linear time-invariant system $\mathscr{B}$ and an exact trajectory w_{f} of the system $\mathscr{B}$,

$$\text{find } w_{\mathrm{p}}, \text{ such that } w = w_{\mathrm{p}} \wedge w_{\mathrm{f}} \in \mathscr{B}. \tag{SE}$$

The aim of (SE) is to estimate the "past", *i.e.*, the first T_{p} samples of a trajectory $w = w_{\mathrm{p}} \wedge w_{\mathrm{f}}$, with the "future", *i.e.*, the last T_{f} samples w_{f} known exactly.

If w_{f} is not a trajectory of $\mathscr{B}$, the model-based state estimation problem becomes the Kalman smoothing problem. The classical Kalman smoother assumes that the output y_{f} is noisy and the input u_{f} is exact. The approximation problem then is

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{w}_{\mathrm{p}} \text{ and } \widehat{y}_{\mathrm{f}} \quad \|y_{\mathrm{f}} - \widehat{y}_{\mathrm{f}}\|_2 \\ &\text{subject to} && \widehat{w}_{\mathrm{p}} \wedge (u_{\mathrm{f}}, \widehat{y}_{\mathrm{f}}) \in \mathscr{B}. \end{aligned} \tag{OE-KS}$$

As a byproduct of computing the initial conditions estimate $\widehat{w}_{\mathrm{p}}$, (OE-KS) determines an approximation of the output $\widehat{y}_{\mathrm{f}}$ (the smoothed output), which is the best estimate of the noisy output y_{f}, given the model $\mathscr{B}$. Problem (OE-KS) is also a missing data estimation problem, however, the output y_{f} is approximated rather than fitted exactly.

When both u_{f} and y_{f} are noisy (inexact), the Kalman smoothing problem becomes

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{w}_{\mathrm{p}} \text{ and } \widehat{w}_{\mathrm{f}} \quad \|w_{\mathrm{f}} - \widehat{w}_{\mathrm{f}}\|_2 \\ &\text{subject to} && \widehat{w}_{\mathrm{p}} \wedge \widehat{w}_{\mathrm{f}} \in \mathscr{B}. \end{aligned} \tag{EIV-KS}$$

It is refered to as the errors-in-variables Kalman smoother. The solution is given by a modification of the ordinary Kalman smoother (Markovsky and De Moor, 2005).

6.3.2 Problem statement without a given model

The data-driven version of the state estimation problem is: Given trajectories w_d and w_f of a linear time-invariant system $\mathscr{B}$,

$$\text{find } w_\mathrm{p}, \text{ such that } w = w_\mathrm{p} \wedge w_\mathrm{f} \in \mathscr{B}_{\mathrm{mpum}}(w_\mathrm{d}). \tag{DD-SE}$$

Although the problem formulation (DD-SE) involves the most powerful unfalsified model $\mathscr{B}_{\mathrm{mpum}}(w_\mathrm{d})$, solution methods need not identify explicitly a representation of $\mathscr{B}_{\mathrm{mpum}}(w_\mathrm{d})$ in order to find the quantity of interest w_p. For example, the methods based on the nuclear norm heuristic do not involve in any way a model representation.

When w_d is inexact, prior knowledge about the model is needed in order to make the design problem well posed. Often, the prior knowledge is the model class $\mathscr{L}_{\mathtt{m},\ell}$ to which the unknown data generating system belongs, *i.e.*, the model complexity $(\mathtt{m},\ell)$ is a priori specified. With this prior knowledge, the data-driven versions of the state estimation problems (OE-KS) and (EIV-KS) is well posed and becomes

$$\begin{aligned} &\text{minimize over } \widehat{w}_\mathrm{d} \text{ and } \widehat{y}_\mathrm{f} \quad \underbrace{\|y_\mathrm{f} - \widehat{y}_\mathrm{f}\|_2^2}_{\text{estimation error}} + \underbrace{\|w_\mathrm{d} - \widehat{w}_\mathrm{d}\|_2^2}_{\text{identification error}} \\ &\text{subject to} \quad (u_\mathrm{f}, \widehat{y}_\mathrm{f}) \in \mathscr{B}_{\mathrm{mpum}}(\widehat{w}_\mathrm{d}) \in \mathscr{L}_{\mathtt{m},\ell} \end{aligned} \tag{DD-OE-KS}$$

in the output error setting, when the input is exact, and

$$\begin{aligned} &\text{minimize over } \widehat{w}_\mathrm{d} \text{ and } \widehat{w} \quad \underbrace{\|w_\mathrm{f} - \widehat{w}_\mathrm{f}\|_2^2}_{\text{estimation error}} + \underbrace{\|w_\mathrm{d} - \widehat{w}_\mathrm{d}\|_2^2}_{\text{identification error}} \\ &\text{subject to} \quad \widehat{w} \in \mathscr{B}_{\mathrm{mpum}}(\widehat{w}_\mathrm{d}) \in \mathscr{L}_{\mathtt{m},\ell}, \end{aligned} \tag{DD-EIV-KS}$$

in the errors-invariables setting, when both the inputs and the outputs are noisy.

The classical approach for state estimation involves the two steps.

1. *Identification*
   Given w_d and $(\mathtt{m},\ell)$, compute a representation of the model $\mathscr{B} = \mathscr{B}_{\mathrm{mpum}}(\widehat{w}_\mathrm{d})$, where $\widehat{w}_\mathrm{d}$ is the data w_d, when w_d is exact or a solution to an optimization problem

   $$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{y}_\mathrm{d} \quad \|y_\mathrm{d} - \widehat{y}_\mathrm{d}\|_2 \\ &\text{subject to} \quad \mathscr{B}_{\mathrm{mpum}}\big((u_\mathrm{d}, \widehat{y}_\mathrm{d})\big) \in \mathscr{L}_{\mathtt{m},\ell} \end{aligned} \tag{OE-ID}$$

   when u_d is exact but y_d is noisy (output error setting), or

   $$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{w}_\mathrm{d} \quad \|w_\mathrm{d} - \widehat{w}_\mathrm{d}\|_2 \\ &\text{subject to} \quad \mathscr{B}_{\mathrm{mpum}}(\widehat{w}_\mathrm{d}) \in \mathscr{L}_{\mathtt{m},\ell} \end{aligned} \tag{EIV-ID}$$

   when w_d is noisy (errors-invariables setting).

2. *Model-based design*
   Solve (SE), (OE-KS), or (EIV-KS), using the representation of $\mathscr{B}$ from step 1.

Note that the optimization criterion of the data-driven problem (DD-EIV-KS) involves a mixture of the identification and filtering/control errors, while the identification criteria (OE-ID) and (EIV-ID) are agnostic to the design objective.

Other examples that fit the generic approach for data-driven filtering/control, based on missing data estimation, are simulation, partial realization, and control.

- *Simulation:* Given initial conditions w_p and input u_f, the objective is to find the corresponding output y_f of the system, *i.e.*,

  $$\text{find } y_\mathrm{f}, \text{ such that } w_\mathrm{p} \wedge (u_\mathrm{f}, y_\mathrm{f}) \in \mathscr{B}. \tag{SIM}$$

- *Noisy partial realization:* given the first $T+1$ samples $H(0), H(1), \ldots, H(T)$ of an impulse response, the objective of the partial realization problem is to find the remaining samples $H(T+1), H(T+2), \ldots$
- *2-norm optimal output tracking:* given initial conditions w_p, and an output y_f, the objective is to find a control input $\widehat{u}_\mathrm{f}$, such that

  $$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{u}_\mathrm{f} \text{ and } \widehat{y}_\mathrm{f} \quad \|y_\mathrm{f} - \widehat{y}_\mathrm{f}\|_2 \\ &\text{subject to} \quad w_\mathrm{ini} \wedge (\widehat{u}_\mathrm{f}, \widehat{y}_\mathrm{f}) \in \mathscr{B}. \end{aligned} \tag{CTR}$$

  By construction, a solution $\widehat{u}_\mathrm{f}$ of (CTR) is 2-norm optimal tracking control signal.

Table 6.2: Different data-driven design examples fit into the missing data estimation setting by fixing different parts u_p, y_p, u_f, y_f of the trajectory $w \in \mathscr{B}$ as missing (?), exact (E), or noisy (N).

example	reference	u_p	y_p	u_f	y_f
simulation	(SIM)	E	E	E	?
partial realization	(Kalman, 1979)	E	E	E	E/?
state estimation	(SE)	?	?	E	E
classical smoothing	(OE-KS)	?	?	E	N
EIV smoothing	(EIV-KS)	?	?	N	N
noisy realization	(De Moor, 1994)	E	E	E	N/?
output tracking	(CTR)	E	E	?	N

6.4 Solution via matrix completion

In this section, we show the link between data-driven filtering/control and weighted mosaic-Hankel structured low-rank approximation and completion. The data-driven problems, considered in Section 6.3, aim to minimize the "size" of the error signal $e := w - \widehat{w}$, where w contains given data (exact or noisy) as well as missing values and $\widehat{w}$ is a trajectory of the system. As in Section 4.1.2, we encode information about exact, noisy, and missing data by the weights $v_i(t) \geq 0$ of the semi-norm

$$\|e\|_v := \sqrt{\sum_{t=1}^{T}\sum_{i=1}^{\mathtt{q}} v_i(t)e_i^2(t)}.$$

Table 6.3: The information about exact, noisy, and missing data elements $w_i(t)$ is encoded in the weights $v_i(t)$ of the semi-norm $\|\cdot\|_v$.

weight	used	to	by
$v_i(t) = \infty$	if $w_i(t)$ is exact	interpolate $w_i(t)$	$e_i(t) = 0$
$v_i(t) \in (0,\infty)$	if $w_i(t)$ is noisy	approximate $w_i(t)$	min $\|e_i(t)\|$
$v_i(t) = 0$	if $w_i(t)$ is missing	fill in $w_i(t)$	$\widehat{w} \in \widehat{\mathscr{B}}$

With this notation, the examples of data-driven problems, shown in Table 6.2, become special cases of the following generic problem

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{w}_{\mathrm{d}} \text{ and } \widehat{w} \quad \|w_{\mathrm{d}} - \widehat{w}_{\mathrm{d}}\|_2^2 + \|w - \widehat{w}\|_v^2 \\ &\text{subject to} && \widehat{w} \in \mathscr{B}_{\mathrm{mpum}}(\widehat{w}_{\mathrm{d}}) \in \mathscr{L}_{\mathtt{m},\ell}, \end{aligned} \tag{DD-SP}$$

for a suitable choice of the trajectory w and the weights v.

Note that the constraint of (DD-SP) is equivalent to the constraint

$$\text{there is } \widehat{\mathscr{B}} \in \mathscr{L}_{\mathtt{m},\ell}, \text{ such that } \widehat{w}_{\mathrm{d}}, \widehat{w} \in \widehat{\mathscr{B}},$$

i.e., both $\widehat{w}_{\mathrm{d}}$ and $\widehat{w}$ should be exact trajectories of a bounded complexity linear time-invariant system $\widehat{\mathscr{B}}$. For the application of structured low-rank approximation in system identification (see Section 5.2), we use Lemma 5.10 in order to relate the condition $w \in \mathscr{B} \in \mathscr{L}_{\mathtt{m},\ell}$ to a rank constraint of a Hankel matrix constructed from w. The following lemma generalizes this result to the case of two trajectories w^1 and w^2. The equivalent condition turns out to be rank deficiency of a mosaic-Hankel matrix

$$\mathscr{H}_{\ell+1}(w^1, w^2) := \begin{bmatrix} \mathscr{H}_{\ell+1}(w^1) & \mathscr{H}_{\ell+1}(w^2) \end{bmatrix},$$

i.e., a block matrix, which blocks are Hankel (Heinig, 1995).

Lemma 6.1 (Markovsky (2017)). *Let* $\mathtt{p}$ *and* ℓ *be, respectively, the number of outputs and the lag of a linear time-invariant system* $\mathscr{B}$*. Then,*

$$w^1, w^2 \in \mathscr{B} \quad \Longleftrightarrow \quad \operatorname{rank}\left(\mathscr{H}_{\ell+1}(w^1, w^2)\right) \leq \mathtt{q}\ell + \mathtt{m}.$$

Proof. Let $\mathscr{B} = \ker\left(R(z)\right)$ be a kernel representation of the system.

$$w^i \in \mathscr{B} \in \mathscr{L}_{\mathtt{m},\ell} \quad \Longleftrightarrow \quad \underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix}}_{R} \mathscr{H}_{\ell+1}(w^i) = 0, \text{ for } i = 1,2.$$

The $\mathtt{p} \times \mathtt{q}(\ell+1)$ matrix R is full row-rank. Then

$$\begin{cases} R \in \mathbb{R}^{\mathtt{p}\times\mathtt{q}(\ell+1)} \text{ full row rank} \\ R\begin{bmatrix} \mathscr{H}_{\ell+1}(w^1) & \mathscr{H}_{\ell+1}(w^2) \end{bmatrix} = 0 \end{cases} \quad \Longleftrightarrow \quad \operatorname{rank}\left(\mathscr{H}_{\ell+1}(w^1, w^2)\right) \leq \mathtt{q}\ell + \mathtt{m}.$$

Using Lemma 6.1, we obtain an equivalent weighted mosaic-Hankel structured low-rank approximation and completion problem to the data-driven problem (DD-SP).

Proposition 6.2 (Markovsky (2017)) *Problem (DD-SP) is equivalent to the weighted mosaic-Hankel structured low-rank matrix approximation and completion problem*

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{w}_{\mathrm{d}} \text{ and } \widehat{w} \quad \|w_{\mathrm{d}} - \widehat{w}_{\mathrm{d}}\|_2^2 + \|w - \widehat{w}\|_v^2 \\ &\textit{subject to} && \operatorname{rank}\left(\mathscr{H}_{\ell+1}(\widehat{w}_{\mathrm{d}}, \widehat{w})\right) \leq \mathtt{q}\ell + \mathtt{m}. \end{aligned} \tag{WSLRA}$$

In the next section, we show numerical results obtained with the structured low-rank approximation and completion method based on the variable projection.

Numerical example: data-driven impulse response simulation

Simulation is a classic problem in system theory and numerical linear algebra, for which many solutions exist, *e.g.*, for systems with no inputs, the problem is related to the computation of the matrix exponential. The classical simulation methods require a representation (state space, transfer function, convolution kernel, *etc.*) of the model. The data-driven version of the simulation problem (SIM) is a mosaic-Hankel structured low-rank matrix completion problem with fixed (exact) and missing data

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{w}_{\mathrm{d}} \text{ and } \widehat{w} \quad \|w_{\mathrm{d}} - \widehat{w}_{\mathrm{d}}\|_2 \\ &\text{subject to} && \operatorname{rank}\left(\begin{bmatrix} \mathscr{H}(\widehat{w}_{\mathrm{d}}) & \mathscr{H}(\widehat{w}) \end{bmatrix}\right) \leq \mathtt{q}\ell + \mathtt{m}, \\ & && \widehat{w}_{\mathrm{p}} = w_{\mathrm{p}}, \quad \text{and} \quad \widehat{u}_{\mathrm{f}} = u_{\mathrm{f}}. \end{aligned} \tag{DD SIM}$$

For the solution of (DD SIM), we use the variable projection method of Section 4.4 implementation in the SLRA package (see Exercises 5.4 and 5.5):

168a ⟨*Using the SLRA package for solving (DD SIM)* 168a⟩≡ (169e)

```
opt.exct = {[], 1}; opt.wini = {[], 0};
[sysh, info, wh] = ident({wd wf}, 1, n, opt); hh = wh{2}(:, 2);
```

In the simulation example, the data w_{d} is generating in the errors-in-variables setting (EIV), where the true system is

$$\bar{\mathscr{B}} = \{\,(u,y) \mid u - \sigma u + \sigma^2 u = 0.81y - 1.456\sigma y + \sigma^2 y\,\}$$

168b ⟨*Example data-driven impulse response estimation* 168b⟩≡ 169a▷

```
clear all, n = 2; Td = 100; Tf = 10; s = 0.01; rng('default')
sys0 = ss(tf([1 -1 1], [1 -1.456 0.81], 1));
ud0 = rand(Td, 1); yd0 = lsim(sys0, ud0); wd0 = [ud0 yd0];
wt = randn(Td, 2); wd = wd0 + 0.1 * wt / norm(wt) * norm(wd0);
```

and the to-be-found response y_f is the impulse response $\bar{h}$ of $\bar{\mathscr{B}}$. An estimate $\widehat{h}$ of $\bar{h}$ is validated in terms of the relative error $e := \|\bar{h} - \widehat{h}\|_2 / \|\bar{h}\|_2$.

169a ⟨*Example data-driven impulse response estimation* 168b⟩+≡ ◁168b 169e▷
```
h0 = impulse(sys0, Tf - 1);
e = @(hh) norm(h0 - hh) / norm(h0);
```

In order to map the special case of impulse response estimation into the general data-driven simulation problem (DD SIM), we define the trajectory $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ as

$$u = u_\mathrm{p} \wedge u_\mathrm{f} = (\overbrace{0,\ldots,0}^{\ell}, \overbrace{1,0,\ldots,0}^{t+1}), \quad y = y_\mathrm{p} \wedge y_\mathrm{f} = (\overbrace{0,\ldots,0}^{\ell}, \overbrace{h(0),h(1),\ldots,h(t)}^{t+1}).$$

i.e., the response of the system to the pulse input under zero initial conditions.

169b ⟨*Trajectory w in (DD SIM) for impulse response estimation* 169b⟩≡ (169e)
```
uf = zeros(Tf, 1); uf(1) = 1;
yf = NaN * ones(Tf, 1); wf = [uf yf];
```

The data-driven simulation method (DD SIM), using the SLRA package, is compared with the subspace-type method `uy2h` of Section 3.2:

169c ⟨*Subspace method for data-driven impulse response estimation* 169c⟩≡ (169e)
```
hh_ss = uy2h(wd(:, 1), wd(:, 2), 2, 2, Tf);
```

as well as model identification followed by model-based simulation:

169d ⟨*System identification followed by model-based simulation* 169d⟩≡ (169e)
```
[sysh_id, info_id] = ident(wd, 1, n);
hh_id = impulse(sysh_id, Tf - 1);
```

The obtained results in the simulation example

169e ⟨*Example data-driven impulse response estimation* 168b⟩+≡ ◁169a
```
⟨Trajectory w in (DD SIM) for impulse response estimation 169b⟩
⟨Using the SLRA package for solving (DD SIM) 168a⟩
⟨Subspace method for data-driven impulse response estimation 169c⟩
⟨System identification followed by model-based simulation 169d⟩
results = [e(hh) e(hh_id) e(hh_ss)]
```

show that the error `e(hh)` of the data-driven method is equal to the error `e(hh_id)` of the method identifying a model and simulating its impulse response. The reason for this is that the objective function of the data-driven problem coincides with the objective function of the identification problem. Indeed, $\|w - \widehat{w}\|_v = 0$ because w contains only exact and missing data. Another justification of the equivalence of the classical and the data-driven methods for data-driven simulation is that in this case the trajectory w_f does not carry information about the data generating system. Thus, the system identification and data-driven simulation methods use the same data.

The error `e(hh)` of the data-driven method, however, is smaller than the error `e(hh_ss)` of the subspace method `uy2h`. The reason for this is that the subspace method is suboptimal. Indeed, it does not use nonlinear optimization, while the other methods optimize cost functions that make them statistically optimal (maximum likelihood estimators) in the errors-in-variables setting.

6.5 Notes and references

Much work is done separately on identification and model-based design, but relatively little work on their interplay in solving the overall problem. The cliche "all models are wrong but some are useful" is true when the model-based methods are applied in practice, where there is no "true" model in the model class. The question occurs "*what is the best model for the problem at hand?*" The identification literature answers instead questions about the closeness of the identified model to a true model and does not take into account the subsequent usage of the model for model-based design, *e.g.*, noise filtering, prediction, and control.

The issue of developing identification methods aimed at their intended usage is considered in an area of research known as "identification for control", see, *e.g.*, (Ljung, 2002). The identified model is tuned for maximum performance of the closed-loop system, *i.e.*, the identification criterion is linked to the control objective. The interplay between identification and control is central also in adaptive control, where the modeling and control tasks are solved simultaneously, in real-time. Both identification for control and adaptive control, however, use model-based methods.

Data-driven control, also known as model-free control, has its roots in classical heuristics for proportional-integral-differential (PID) controller tuning such as the Ziegler–Nichols method (Ziegler and Nichols, 1942). Rigorous data-driven methods, however, appeared only in the late 90's (Chan, 1996; Favoreel, 1999; Hjalmarsson et al, 1998; Safonov and Tsao, 1997). Since then data-driven control has gained a lot of interest as evident by the large number of publications.

Although the particular problems considered range from LQG to fuzzy control, the corresponding methods developed can be classified into four main approaches:

- *Subspace-type data-driven methods* are proposed for solution of $\mathscr{H}_2/\mathscr{H}_\infty$ control problems in (Favoreel, 1999; Markovsky and Rapisarda, 2008; Shi and Skelton, 2000; Woodley, 2001). The signal of interest is constrained to belong to a subspace computed from the measured data only. Viewed abstractly, the subspace is a model for the signal, although it is not parameterized in a familiar transfer function, state-space, *etc*. form.
- Similarly to the subspace methods, the *virtual reference feedback tuning methods* (Campi et al, 2002; Van Heusden et al, 2011) design the controller without resorting on nonlinear optimization methods. Contrary to the subspace methods, however, they produce a feedback controller and operate recursively in real-time.
- An adaptive approach, called *controller unfalsification*, is developed in (Safonov and Cabral, 2001; Safonov and Tsao, 1997). The controller is viewed as an exclusion rule (Polderman and Willems, 1998) and the main idea is to reject (falsify) controllers using previously collected experimental data from the plant.
- In *iterative feedback tuning* the controller parameters are optimized by a gradient type method minimizing the control objective, which depends on measured data only (Hildebrand et al, 2004; Hjalmarsson et al, 1998).

For an extensive list of references, see the overview (Hou and Wang, 2013).

Data-driven subspace methods are used in metrology for dynamic measurements (speed-up of a sensor) (Markovsky, 2015a,b). The classical approach (Eichstädt et al, 2010) for dynamic measurements is design of a compensator, which requires a model of the sensor dynamics. In some cases however, the sensor dynamic is not know a priori because it depends on environmental parameters that are variable and/or unpredictable. In dynamic weighing, the to-be-measured mass affects the measurement process dynamics, so that again the assumption that it is a priori known is unrealistic. For these situations, adaptive methods are proposed (Shu, 1993). The data-driven methods have the advantage over the adaptive methods of being computationally cheaper and therefore more suitable for real-time implementation on a digital signal processor.

Exercises

6.1 (Missing values compleation with given model). Given a linear time-invariant system $\mathscr{B} \in \mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}$ and a sequence $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{q}})^T$ with missing values $w_{\mathrm{d},\mathscr{I}_{\mathrm{m}}} = \mathtt{NaN}$ and given values $w_{\mathrm{d},\mathscr{I}_{\mathrm{g}}}$, check if $w_{\mathrm{d}} \in \mathscr{B}|_T$, *i.e.*, find a complete sequence $\widehat{w} \in (\mathbb{R}^{\mathtt{q}})^T$ that agrees with the given data $\widehat{w}_{\mathscr{I}_{\mathrm{g}}} = w_{\mathrm{d},\mathscr{I}_{\mathrm{g}}}$ and is a trajectory of the system $\mathscr{B}$, or assert that such a trajectory does not exist.

6.2 (Subspace metod for data-driven step response simulation).
Implement the method developed in Exercise 3.5 and study empirically the estimation accuracy in the present of noise on the given trajectory.

6.3 (Data-driven step response simulation, using the SLRA package).
Use the SLRA package to solve the data-driven simulation problem (DD SIM) in the special case of a step input and zero initial conditions (step response simulation).

References

Campi M, Lecchini A, Savaresi S (2002) Virtual reference feedback tuning: a direct method for the design of feedback controllers. Automatica 38:1337–1346

Chan JT (1996) Data-based synthesis of a multivariable linear-quadratic regulator. Automatica 32:403–407

De Moor B (1994) Total least squares for affinely structured matrices and the noisy realization problem. IEEE Trans Signal Proc 42(11):3104–3113

Eichstädt S, Elster C, Esward T, Hessling J (2010) Deconvolution filters for the analysis of dynamic measurement processes: a tutorial. Metrologia 47:522–533

Favoreel W (1999) Subspace methods for identification and control of linear and bilinear systems. PhD thesis, ESAT, K.U.Leuven

Furuta K, Wongsaisuwan M (1995) Discrete-time LQG dynamic controller design using plant Markov parameters. Automatica 31(9):1317–1324

Gevers M (2004) Identification for control: Achievements and open problems. IFAC Proceedings 37(9):401–412

Heinig G (1995) Generalized inverses of Hankel and Toeplitz mosaic matrices. Linear Algebra Appl 216(0):43–59

Hildebrand R, Lecchini A, Solari G, Gevers M (2004) Prefiltering in iterative feedback tuning: Optimization of the prefilter for accuracy. IEEE Trans Automat Contr 49:1801–1806

Hjalmarsson H, Gevers M, Gunnarsson S, Lequin O (1998) Iterative feedback tuning: theory and applications. IEEE Control Systems Magazine 18:26–41

Hou ZS, Wang Z (2013) From model-based control to data-driven control: Survey, classification and perspective. Information Sciences 235:3–35

Ikeda M, Fujisaki Y, Hayashi N (2001) A model-less algorithm for tracking control based on input-ouput data. Nonlinear Analysis 47:1953–1960

Kalman RE (1979) On partial realizations, transfer functions, and canonical forms. Acta Polytechnica Scandinavica 31:9–32

Kawamura Y (1998) Direct construction of lq regulator based on orthogonalization of signals: Dynamical output feedback. Control Lett 34:1–9

Ljung L (2002) Identification for control: simple process models. In: Proceedings of the 41st IEEE Conference on Decision and Control, 2002, vol 4, pp 4652–4657

Markovsky I (2015a) An application of system identification in metrology. Control Eng Practice 43:85–93

Markovsky I (2015b) Comparison of adaptive and model-free methods for dynamic measurement. IEEE Signal Proc Lett 22(8):1094–1097

Markovsky I (2017) A missing data approach to data-driven filtering and control. IEEE Trans Automat Contr 62:1972–1978

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

Markovsky I, Rapisarda P (2008) Data-driven simulation and control. Int J Control 81(12):1946–1959

Polderman J, Willems JC (1998) Introduction to mathematical systems theory. Springer-Verlag

Safonov M, Cabral F (2001) Fitting controllers to data. Control Lett 43(4):299–308

Safonov M, Tsao T (1997) The unfalsified control concept and learning. IEEE Trans Automat Contr 42(6):843–847

Shi G, Skelton R (2000) Markov data-based LQG control. J of Dynamic Systems, Measurement, and Control 122:551–559

Shu W (1993) Dynamic weighing under nonzero initial conditions. IEEE Trans Instrumentation Measurement 42(4):806–811

Van Heusden K, Karimi A, Bonvin D (2011) Data-driven model reference control with asymptotically guaranteed stability. Int J of Adaptive Control and Signal Proc 25(4):331–351

Woodley B (2001) Model free subspace based H_∞ control. PhD thesis, Stanford University

Ziegler J, Nichols N (1942) Optimum settings for automatic controllers. Trans American Society of Mechanical Engineers 64:759–768

Chapter 7
Nonlinear modeling problems

With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.

J. von Neumann

Applied to nonlinear modeling problem, the maximum likelihood estimation principle leads to nonconvex optimization problems and yields inconsistent estimators in the errors-in-variables setting. This chapter presents a computationally cheap and statistically consistent estimation method based on a bias correction procedure, called *adjusted least squares estimation*. The adjusted least squares method is applied to curve fitting (static modeling) and system identification.

Section 7.1 presents a general nonlinear data modeling framework. The model class consists of affine varieties with bounded complexity (dimension and degree) and the fitting criteria are algebraic and geometric. Section 7.2 shows that the underlying computational problem is *polynomially structured low-rank approximation*. In the algebraic fitting method, the approximating matrix is unstructured and the corresponding problem can be solved globally and efficiently. The geometric fitting method aims to solve the polynomially structured low-rank approximation problem, which is nonconvex and has no analytic solution. The equivalence of nonlinear data modeling and low-rank approximation unifies existing curve fitting methods, showing that algebraic fitting is a relaxation of geometric fitting, obtained by removing the structure constraint. Motivated by the fact that the algebraic fitting method is efficient but statistically inconsistent, Section 7.3.3 proposes a *bias correction procedure*. The resulting adjusted least squares method yields a consistent estimator. Simulation results show that it is effective also for small sample sizes.

Section 7.4 considers the class, called *polynomial time-invariant*, of discrete-time, single-input, single-output, nonlinear dynamical systems described by a polynomial difference equation. The identification problem is: 1) find the monomials appearing in the difference equation representation of the system (structure selection), and 2) estimate the coefficients of the equation (parameter estimation). Since the model representation is linear in the parameters, the parameter estimation by minimization of the 2-norm of the equation error leads to unstructured low-rank approximation. However, knowledge of the model structure is required and even with the correct model structure, the method is statistically inconsistent. For the structure selection, we propose to use 1-norm regularization and for the bias correction, we use the adjusted least squares method.

7.1 A framework for nonlinear data modeling

Identifying a curve in a set of curves that best fits given data points is a common problem in computer vision, statistics, and coordinate metrology. In the applications, the curve that is fitted to the data is a model for the data and, correspondingly, the set of candidate curves is the model class.

Data modeling problems are specified by choosing a model class and a fitting criterion. The fitting criterion is maximisation of a measure for fit between the data and a model. Equivalently, the criterion can be formulated as minimization of a measure for lack of fit (misfit) between the data and a model. Data modeling problems can be classified according to the type of model and the type of fitting criterion as follows:

- linear/affine vs nonlinear model class,
- algebraic vs geometric fitting criterion.

A model is a subset of the data space. The model is linear if it is a subspace. Otherwise, it is nonlinear. A geometric fitting criterion minimises the sum-of-squares of the Euclidean distances from the data points to a model. An algebraic fitting criterion minimises an equation error (residual) in a representation of the model. In general, the algebraic fitting criterion has no simple geometric interpretation. Problems using linear model classes and algebraic criteria are easier to solve numerically than problems using nonlinear model classes and geometric criteria.

Section 7.1.1 considers nonlinear static models: kernels of systems of multivariable polynomials (affine varieties). The complexity of such a model is defined as the pair of the variety's dimension and the degree of its polynomial representation. Section 7.1.2 shows examples of conic section fitting and subspace clustering.

7.1.1 Models defined by solution sets of polynomial equations

Consider first static multivariate models. The to-be-modelled data

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

is a set of N real $\mathtt{q}$-dimensional vectors—the observations, also called data points. A model for the data $\mathscr{D}$ is a subset of the data space $\mathbb{R}^{\mathtt{q}}$ and a model class $\mathscr{M}^{\mathtt{q}}$ for $\mathscr{D}$ is a set of subsets of $\mathbb{R}^{\mathtt{q}}$. For example, the linear model class in $\mathbb{R}^{\mathtt{q}}$ consists of all subspaces of $\mathbb{R}^{\mathtt{q}}$. An example of a nonlinear model class in $\mathbb{R}^2$ is the set of the conic sections. When the dimension $\mathtt{q}$ of the data space is understood from the context, it is skipped from the notation of the model class.

In nonlinear data modeling problems, the model is usually represented by an *explicit function* $y = f(u)$, where $d = \Pi \operatorname{col}(u, y)$, with Π a permutation matrix. The corresponding statistical estimation problem is *regression*. As in the linear case, we call the functional relation $y = f(u)$ among the variables u and y, an input/output representation of the model

$$\mathscr{B} = \{\, \Pi \operatorname{col}(u,y) \mid y = f(u) \,\}. \tag{I/O}$$

Input-output representations are appealing because they display a *causal relation* among the variables: some variables (inputs) cause other variables (outputs). Also from a computational point of view, all outcomes $d \in \mathscr{B}$ are conveniently parameterized by the independent variable y.

The alternative kernel representation

$$\mathscr{B} = \ker(R) := \{\, d \in \mathbb{R}^{\mathtt{q}} \mid R(d) = 0 \,\} \tag{KER}$$

defines the model by a relation, also called an *implicit function* $R(d) = 0$. Clearly, (I/O) is a special case of (KER). In the linear case, we argued that the kernel representation has advantages over the input/output representation. This is even more so, in the nonlinear case. Consider, for example, data fitting by a conic section model. Only parabolas and lines can be represented by functions. Hyperbolas, ellipses, and the vertical line $\{\, (u,y) \mid u = 0 \,\}$ are not graphs of a function $y = f(u)$ and therefore can not be modeled by an input/output representation.

Using the kernel representation however complicates the basic problem of describing the behavior $\mathscr{B}$, corresponding to a given function R. Indeed, this problem is equivalent to solving an equation $R(d) = 0$. Finding all solutions of a nonlinear equation is in general an intractable problem.

The complexity of a linear static model $\mathscr{B}$ is defined as the dimension of $\mathscr{B}$, *i.e.*, the smallest number $\mathtt{m}$, such that there is a linear function $P : \mathbb{R}^{\mathtt{m}} \to \mathbb{R}^{\mathtt{q}}$, for which

$$\mathscr{B} = \operatorname{image}(P) := \{\, P(\ell) \mid \ell \in \mathbb{R}^{\mathtt{m}} \,\}. \tag{IMAGE}$$

Similarly, the dimension of a nonlinear model $\mathscr{B}$ is defined as the smallest natural number $\mathtt{m}$, such that there is a function $P : \mathbb{R}^{\mathtt{m}} \to \mathbb{R}^{\mathtt{q}}$, for which (IMAGE) holds. In the context of nonlinear models, however, the model dimension alone is not sufficient to define the model complexity. For example, in $\mathbb{R}^2$ both a linear model (a line passing through the origin) and an ellipse have dimension equal to one, however, it is intuitively clear that the ellipse is a more "complex" model than the line.

The missing element in the definition of the model complexity in the nonlinear case is the "complexity" of the function P. In what follows, we restrict to models that can be represented as kernels of polynomial functions, *i.e.*, we consider *affine varieties*. The complexity of an affine variety (IMAGE) is defined as the pair $(\mathtt{m},\mathtt{d})$, where $\mathtt{m}$ is the dimension of $\mathscr{B}$ and $\mathtt{d}$ is the degree of R. This definition allows us to distinguish a linear or affine model ($\mathtt{d} = 1$) from a nonlinear model ($\mathtt{d} > 1$) with the same dimension. For a model $\mathscr{B}$ with complexity $(\mathtt{m},\mathtt{d})$, we call $\mathtt{d}$ the degree of $\mathscr{B}$.

The complexity of a model class is the maximal complexity (in a lexicographic ordering of the pairs $(\mathtt{m},\mathtt{d})$) over all models in the class. The model class of complexity bounded by $(\mathtt{m},\mathtt{d})$ is denoted by $\mathscr{P}_{\mathtt{m},\mathtt{d}}$.

7.1.2 Special cases

The model class $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}$ and the related exact and approximate modeling problems (EM) and (AM) are illustrated next on specific examples.

1. *Linear/affine model class of bounded complexity.*
   An affine model $\mathscr{B}$ (*i.e.*, an affine set in $\mathbb{R}^{\mathtt{q}}$) is an affine variety, defined by a first order polynomial through kernel or image representation. The dimension of the affine variety coincides with the dimension of the affine set. Therefore, $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},1}$ is an affine model class in $\mathbb{R}^{\mathtt{q}}$ with complexity bounded by $\mathtt{m}$. The linear model class in $\mathbb{R}^{\mathtt{q}}$, with dimension bounded by $\mathtt{m}$, is a subset $\mathscr{L}^{\mathtt{q}}_{\mathtt{m},0}$ of $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},1}$.
2. *Geometric fitting by a linear model.*
   Approximate data modeling using the linear model class $\mathscr{L}^{\mathtt{q}}_{\mathtt{m}}$ and the geometric fitting criterion (dist) is a low-rank approximation problem

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{\mathscr{D}} \quad \left\| \Phi(\mathscr{D}) - \Phi(\widehat{\mathscr{D}}) \right\|_{\mathrm{F}} \\ \text{subject to} & \operatorname{rank}\left(\Phi(\widehat{\mathscr{D}})\right) \leq \mathtt{m}, \end{array} \tag{LRA}$$

   where

$$\Phi(\mathscr{D}) := \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}.$$

   The rank constraint in (LRA) is equivalent to the constraint that the data $\widehat{\mathscr{D}}$ is exact for a linear model of dimension bounded by $\mathtt{m}$. This justifies the statement that exact modeling is an ingredient of approximate modeling.
3. *Algebraic curves.*
   In the special case of a curve in the plane, we use the notation

$$x := \text{first component } d_{1\cdot} \text{ of } d \quad \text{and} \quad y := \text{second component } d_{2\cdot} \text{ of } d.$$

   Note that $w = \operatorname{col}(x,y)$ is not necessarily an input/output partitioning of the variables. An affine variety of dimension one is called an algebraic curve. A second order algebraic curve

$$\mathscr{B}(A,b,c) = \{\, d \mid d^\top A d + b^\top d + c = 0 \,\}, \tag{$\mathscr{B}(A,b,c)$}$$

   where $A = A^\top$, b, and c are parameters, is a *conic section*.
4. *Subspace clustering.*
   In the subspace clustering problem, the data $\mathscr{D}$ is fitted by a model $\mathscr{B} \subset \mathbb{R}^{\mathtt{q}}$ that is the union of n-subspaces $\mathscr{B}_1, \ldots, \mathscr{B}_n$ with bounded dimensions

$$\dim(\mathscr{B}_1) \leq r_1, \ldots, \dim(\mathscr{B}_n) \leq r_n.$$

   The union of subspaces model admits a representation

$$\mathscr{B}(R^1, \ldots, R^n) = \{\, d \in \mathbb{R}^{\mathtt{q}} \mid (R^1 d) \cdots (R^n d) = 0 \,\},$$

where $R^1 \in \mathbb{R}^{(\mathtt{q}-r_1)\times \mathtt{q}}$, ..., $R^n \in \mathbb{R}^{(\mathtt{q}-r_n)\times \mathtt{q}}$ are parameters of the model. At least one of the R^i is assumed to be nonzero in order to avoid the trivial model $\mathscr{B}(0,\ldots,0) = \mathbb{R}^{\mathtt{q}}$. Note that in the case $\mathtt{q} = 2$ and $n = 2$, with $r_1 = r_2 = 1$, the union of two lines model $\mathscr{B}(R^1, R^2)$ is a special conic section $\mathscr{B}(A, b, c)$, with

$$A = (R^1)^\top R^2 + (R^2)^\top R^1, \quad b = 0, \quad \text{and} \quad c = 0.$$

Fitting a set of points $\mathscr{D}$ in $\mathbb{R}^{\mathtt{q}}$ by a union of lines model $\mathscr{B}(R^1,\ldots,R^n)$ is a type of a clustering problem. Indeed, the data $\mathscr{D}$ is clustered into the r subspaces:

$$\mathscr{B}_i = \mathscr{B}(R^i) = \{\, d \in \mathbb{R}^{\mathtt{q}} \mid R^i d = 0 \,\} \quad \text{for } i = 1,\ldots,n.$$

Next, we consider a simplified version of the subspace clustering problem when $\mathtt{q} = 2$ and $r = 2$ and the data is fitted exactly. The problem is: Given a data set $\mathscr{D}$, find $\widehat{\mathscr{B}} = \mathscr{B}(R^1, R^2)$, such that $\mathscr{D}$ is fitted exactly by $\widehat{\mathscr{B}}$. The data points $d_i \in \mathbb{R}^2$, $i = 1,\ldots,N$ lie on a union of two lines if and only if there are vectors R^1 and R^2, at least one of which is nonzero, such that

$$(R^1 d_i)(R^2 d_i) = 0, \quad \text{for } i = 1,\ldots,N.$$

This condition can be written in a matrix form as

$$\underbrace{\begin{bmatrix} R^1_1 R^2_1 & R^1_1 R^2_2 + R^1_2 R^2_1 & R^1_2 R^2_2 \end{bmatrix}}_{\theta} \underbrace{\begin{bmatrix} x_1^2 & \cdots & x_N^2 \\ x_1 y_1 & \cdots & x_N y_N \\ y_1^2 & \cdots & y_N^2 \end{bmatrix}}_{\Phi(\mathscr{D})} = 0.$$

(R^i_j is the jth element of the vector R^i.) We showed that if $\mathscr{D} \subset \widehat{\mathscr{B}}$ holds,

$$\operatorname{rank}(\Phi(\mathscr{D})) \leq 2.$$

In subspace clustering, the rank constraint is only a *necessary* condition for exact data fitting. In addition, $\theta \in \operatorname{left\,ker}\big(\Phi(\mathscr{D})\big)$ should have the following structure

$$\begin{aligned} \theta_1 &= 1 \\ \theta_2 &= \alpha + \beta \\ \theta_3 &= \alpha\beta, \end{aligned}$$

for some α and β. This is a polynomial factorization condition that makes possible to map the estimated parameter θ to the the model parameters R^1, R^2 by solving the equations:

$$\begin{aligned} \theta_1 &= R^1_1 R^2_1 \\ \theta_2 &= R^1_1 R^2_2 + R^1_2 R^2_1 \\ \theta_3 &= R^1_2 R^2_2. \end{aligned} \qquad \text{(FACTORIZE)}$$

Applied to the data $d_1 = \left[\begin{smallmatrix} -1 \\ -1 \end{smallmatrix}\right]$, $d_2 = \left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right]$, $d_3 = \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$, $d_4 = \left[\begin{smallmatrix} -1 \\ 1 \end{smallmatrix}\right]$ the kernel computation of the matrix $\Phi(\mathscr{D})$, followed by the solution of (FACTORIZE) yields the exact fit shown in Figure 7.1. Note that the obtained model $\mathscr{B}(R^1, R^2)$ is a particular conic section fitting exactly the data.

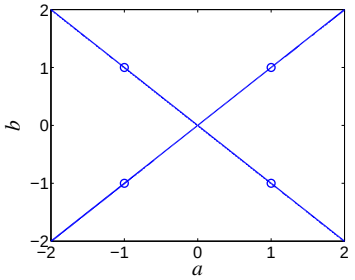


Fig. 7.1: Subspace clustering example: fitting data (circles) by a union of two lines.

7.2 Nonlinear low-rank approximation

This section considers geometric and algebraic fitting problems with the static polynomial model class $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}$ of bounded complexity. Section 7.2.1 introduces a parametrization of the model, defined by a kernel representations. The model structure is the set of monomials that appear in the representations. Due to the combinatorial increase of the number of monomials as a function of the number of variables and the degree, the structure detection problem (finding which monomials have zero coefficients) is a key problem in nonlinear data modeling. Section 7.2.2 shows that the geometric fitting problem with the model class $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}$ is equivalent to polynomially structured low-rank approximation and the geometric fitting problem is a relaxation of the geometric fitting problem when the structure is ignored.

7.2.1 Parametrization of the kernel representations

Consider a kernel representation (KER) of an affine variety $\mathscr{B} \in \mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}$, parametrized by a $\mathtt{p} \times 1$ multivariable polynomial R. The number of monomials in $\mathtt{q}$ variables with degree $\mathtt{d}$ or less is

$$\mathtt{q}_{\text{ext}} := \binom{\mathtt{q}+\mathtt{d}}{\mathtt{d}} = \frac{(\mathtt{q}+\mathtt{d})!}{\mathtt{d}!\mathtt{q}!}. \qquad (\mathtt{q}_{\text{ext}})$$

Define the vector of all such monomials

$$\phi(d) := \begin{bmatrix} \phi_1(d) & \cdots & \phi_{\mathtt{q}_{\text{ext}}}(d) \end{bmatrix}^\top.$$

The polynomial R can be written as

$$R_\Theta(d) = \sum_{k=1}^{\mathtt{q}_{\text{ext}}} \Theta_k \phi_k(d) = \Theta\phi(d), \qquad (R_\Theta)$$

where Θ is an $\mathtt{p} \times \mathtt{q}_{\text{ext}}$ parameter matrix.

In what follows, we assume that the monomials are ordered in $\phi(d)$ in decreasing degree according to the lexicographic ordering (with alphabet the indexes of d). For example, with $\mathtt{q} = 2$, $\mathtt{d} = 2$, and $d = \operatorname{col}(x, y)$,

$$\begin{aligned} \mathtt{q}_{\text{ext}} = 6 \quad \text{and} \quad \phi^\top(x,y) &= \begin{bmatrix} \phi_1 & \phi_2 & \phi_3 & \phi_4 & \phi_5 & \phi_6 \end{bmatrix} \\ &= \begin{bmatrix} x^2 & xy & x & y^2 & y & 1 \end{bmatrix} \end{aligned}$$

In general,

$$\phi_k(d) = d_{1\cdot}^{\mathtt{d}_{k1}} \cdots d_{\mathtt{q}\cdot}^{\mathtt{d}_{k\mathtt{q}}}, \quad \text{for } k = 1, \ldots, \mathtt{q}_{\text{ext}}, \qquad (\phi_k)$$

where

- $d_{1\cdot}, \ldots, d_{\mathtt{q}\cdot} \in \mathbb{R}$ are the elements of $d \in \mathbb{R}^{\mathtt{q}}$, and
- $\mathtt{d}_{ki} \in \mathbb{Z}_+$, is the degree of the ith element of d in the kth monomial ϕ_k.

The matrix formed from the degrees $\mathtt{d}_{ki}$

$$\mathtt{D} = \begin{bmatrix} \mathtt{d}_{ki} \end{bmatrix} \in \mathbb{R}^{\mathtt{q}_{\text{ext}} \times \mathtt{q}}$$

uniquely defines the vector of monomials ϕ. The degrees matrix D depends only on the number of variables q and the degree d. For example, with $\mathtt{q} = 2$ and $\mathtt{d} = 2$,

$$\mathtt{D}^\top = \begin{bmatrix} 2 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 1 & 0 \end{bmatrix}.$$

The function `monomials` generates an implicit function `phi` that evaluates the 2-variate vector of monomials ϕ, with degree d.

179a ⟨*Monomials constructor* 179a⟩≡ 179b▷

```
function [Deg, phi] = monomials(deg)
```

Defines:
`monomials`, used in chunks 184b and 187a.

First an extended degrees matrix $\mathtt{D}_{\text{ext}} \in \{0, 1, \ldots, \mathtt{d}\}^{(\mathtt{d}+1)^2 \times 2}$, corresponding to all monomials $x^{\mathtt{d}_x} y^{\mathtt{d}_y}$ with degrees at most d, is generated. It can be verified that

$$\mathtt{D}_{\text{ext}} = \begin{bmatrix} \mathbf{r}_\mathtt{d} \otimes \mathbf{1}_{\mathtt{d}+1} & \mathbf{1}_{\mathtt{d}+1} \otimes \mathbf{r}_\mathtt{d} \end{bmatrix}, \quad \text{where } \mathbf{r}_\mathtt{d} := \begin{bmatrix} 0 & 1 & \cdots & \mathtt{d} \end{bmatrix}^\top$$

is such a matrix; moreover, the monomials are ordered in decreasing degree.

179b ⟨*Monomials constructor* 179a⟩+≡ ◁179a 180▷

```
Deg_ext = [kron([0:deg]', ones(deg + 1, 1)), ...
           kron(ones(deg + 1, 1), [0:deg]')];
```


Then the rows of $\mathtt{D}_{\text{ext}}$ are scanned and those with degree less than or equal to d are selected to form a matrix D.

180 ⟨*Monomials constructor* 179a⟩+≡ ◁179b

```
str = []; Deg = []; q = 2;
for i = 1:size(Deg_ext, 1)
    if (sum(Deg_ext(i, :)) <= deg)
        for k = q:-1:1,
            str = sprintf('.* d(%d,:) .^ %d %s', ...
                          k, Deg_ext(i, k), str);
        end
        str = sprintf('; %s', str(4:end));
        Deg = [Deg_ext(i, :); Deg];
    end
end
eval(sprintf('phi = @(d) [%s];', str(2:end)))
```

Minimality of the kernel representation is equivalent to the condition that the parameter Θ is full row rank. The nonuniqueness of R_Θ corresponds to a nonuniqueness of Θ. The parameters Θ and $Q\Theta$, where Q is a nonsingular matrix, define the same model. Therefore, without loss of generality, we can assume that the representation is minimal and normalise it, so that

$$\Theta\Theta^\top = I_\mathtt{p}.$$

Note that a $\mathtt{p} \times \mathtt{q}_{\text{ext}}$ full row rank matrix Θ defines via (R_Θ) a polynomial matrix R_Θ, which defines a minimal kernel representation (KER) of a model $\mathscr{B}_\Theta$ in $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}$. Therefore, Θ defines a function

$$\mathscr{B}_\Theta : \mathbb{R}^{\mathtt{p} \times \mathtt{q}_{\text{ext}}} \to \mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}.$$

Vice verse, a model $\mathscr{B}$ in $\mathscr{P}^{\mathtt{q}}_{\mathtt{m},\mathtt{d}}$ corresponds to a (nonunique) $\mathtt{p} \times \mathtt{q}_{\text{ext}}$ full row rank matrix Θ, such that $\mathscr{B} = \mathscr{B}_\Theta$. For a given q, there are one-to-one mappings

$$\mathtt{q}_{\text{ext}} \leftrightarrow \mathtt{d} \quad \text{and} \quad \mathtt{p} \leftrightarrow \mathtt{m},$$

defined by ($\mathtt{q}_{\text{ext}}$) and $\mathtt{p} = \mathtt{q} - \mathtt{m}$, respectively.

7.2.2 *Curve fitting* $\iff$ *polynomial low rank approximation*

We show that the approximate modeling problems (AM) and (EM) for the model class of affine varieties with bounded complexity $\mathscr{P}_{\mathtt{m},\mathtt{d}}$ are equivalent to low-rank approximation problems.

Proposition 7.1 (Algebraic fit $\iff$ unstructured low-rank approximation)
Algebraic fitting with the model class $\mathscr{P}_{\mathtt{m},\mathtt{d}}$

$$\begin{aligned} &\text{minimize} \quad \textit{over } \Theta \in \mathbb{R}^{\mathtt{p}\times \mathtt{q}_{\mathrm{ext}}} \quad \sqrt{\textstyle\sum_{j=1}^{N} \left\| R_\Theta(d_j) \right\|_{\mathrm{F}}^2} \\ &\textit{subject to} \quad \Theta\Theta^\top = I_{\mathtt{p}} \end{aligned} \tag{AM$'_\Theta$}$$

is equivalent to the unstructured low-rank approximation

$$\begin{aligned} &\text{minimize} \quad \textit{over } \widehat{\Phi} \in \mathbb{R}^{\mathtt{q}\times \mathtt{p}} \quad \left\| \Phi_{\mathtt{d}}(\mathscr{D}) - \widehat{\Phi} \right\|_{\mathrm{F}} \\ &\textit{subject to} \quad \operatorname{rank}(\widehat{\Phi}) \leq \mathtt{q}_{\mathrm{ext}} - \mathtt{p}. \end{aligned} \tag{LRA}$$

Proof. Using the polynomial representation (R_Θ), the squared cost function of (AM$'_\Theta$) can be rewritten as a quadratic form

$$\begin{aligned} \textstyle\sum_{j=1}^{N} \left\| R_\Theta(d_j) \right\|_{\mathrm{F}}^2 &= \left\| \Theta \Phi_{\mathtt{d}}(\mathscr{D}) \right\|_{\mathrm{F}}^2 \\ &= \operatorname{trace}\left(\Theta \Phi_{\mathtt{d}}(\mathscr{D}) \Phi_{\mathtt{d}}^\top(\mathscr{D}) \Theta^\top \right) = \operatorname{trace}\left(\Theta \Psi_{\mathtt{d}}(\mathscr{D}) \Theta^\top \right). \end{aligned}$$

Therefore, the algebraic fitting problem is equivalent to an eigenvalue problem for $\Psi_{\mathtt{d}}(\mathscr{D})$ or, equivalently, to low-rank approximation problem for $\Phi_{\mathtt{d}}(\mathscr{D})$. □

Proposition 7.2 (Geometric fit $\iff$ polynomial low rank approximation) *Geometric fitting with the model class* $\mathscr{P}_{\mathtt{m},\mathtt{d}}$

$$\text{minimize} \quad \textit{over } \mathscr{B} \in \mathscr{P}_{\mathtt{m},\mathtt{d}} \quad \operatorname{dist}(\mathscr{D}, \mathscr{B}) \tag{AM}$$

is equivalent to polynomially structured low-rank approximation

$$\begin{aligned} &\text{minimize} \quad \textit{over } \widehat{D} \in \mathbb{R}^{\mathtt{q}\times N} \quad \left\| D - \widehat{D} \right\|_{\mathrm{F}} \\ &\textit{subject to} \quad \operatorname{rank}\left(\Phi_{\mathtt{d}}(\widehat{D}) \right) \leq \mathtt{q}_{\mathrm{ext}} - \mathtt{p}. \end{aligned} \tag{PSLRA}$$

Proof. Problem (AM) is equivalent to

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{\mathscr{D}} \text{ and } \mathscr{B} \quad \sqrt{\textstyle\sum_{j=1}^{N} \left\| d_j - \widehat{d}_j \right\|_2^2} \\ &\text{subject to} \quad \widehat{\mathscr{D}} \subset \mathscr{B} \in \mathscr{P}_{\mathtt{m},\mathtt{d}}. \end{aligned} \tag{$*$}$$

Using the condition

$$\widehat{\mathscr{D}} \subset \mathscr{B} \in \mathscr{P}_{\mathtt{m},\mathtt{d}} \quad \Longrightarrow \quad \operatorname{rank}\left(\Phi_{\mathtt{d}}(\widehat{\mathscr{D}}) \right) \leq \mathtt{q}_{\mathrm{ext}} - \mathtt{p} \tag{MPUM}$$

to replace the constraint of ($*$) with a rank constraint for the structured matrix $\Phi_{\mathtt{d}}(\widehat{\mathscr{D}}) = \Phi_{\mathtt{d}}(\widehat{D})$, this latter problem becomes a polynomially structured low-rank approximation problem (PSLRA). □

Propositions 7.1 and 7.2 show a relation between the algebraic and geometric fitting problems.

Corollary 7.3. *Algebraic fitting (AM$'_\Theta$) is a relaxation of geometric fitting (AM), obtained by removing the structure constraint of the approximating matrix* $\Phi_{\mathtt{d}}(\widehat{D})$.

7.3 Computational algorithms

In the linear case, the misfit computation problem is a linear least norm problem. This fact is effectively used in the variable projection method. In the nonlinear case, the misfit computation problem is a nonconvex optimization problem. Thus the elimination step of the variable projection approach is not possible in the nonlinear case. This requires the approximation $\widehat{\mathscr{D}} = \{\widehat{d}_1, \ldots, \widehat{d}_N\}$ to be treated as an extra optimization variable together with the model parameter Θ. As a result, the computational complexity and sensitivity to local minima increases in the nonlinear case. Therefore, the choice of the initial approximation is critical. The default initial approximation is obtained from a direct method such as the algebraic fitting method. Next, we present a modification of the algebraic fitting method that is motivated by the objective of obtaining an unbiased estimate in the errors-in-variables setup.

7.3.1 Bias corrected low-rank approximation

Assume that the data $\mathscr{D}$ is generated according to the errors-in-variables model

$$\begin{aligned} d_j = \bar{d}_j + \tilde{d}_j, \quad \text{where } \bar{d}_j \in \bar{\mathscr{B}} \in \mathscr{P}_{\mathtt{m},\mathtt{q}} \\ \text{and} \quad \operatorname{vec}\left(\begin{bmatrix} \tilde{d}_1 & \cdots & \tilde{d}_N \end{bmatrix} \right) \sim \mathrm{N}(0, s^2 I_{\mathtt{q}N}). \end{aligned} \tag{EIV}$$

Here $\bar{\mathscr{B}}$ is the to-be-estimated true model. The estimate $\widehat{\mathscr{B}}$ obtained by the algebraic fitting method (AM$'_\Theta$) is biased, *i.e.*, $\mathbf{E}(\widehat{\mathscr{B}}) \neq \bar{\mathscr{B}}$. In this section, we derive a bias correction procedure. The correction depends on the noise variance s^2, however, the noise variance can be estimated from the data $\mathscr{D}$ together with the model parameter $\widehat{\Theta}$. The resulting bias corrected estimate $\widehat{\mathscr{B}}_{\mathrm{c}}$ is invariant to rigid transformations. Simulation results show that $\widehat{\mathscr{B}}_{\mathrm{c}}$ has smaller orthogonal distance to the data than alternative direct methods.

Define the matrices

$$\Psi := \Phi_{\mathtt{d}}(\mathscr{D}) \Phi_{\mathtt{d}}^\top(\mathscr{D}) \quad \text{and} \quad \bar{\Psi} := \Phi_{\mathtt{d}}(\bar{\mathscr{D}}) \Phi_{\mathtt{d}}^\top(\bar{\mathscr{D}})$$

The algebraic fitting method computes the estimate $\widehat{\Theta}$ from the eigenvalue decomposition of Ψ. We construct a "corrected" matrix Ψ_{c}, such that

$$\mathbf{E}(\Psi_{\mathrm{c}}) = \bar{\Psi}. \tag{$*$}$$

This property ensures that the corrected estimate $\widehat{\Theta}_{\mathrm{c}}$, obtained from the eigenvectors related to the $\mathtt{p}$ smallest eigenvalues of Ψ_{c}, is unbiased.

182 ⟨*Bias corrected low-rank approximation* 182⟩≡

```
function [th, sh] = bclra(D, deg)
[q, N] = size(D); qext = nchoosek(q + deg, deg);
```

⟨*construct the corrected matrix* Ψ_c 184b⟩
⟨*estimate* s^2 *and* θ 185b⟩

Defines:
bclra, used in chunk 187a.

The key tool to achieve bias correction is the sequence of the Hermite polynomials, defined by the recursion

$$h_0(x) = 1, \quad h_1(x) = x, \quad \text{and}$$
$$h_k(x) = x h_{k-1}(x) - (k-2) h_{k-2}(x), \quad \text{for } k = 2, 3, \ldots$$

(See Table 7.1 for explicit expressions of $h_2, \ldots, h_{10}$.) The Hermite polynomials

Table 7.1: Explicit expressions of the Hermite polynomials $h_2, \ldots, h_{10}$.

$$\begin{aligned}
h_2(x) &= x^2 - 1 \\
h_3(x) &= x^3 - 3x \\
h_4(x) &= x^4 - 6x^2 + 3 \\
h_5(x) &= x^5 - 10x^3 + 15x \\
h_6(x) &= x^6 - 15x^4 + 45x^2 - 15 \\
h_7(x) &= x^7 - 21x^5 + 105x^3 - 105x \\
h_8(x) &= x^8 - 28x^6 + 210x^4 - 420x^2 + 105 \\
h_9(x) &= x^9 - 36x^7 + 378x^5 - 1260x^3 + 945x \\
h_{10}(x) &= x^{10} - 45x^8 + 630x^6 - 3150x^4 + 4725x^2 - 945
\end{aligned}$$

have the deconvolution property

$$\mathbf{E}\left(h_k(\bar{x} + \widetilde{x})\right) = \bar{x}^k, \quad \text{where } \widetilde{x} \sim \mathrm{N}(0,1). \tag{$**$}$$

The following code generates a cell array h of implicit function that evaluate the sequence of Hermite polynomials: h{k+1}(d) = $h_k(d)$. (The difference in the indexes of the h and h is due to MATLAB convention indexes to be positive integers.)

183 ⟨*define the Hermite polynomials* 183⟩≡ (184b)
```
h{1} = @(x) 1; h{2} = @(x) x;
for k = 3:(2 * deg + 1)
  h{k} = @(x) [x * h{k - 1}(x) zeros(1, mod(k - 2, 2))] ...
                - [0 (k - 2) * h{k - 2}(x)];
end
```

We have,

$$\Psi = \sum_{\ell=1}^{N} \phi(d_\ell)\phi^\top(d_\ell) = \sum_{\ell=1}^{N} \left[\phi_i(d_\ell)\phi_j(d_\ell)\right],$$

and, from (ϕ_k), the (i,j)th element of Ψ is

$$\psi_{ij} = \sum_{\ell=1}^{N} d_{1\ell}^{\mathtt{d}_{i1}+\mathtt{d}_{j1}} \cdots d_{\mathtt{q}\ell}^{\mathtt{d}_{i\mathtt{q}}+\mathtt{d}_{j\mathtt{q}}} = \sum_{\ell=1}^{N} \prod_{k=1}^{\mathtt{q}} (d_{0,k\ell} + \widetilde{d}_{k\ell})^{\mathtt{d}_{i\mathtt{q}}+\mathtt{d}_{j\mathtt{q}}}.$$

By assumption, $\widetilde{d}_{k\ell}$ are independent, zero mean, normally distributed (see (EIV)). Then, using the deconvolution property ($**$) of the Hermit polynomials,

$$\psi_{\mathrm{c},ij} := \sum_{\ell=1}^{N} \prod_{k=1}^{\mathtt{q}} h_{\mathtt{d}_{ik}+\mathtt{d}_{jk}}(d_{k\ell}) \tag{ψ_{ij}}$$

has the unbiasness property ($*$), *i.e.*,

$$\mathbf{E}(\psi_{\mathrm{c},ij}) = \sum_{\ell=1}^{N} \prod_{k=1}^{\mathtt{q}} d_{0,k\ell}^{\mathtt{d}_{ik}+\mathtt{d}_{jk}} =: \psi_{0,ij}.$$

The elements $\psi_{\mathrm{c},ij}$ of the corrected matrix are even polynomials of s of degree less than or equal to

$$\mathtt{d}_\psi = \left\lceil \frac{\mathtt{q}\mathtt{d}+1}{2} \right\rceil.$$

The following code constructs a $1 \times (\mathtt{d}_\psi + 1)$ vector of the coefficients of $\psi_{\mathrm{c},ij}$ as a polynomial of s^2. Note that the product of Hermite polynomials in (ψ_{ij}) is a convolution of their coefficients.

184a ⟨*construct* $\psi_{c,ij}$ 184a⟩≡ (184b)
```
Deg_ij = Deg(i, :) + Deg(j, :);
for l = 1:N
  psi_ijl = 1;
  for k = 1:q
    psi_ijl  = conv(psi_ijl, h{Deg_ij(k) + 1}(D(k, l)));
  end
  psi_ijl = [psi_ijl zeros(1, dpsi - length(psi_ijl))];
  psi(i, j, :) = psi(i, j, :) + ...
                 reshape(psi_ijl(1:dpsi), 1, 1, dpsi);
end
```

The corrected matrix

$$\Psi_{\mathrm{c}}(s^2) = \Psi_{\mathrm{c},0} + s^2\Psi_{\mathrm{c},1} + \cdots + s^{2\mathtt{d}_\psi}\Psi_{\mathrm{c},\mathtt{d}_\psi}$$

is then obtained by computing its elements in the lower triangular part

184b ⟨*construct the corrected matrix* Ψ_c 184b⟩≡ (182) 185a▷
```
⟨define the Hermite polynomials 183⟩
Deg  = monomials(deg);
dpsi = ceil((q * deg + 1) / 2);
psi  = zeros(qext, qext, dpsi);
for i = 1:qext
  for j = 1:qext
    if i >= j
      ⟨construct ψ_{c,ij} 184a⟩
    end
  end
end
```
Uses monomials 179a.

and using the symmetry property to fill in the upper triangular part

185a ⟨*construct the corrected matrix* Ψ_c 184b⟩+≡ (182) ◁184b
```
for k = 1:dpsi,
  psi(:, :, k) = psi(:, :, k) + triu(psi(:, :, k)', 1);
end
```

The rows of the parameter $\widehat{\Theta}$ form a basis for the p-dimensional (approximate) null space of $\Psi_c(s^2)$

$$\Theta\Psi_c(s^2) = 0.$$

Computing simultaneously s and Θ is a *polynomial eigenvalue problem:* the noise variance estimate is the minimum eigenvalue and Θ is a corresponding eigenvector.

185b ⟨*estimate* s^2 *and* θ 185b⟩≡ (182)
```
[evec, ev] = polyeig_(psi); ev(find(ev < 0)) = inf;
[sh2, min_ind]  = min(ev);
sh = sqrt(sh2); th = evec(:, min_ind);
```

(The function `polyeig_` is a minor modification of the standard MATLAB function `polyeig`. The input to `polyeig_` is a 3-dimensional tensor while the input to `polyeig` is a sequence of matrices—the slices of the tensor in the third dimension.)

7.3.2 Method based on local optimization

The nonlinearly structured low-rank approximation problem (PSLRA) is solved numerically using Optimization Toolbox.

185c ⟨*Polynomially structured low-rank approximation* 185c⟩≡ 185d▷
```
function [th, Dh, info] = pslra(D, phi, r, xini)
[q, N] = size(D); nt = size(phi(D), 1);
```
Defines:
`pslra`, used in chunk 187a.

If not specified, the initial approximation is taken as the algebraic fit and the data.

185d ⟨*Polynomially structured low-rank approximation* 185c⟩+≡ ◁185c 185e▷
```
if (nargin < 4) | isempty(xini)
    tini = lra(phi(D), r); xini = [D(:); tini(:)];
end
```
Uses `lra` 105a.

Anonymous functions that extract the data approximation $\widehat{\mathscr{D}}$ and the model parameter θ from the optimization parameter `x` are defined next.

185e ⟨*Polynomially structured low-rank approximation* 185c⟩+≡ ◁185d 185f▷
```
Dh = @(x) reshape(x(1:(q * N)), q, N);
th = @(x) reshape(x((q * N + 1):end), nt - r, nt)';
```

The optimization problem is set and solved, using the Optimization Toolbox:

185f ⟨*Polynomially structured low-rank approximation* 185c⟩+≡ ◁185e
```
⟨set optimization solver and options 115d⟩
prob.objective = @(x) norm(D - Dh(x), 'fro');
prob.nonlcon   = @(x) deal([], ...
          [th(x)' * phi(Dh(x)), th(x)' * th(x) - eye(nt - r)]);
prob.x0 = xini;
⟨call optimization solver 115e⟩ Dh = Dh(x); th = th(x);
```


7.3.3 Numerical examples

In this section, we apply the algebraic and geometric fitting methods on algebraic curve fitting examples. In all examples, except for the last one, the data $\mathscr{D}$ is simulated in the errors-in-variables setup, see (EIV) on page 182. The perturbations $\tilde{d}_j$, $j = 1,\ldots,N$ are independent, zero mean, normally distributed 2×1 vectors with covariance matrix $s^2 I_2$. The true model $\bar{\mathscr{B}} = \ker(\bar{r})$, the number of data points N, and the perturbation standard deviation s are simulation parameters. The true model is plotted by a solid line, the data points by circles, the algebraic fit by a dotted line, the bias corrected fit by dashed dotted line, and the geometric fit by a dashed line.

Test function

The test script `test_curve_fitting` assumes that the simulation parameters:

- polynomial r in x and y, defined as a symbolic object;
- degree `d` of r;
- number of data points N;
- noise standard deviation s; and
- coordinates `ax` of a rectangle for plotting the results

are already defined.

186a ⟨*Test curve fitting* 186a⟩≡
⟨*initialize the random number generator* 25⟩
⟨*default parameters* 186b⟩
⟨*generate data* 186c⟩
⟨*fit data* 187a⟩
⟨*plot results* 187b⟩
Defines:
`test_curve_fitting`, used in chunks 188–90.

If not specified, $\mathtt{q} = 2$, $\mathtt{m} = 1$.

186b ⟨*default parameters* 186b⟩≡ (186a)
```
if ~exist('q'), q = 2; end
if ~exist('m'), m = 1; end
if ~exist('xini'), xini = []; end
```

The true (`D0`) and noisy (`D`) data points are generated by plotting the true model

186c ⟨*generate data* 186c⟩≡ (186a) 186d▷
```
figure,
H = plot_model(r, ax, 'LineStyle', '-', 'color', 'k');
```
Uses `plot_model` 187c.

and sampling N equidistant points on the curve

186d ⟨*generate data* 186c⟩+≡ (186a) ◁186c
```
D0 = [];
for h = H',
  D0 = [D0 [get(h, 'XData'); get(h, 'YData')]];
end
```

```
D0 = D0(:, round(linspace(1, size(D0, 2), N)));
D  = D0 + s * randn(size(D0));
```

The data is fitted by the algebraic (`lra`), bias corrected (`bclra`), and geometric (`pslra`) fitting methods:

187a ⟨*fit data* 187a⟩≡ (186a 190b)

```
qext = nchoosek(q + deg, deg); p = q - m;
[Deg, phi] = monomials(deg);
th_exc   = lra(phi(D), qext - p)';
th_ini   = bclra(D, deg);
[th, Dh] = pslra(D, phi, qext - p, xini);
```

Uses `bclra` 182, `lra` 105a, `monomials` 179a, and `pslra` 185c.

The noisy data and the fitted models are plotted on top of the true model:

187b ⟨*plot results* 187b⟩≡ (186a 190b)

```
hold on; plot(D(1,:), D(2,:), 'o', 'markersize', 7);
plot_model(th2poly(th_exc, phi), ax, ...
                   'LineStyle', ':', 'color', 'k');
plot_model(th2poly(th_ini, phi), ax, ...
                   'LineStyle', '-.', 'color', 'r');
plot_model(th2poly(th, phi), ax, ...
                   'LineStyle', '-', 'color', 'b');
axis(ax); print_fig(sprintf('%s-est', name))
```

Uses `plot_model` 187c and `th2poly` 187d.

Plotting the algebraic curve

$$\mathscr{B} = \{ d \mid \phi(d)\theta = 0 \}$$

in a region, defined by the vector `rect`, is done with the function `plot_model`.

187c ⟨*Plot the model* 187c⟩≡

```
function H = plot_model(r, rect, varargin)
H = ezplot(r, rect);
if nargin > 2, for h = H', set(h, varargin{:}); end, end
```

Defines:
`plot_model`, used in chunks 186c, 187b, and 253.

The function `th2poly` converts a vector of polynomial coefficients to a function that evaluates that polynomial.

187d ⟨$\Theta \mapsto R_\Theta$ 187d⟩≡

```
function r = th2poly(th, phi), r = @(x, y) th' * phi([x y]');
```

Defines:
`th2poly`, used in chunk 187b.

Fitting algebraic curves in $\mathbb{R}^2$

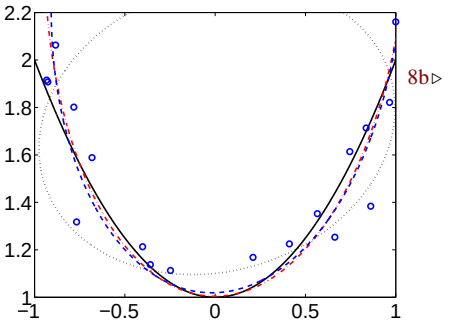
Simulation 1: Parabola

$\mathscr{B} = \{ \mathrm{col}(x,y) \mid y = x^2 + 1 \}$

188a ⟨*Curve fitting examples* 188a⟩≡

```
clear all
name = 'parabola';
N = 20; s = 0.1;
deg = 2; syms x y;
r = x^2 - y + 1;
ax = [-1 1 1 2.2];
test_curve_fitting
```

Uses `test_curve_fitting` 186a.



8b▷

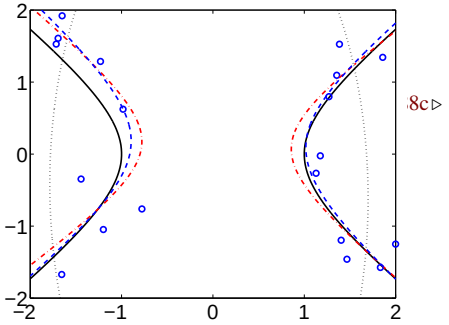
Simulation 2: Hyperbola

$\mathscr{B} = \{ \mathrm{col}(x,y) \mid x^2 - y^2 - 1 = 0 \}$

188b ⟨*Curve fitting examples* 188a⟩+≡

```
name = 'hyperbola';
N = 20; s = 0.3;
deg = 2; syms x y;
r = x^2 - y^2 - 1;
ax = [-2 2 -2 2];
test_curve_fitting
```

Uses `test_curve_fitting` 186a.



8c▷

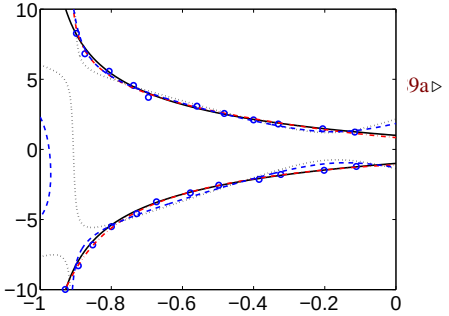
Simulation 3: Cissoid

$\mathscr{B} = \{ \mathrm{col}(x,y) \mid y^2(1+x) = (1-x)^3 \}$

188c ⟨*Curve fitting examples* 188a⟩+≡

```
name = 'cissoid';
N = 25; s = 0.02;
deg = 3; syms x y;
r = y^2 * (1 + x) ...
    - (1 - x)^3;
ax = [-1 0 -10 10];
test_curve_fitting
```

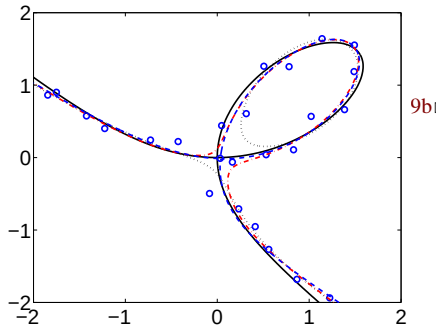
Uses `test_curve_fitting` 186a.



9a▷

Simulation 4: Folium of Descartes
$\mathscr{B} = \{\,\mathrm{col}(x,y) \mid x^3+y^3-3xy=0\,\}$

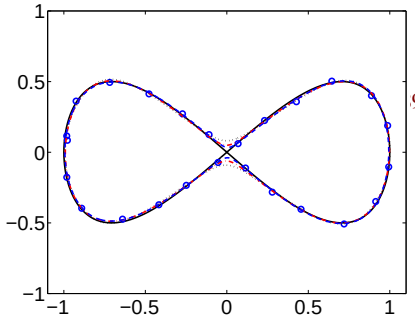
189a ⟨*Curve fitting examples* 188a⟩+≡
```
name = 'folium';
N = 25; s = 0.1;
deg = 3; syms x y;
r = x^3 + y^3 - 3 * x * y;
ax = [-2 2 -2 2];
test_curve_fitting
```
Uses `test_curve_fitting` 186a.



9b▷

Simulation 5: Eight curve
$\mathscr{B} = \{\,\mathrm{col}(x,y) \mid y^2-x^2+x^4=0\,\}$

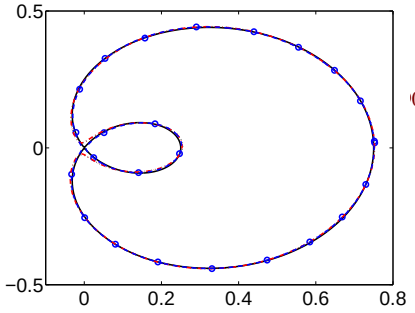
189b ⟨*Curve fitting examples* 188a⟩+≡
```
name = 'eight';
N = 25; s = 0.01;
deg = 4; syms x y;
r = y^2 - x^2 + x^4;
ax = [-1.1 1.1 -1 1];
test_curve_fitting
```
Uses `test_curve_fitting` 186a.



9c▷

Simulation 6: Limacon of Pascal
$\mathscr{B} = \{\,\mathrm{col}(x,y) \mid y^2+x^2-(4x^2-2x+4y^2)^2=0\,\}$

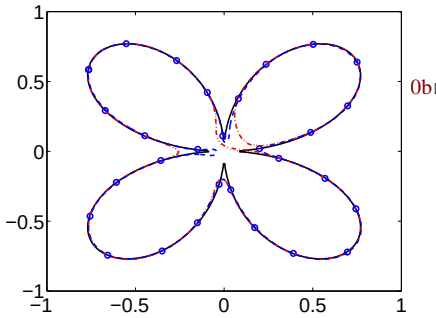
189c ⟨*Curve fitting examples* 188a⟩+≡
```
name = 'limacon';
N = 25; s = 0.002;
deg = 4; syms x y;
r = y^2 + x^2 - (4 * x^2 ...
    - 2 * x + 4 * y^2)^2;
ax = [-.1 .8 -0.5 .5];
test_curve_fitting
```
Uses `test_curve_fitting` 186a.



0a▷

Simulation 7: Four-leaved rose
$\mathscr{B} = \{\,(x,y) \mid (x^2+y^2)^3-4x^2y^2=0\,\}$

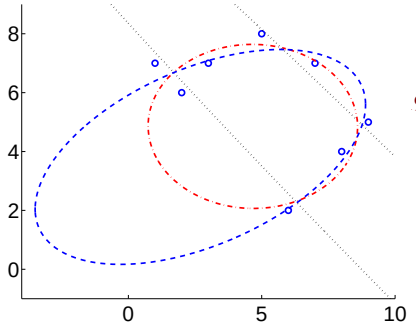
190a ⟨*Curve fitting examples* 188a⟩+≡
```
name = 'rose';
N = 30; s = 0.002;
deg = 6; syms x y;
r = (x^2 + y^2)^3 ...
    - 4 * x^2 * y^2;
ax = [-1 1 -1 1];
test_curve_fitting
```
Uses `test_curve_fitting` 186a.



0b▷

Simulation 8: "Special data" example from (Gander et al, 1994)

190b ⟨*Curve fitting examples* 188a⟩+≡
```
name = 'special-data';
D = [1 2 5 7 9 3 6 8 ;
     7 6 8 7 5 7 2 4 ];
D0 = D; deg = 2;
xini = [D(:)' 1 0 0 1 0 -1]'
figure, ax = [-4 10 -1 9];
⟨fit data 187a⟩ ⟨plot results 187b⟩
```



90a

7.4 Identification of polynomial time-invariant systems

In this section, we considers a nonlinear identification problem. Section 7.4.1 defines the model class, called polynomial time-invariant. The identification problem with known model structure is considered in Section 7.4.2 and the structure selection problem (*i.e.*, selecting the monomials that appear in the difference equation representation of the model) is considered in Section 7.4.3. We address the structure selection problem by adding a sparsity inducing 1-norm regularizer in the cost function. The 1-norm regularizer imposes the prior knowledge of a small number of monomials, which is often physically meaningful. The number of monomials in the model structure is also a measure for the complexity of the model, so that the regularized cost function reflects the accuracy vs complexity trade-off. Once the model structure is selected, the model parameters are re-estimated using the adjusted least squares method. Section 7.4.4 shows a simulation example of the overall method.

7.4.1 Polynomial time-invariant model class

A polynomial time-invariant model is defined by a higher order difference equation

$$R\big(w, \sigma w, \ldots, \sigma^\ell w\big) = 0, \tag{KER}$$

where R is a multivariable polynomial. We consider the special case of (KER) with two manifest variables ($\mathtt{q} = 2$), input/output partitioning $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$, and R of the form

$$R(w, \sigma w, \ldots, \sigma^\ell w) = f(x) - \sigma^\ell y, \tag{R}$$

where

$$x := \operatorname{col}(w, \sigma w, \ldots, \sigma^{\ell-1} w, \sigma^\ell u).$$

Therefore, the model class considered is defined by the difference equation

$$\sigma^\ell y = f(x). \tag{I/O}$$

The assumption (KER) about R allows us to compute the response of the model to a given input and initial condition by recursive evaluation of $y(t)$, for $t = 1, 2, \ldots$

Note that f is a n_x-variate polynomial, where

$$n_x := \dim(x) = 2\ell + 1.$$

It is linearly parameterized by a parameter vector θ, with $\dim(\theta) =: n_\theta$:

$$\begin{aligned} f(x) &= \theta_1 \underbrace{x_1^{n_{11}} \cdots x_{n_x}^{n_{1n_x}}}_{\phi_1(x)} + \cdots + \theta_{n_\theta} \underbrace{x_1^{n_{n_\theta 1}} \cdots x_{n_x}^{n_{n_\theta n_x}}}_{\phi_{n_\theta}(x)} \\ &= \begin{bmatrix} \theta_1 & \cdots & \theta_{n_\theta} \end{bmatrix} \begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_{n_\theta}(x) \end{bmatrix} = \theta^\top \phi(x), \end{aligned}$$

The vector of monomials ϕ defines the *model structure*.

For the input/output model (I/O), the parameterization of the function R is

$$R(w, \sigma w, \ldots, \sigma^\ell w) = \begin{bmatrix} \theta^\top & -1 \end{bmatrix} \begin{bmatrix} \phi(x) \\ \sigma^\ell y \end{bmatrix} = \theta_{\text{ext}}^\top \phi_{\text{ext}}(x_{\text{ext}}) = R(x_{\text{ext}}),$$

where $x_{\text{ext}} := \left[\begin{smallmatrix} x \\ \sigma^\ell y \end{smallmatrix}\right] = \operatorname{col}(w, \sigma w, \ldots, \sigma^\ell w)$.

A particular model with representation (I/O) is specified by the model structured ϕ and the model parameter vector θ. With a given model structure, the model

$$\mathscr{B}_\phi(\theta) := \{\, w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right] \mid \text{(I/O) holds} \,\}, \tag{$\mathscr{B}_\phi(\theta)$}$$

depends on the parameter vector θ only.

The model class defined by ($\mathscr{B}_\phi(\theta)$) is denoted by $\mathscr{P}_\phi$. In what follows we will consider model structures consisting of all monomials with bounds on the lag ℓ and the degree $\mathtt{n}_{\max}$. The corresponding model class is denoted by $\mathscr{P}_{\ell,\mathtt{n}_{\max}}$. The number of parameters $n_\theta = \binom{2\ell+1}{n_{\max}}$ is combinatorial in ℓ and $\mathtt{n}_{\max}$.

Example 7.4 (First order quadratic time-invariant model). By definition, a first order ($\ell = 1$) quadratic ($\mathtt{n}_{\max} = 2$) time-invariant model has a representation

$$R(w, \sigma w) = \sum_{i+j+k+l=2} R_{ijkl} u^i y^j (\sigma u)^k (\sigma y)^l.$$

The polynomial R can be written as $R = \theta^\top \phi(x)$ with a vector of model parameters

$$\theta^\top = \begin{bmatrix} R_{2000} & R_{1100} & R_{1010} & R_{1001} & R_{0200} & R_{0110} & R_{0101} & R_{0020} & R_{0011} & R_{0002} \end{bmatrix}$$

and model structure

$$\phi(x) = \begin{bmatrix} u^2 & uy & u\sigma u & u\sigma y & y^2 & y\sigma u & y\sigma y & (\sigma u)^2 & \sigma u \sigma y & (\sigma y)^2 \end{bmatrix}^\top.$$

Therefore, in this simplest possible nontrivial example of a polynomial time-invariant system, the number of parameters is $n_\theta = 10$.

7.4.2 Identification with known structure

The identification problem considered is defined as follows: Given a time series $w_{\text{d}} \in (\mathbb{R}^2)^T$ and model structure ϕ,

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\theta} \text{ and } \widehat{w} \quad \|w - \widehat{w}\| \\ &\text{subject to} && \widehat{w} \in \mathscr{B}_\phi(\widehat{\theta}). \end{aligned} \tag{NLSYSID}$$

It yields the maximum-likelihood estimator for the true parameter vector $\bar{\theta}$ in the errors-in-variables setting. As in the static case, the dynamic modeling problem is equivalent to the polynomially structured low-rank approximation problem

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{w} \quad \|w - \widehat{w}\| \\ &\text{subject to} && \operatorname{rank}\big(\Phi(\widehat{w})\big) \leq n_\theta - 1, \end{aligned} \tag{NLSLRA}$$

where

$$\Phi(\widehat{w}) := \begin{bmatrix} \phi_{\text{ext}}\big(\widehat{x}_{\text{ext}}(\ell+1)\big) & \cdots & \phi_{\text{ext}}\big(\widehat{x}_{\text{ext}}(T)\big) \end{bmatrix}. \tag{$\Phi(\widehat{w})$}$$

Contrary to affine structured low-rank approximation problems, (NLSLRA) does not allow the approximation matrix $\widehat{D}$ to be eliminated analytically, as done in the variable projection method (see Section 4.2). Therefore, the number of optimization variables is of the order of magnitude of the number of data points. This makes the

use of local optimization methods infeasible for medium to large scale polynomially structured low-rank approximation problems.

With exact data, the extended true parameter vector $\bar{\theta}_{\text{ext}}^{\top} := \begin{bmatrix} \bar{\theta}^{\top} & -1 \end{bmatrix}$ is in the left kernel of the matrix $\Phi(w)$. Moreover, provided that the left kernel of $\Phi(w)$ is one dimensional, the true system's parameter vector $\bar{\theta}$ can be computed from a nonzero vector $\widehat{\theta}_{\text{ext}}$ in $\operatorname{left\,ker}\big(\Phi(w)\big)$, by scaling. The condition that $\Phi(w)$ has one dimensional left kernel, *i.e.*, $\operatorname{rank}\big(\Phi(w)\big) = n_\theta - 1$ is a nonlinear equivalent to the persistency of excitation assumption in linear system identification, *cf.*, Lemma 3.7.

With noisy data, a heuristic identification method is obtained by compute an approximate left kernel of $\Phi(w)$ by minimization of the residual error

$$\begin{array}{ll} \text{minimize} & \text{over } \theta_{\text{ext}} \quad \|\theta_{\text{ext}}^{\top}\Phi(w_{\text{d}})\|_2 \\ \text{subject to} & \|\theta_{\text{ext}}\|_2 = 1. \end{array} \tag{LRA}$$

(LRA) is equivalent to unstructured low-rank approximation of the matrix $\Phi(w)$.

The minimization of the residual error (LRA) is a relaxation of the maximum likelihood estimation method (NLSLRA). As in the static case, we use the bias correction procedure, presented in Section 7.3.1, in order to improve the performance of the basic low-rank approximation method (LRA).

7.4.3 Structure selection

We propose a method for structure selection, based on sparse approximation within a larger model class $\mathscr{P}_{\ell,\mathtt{n}_{\max}}$, *i.e.*, we consider all monomials ϕ with a bounded total degree. Assuming that the true model structure $\bar{\phi}$ is included in ϕ, *i.e.*, the monomials in $\bar{\phi}$ are a subset of the monomials in ϕ, the parameter vector of a less complex model, represented within a larger model class is a sparse vector. Only the coefficients corresponding to the monomials that are part of the model's representation are nonzero.

Sparsity can be enforced by an ℓ_1-norm regularizer:

$$\text{minimize} \quad \text{over } \theta \quad \underbrace{\left\| \begin{bmatrix} \theta^{\top} & -1 \end{bmatrix} \Phi(w) \right\|_2}_{\text{fitting error}} + \underbrace{\gamma \|\theta\|_1}_{\text{regularizer}}. \tag{LS-SVM}$$

Problem (LS-SVM) is a least squares support vector machine problem (Suykens and Vandewalle, 1999). It is convex and can be solved globally and efficiently.

The regularization parameter γ is selected, so that the computed parameter vector $\widehat{\theta}$ has a specified sparsity, *i.e.*, the number of nonzero elements in $\widehat{\theta}$ is equal to a specified number n_x.

7.4.4 Identification experiments

In order to validate the method for polynomial time-invariant system identification, presented in Section 7.4.2 and the bias correction method of Section 7.3.1, we perform Monte Carlo experiments with data simulated in the errors-in-variables setting. The true model

$$\bar{\mathscr{B}} = \{ w = \begin{bmatrix} u \\ y \end{bmatrix} \mid \sigma^2 y + a_1 \sigma y + a_0 y = c\alpha y^3 + b_0 u + b_1 \sigma u + b_2 \sigma^2 \}$$

is polynomial time-invariant with

$$f(x) = \theta_1 u(t) + \theta_2 u(t+1) + \theta_3 u(t+2) + \theta_4 y(t) + \theta_5 y^3(t) + \theta_6 y(t+1).$$

It belongs to the class $\mathscr{P}_{2,3}$, *i.e.*, $\ell = 2$ and $\mathtt{n}_{\max} = 3$. The degrees matrix $\mathtt{N}$ is

	$u(t)$	$y(t)$	$u(t+1)$	$y(t+1)$	$u(t+2)$
ϕ_1	1	0	0	0	0
ϕ_2	0	1	0	0	0
ϕ_3	0	0	1	0	0
ϕ_4	0	0	0	1	0
ϕ_5	0	0	0	0	1
ϕ_6	0	3	0	0	0

and the true parameter vector is

$$\bar{\theta} = \begin{bmatrix} -0.5 & 0.25 & -1 & -0.25 & 0.3 & 0.1 \end{bmatrix}^{\top}.$$

Figure 7.2 shows the relative estimation error $e := \|\bar{\theta} - \widehat{\theta}\| / \|\bar{\theta}\|$, averaged over 100 Monte Carlo experiments. The result shows that both (LRA) and the bias correction methods recover the exact model from noise free data. The bias correction, however, reduces the estimation error of the basic low-rank approximation method.

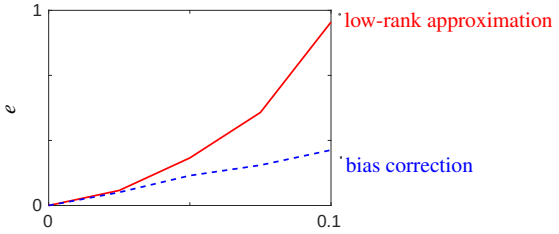


Fig. 7.2: Both unstructured low-rank approximation (solid red line) and bias corrected low-rank approximation (dashed blue line) recover the exact model from exact data however the bias correction reduces the error when the data is noisy.

7.5 Notes and references

Curve fitting

Fitting curves to data is a basic problem in coordinate metrology, see (Van Huffel, 1997, Part IV). In the computer vision literature, there is a large body of work on ellipsoid fitting (see, *e.g.*, (Bookstein, 1979; Fitzgibbon et al, 1999; Gander et al, 1994; Kanatani, 1994; Markovsky et al, 2004)), which is a special case of the data fitting problem in the chapter when the degree of the polynomial is equal to two. The subspace clustering problem (fitting the model that is a union of subspaces to the data) is a generalized principal component analysis problem (Vidal et al, 2005).

The problem of passing from image to kernel representation of the model is known in computer algebra as the *implicialization problem* (Cox et al, 2004, page 96). The inverse—passing from a kernel to an image representation of the model—is a problem of solving a system of multivariable polynomial equations.

Nonlinearly structured low-rank approximation

Relaxation of the nonlinearly structured low-rank approximation problem, based on ignoring the nonlinear structure and thus solving the problem as unstructured low-rank approximation, (*i.e.*, the algebraic fitting method) is known in the machine learning literature as *kernel principal component analysis* (Schölkopf et al, 1999; Vidal et al, 2005). The *principal curves*, introduced in (Hastie and Stuetzle, 1989), lead to a problem of minimizing the sum of squares of the distances from data points to a curve. This is a polynomially structured low-rank approximation problem. More generally, dimensionality reduction by manifold learning, see, *e.g.*, (Zhang and Zha, 2005) is related to the problem of fitting an affine variety to data, which is also polynomially structured low-rank approximation.

Nonlinear system identification

Major classes of nonlinear models, listed in decreasing level of generality, are:

- Volterra series (Boyd et al, 1984),
- Nonlinear state space (Paduart et al, 2010),
- Nonlinear Auto Regressive Exogenous (NARX) (Billings, 2013),
- Block-oriented (Giri and Bai, 2010).

The Volterra series represent the output of the system as a sum of convolutions of what are called the system kernels' with the input. This representation is an universal approximation of a nonlinear system's dynamics as the number of terms in the sum grows (Boyd et al, 1984). The challenge in system identification with this class of systems is the explosion of the system parameters (the kernel functions). A possible solution is to use regularization and therefore prior knowledge about the kernels.

NARX model represents the system by a nonlinear difference equation. The difference equation can be specified by basis expansion, neural network, *etc*. Using expansion in a basis and truncating the infinite series to a finite one, turns the model identification problem into a parameter estimation problem—find the expansion coefficients from the data. As with the Volterra series, the problem of estimating the parameters is ill-posed, so that prior knowledge about the model is needed.

The block-oriented models represent the system as a connection of linear time-invariant dynamic and nonlinear static subsystems. Depending on the topology of the connection network and the types of blocks being used, there are different types of block-oriented models (Hammerstein, Wiener, Wiener-Hammerstein, ...). The problem of structure selection for a block-oriented model is to discover from data the network topology and the type of models in the nodes. Even with given structure, however, the problem of estimating the model parameters may be challenging.

Exercises

7.1 (Exact line fitting). Given a set of points $\mathscr{D} = \{d_1, \ldots, d_N\} \subset \mathbb{R}^2$,

1. find conditions for existence of a line that passes through the points,
2. explain how to test the condition numerically,
3. implement a method for exact line fitting.

7.2 (Approximate line fitting). Given a set of points $\mathscr{D} = \{d_1, \ldots, d_N\} \subset \mathbb{R}^2$, explain how to find a line $\mathscr{B}$ in $\mathbb{R}^2$ that is as close as possible to the data $\mathscr{D}$ in the sense of minimizing the geometric distance.

7.3 (Exact conic section fitting). Given a set of points $\mathscr{D} = \{d_1, \ldots, d_N\} \subset \mathbb{R}^2$,

1. find conditions for existence of an exact conic section $\mathscr{B}(A,b,c)$ fit,
2. propose a numerical method for exact conic section fitting,
3. implement the method and test it on the data $\mathscr{D} = \left\{ \left[\begin{smallmatrix} -1 \\ -1 \end{smallmatrix}\right], \left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right], \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right], \left[\begin{smallmatrix} -1 \\ 1 \end{smallmatrix}\right] \right\}$.

7.4 (Approximate conic section fitting).
Implement the algebraic fitting method in the special case of a conic section model.

References

Billings S (2013) Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains. John Wiley & Sons

Bookstein FL (1979) Fitting conic sections to scattered data. Computer Graphics Image Proc 9:59–71

Boyd S, Chua L, Desoer C (1984) Analytical foundations of Volterra series. IMA Journal of Mathematical Control and Information 1(3):243–282

Cox D, Little J, O'Shea D (2004) Ideals, varieties, and algorithms. Springer

Fitzgibbon A, Pilu M, Fisher R (1999) Direct least-squares fitting of ellipses. IEEE Trans Pattern Anal Machine Intelligence 21(5):476–480

Gander W, Golub G, Strebel R (1994) Fitting of circles and ellipses: Least squares solution. BIT 34:558–578

Giri F, Bai EW (2010) Block-oriented nonlinear system identification, vol 1. Springer

Hastie T, Stuetzle W (1989) Principal curves. J American Statistical Association 84:502–516

Kanatani K (1994) Statistical bias of conic fitting and renormalization. IEEE Trans Pattern Anal Machine Intelligence 16(3):320–326

Markovsky I, Kukush A, Van Huffel S (2004) Consistent least squares fitting of ellipsoids. Numerische Mathematik 98(1):177–194

Paduart J, Lauwers L, Swevers J, Smolders K, Schoukens J, Pintelon R (2010) Identification of nonlinear systems using polynomial nonlinear state space models. Automatica 46(4):647–656

Schölkopf B, Smola A, Müller K (1999) Kernel principal component analysis., MIT Press, Cambridge, MA, pp 327–352

Shklyar S, Kukush A, Markovsky I, Van Huffel S (2007) On the conic section fitting problem. Journal of Multivariate Analysis 98:588–624

Suykens J, Vandewalle J (1999) Least squares support vector machine classifiers. Neural processing letters 9(3):293–300

Usevich K, Markovsky I (2016) Adjusted least squares fitting of algebraic hypersurfaces. Linear Algebra Appl 502:243–274

Van Huffel S (ed) (1997) Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modeling. SIAM, Philadelphia

Vidal R, Ma Y, Sastry S (2005) Generalized principal component analysis (GPCA). IEEE Trans Pattern Analysis and Machine Intelligence 27(12):1945–1959

Zhang Z, Zha H (2005) Principal manifolds and nonlinear dimension reduction via local tangent space alignment. SIAM J on Scientific Computing 26:313–338

Chapter 8
Dealing with prior knowledge

It is reasonable to expect that prior knowledge concerning system parameters and structure may enhance rates of convergence or improve other properties such as robustness.

Goodwin et al (1979)

Centering by mean subtraction is a common preprocessing step applied before other data modeling methods are used. Section 8.1 studies different ways of combining linear data modeling by low-rank approximation with data centering. It is shown that in the basic case of approximation in the Frobenius norm and no structure, the two-step procedure of preprocessing the data by mean subtraction and low-rank approximation is optimal. In the case of approximation in either a weighted norm or with a structure constraint, the two-step procedure is suboptimal. We show modifications of the variable projection and alternating projections methods for weighted and structured low-rank approximation with centering.

Section 8.2 considers an approximate rank revealing factorization problem with structure constraints on the normalized factors. Examples of structure, motivated by an application of the problem in microarray data analysis, are sparsity, nonnegativity, periodicity, and smoothness. An alternating projections algorithm is developed. Although the algorithm is motivated by a specific application in microarray data analysis, the approach is applicable to other types of structure.

Section 8.3 considers the problem of solving approximately in the least squares sense an overdetermined system of linear equations with complex valued coefficients, where the elements of the solution vector are constrained to have the same phase. This problem is reduced to a generalized low-rank matrix approximation.

Section 8.4 considers a blind identification problem with prior knowledge about the input in the form of a linear time-invariant autonomous system. A special case of the problem for constant input has an application in metrology for dynamic measurement. The general problem can be viewed alternatively as a gray-box identification problem or an input estimation problem with unknown model dynamics.

8.1 Data preprocessing

A system's behavior can be linearized around a certain operating point, or offset. Usually, these offsets are unknown, and there are two ways to deal with their presence: first, subtracting estimates of these offsets from the input-output data, second, incorporating the offset into the model as an unknown parameter.

Verhaegen et al (2007)

Closely related to the linear model is the affine one. The observations

$$\mathscr{D} = \{ d_1, \ldots, d_N \}$$

satisfy an *affine static model* $\mathscr{B}$ if $\mathscr{D} \subset \mathscr{B}$, where $\mathscr{B}$ is an affine set, *i.e.*, $\mathscr{B} = c + \mathscr{B}'$, with $\mathscr{B}' \subset \mathbb{R}^{\mathtt{q}}$ a linear model and $c \in \mathbb{R}^{\mathtt{q}}$ an offset vector. Obviously, the affine model class contains as a special case the linear model class. The parameter c, however, allows us to account for a constant offset in the data. Consider, for example, the data

$$\mathscr{D} = \{ \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right], \left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right] \},$$

which satisfies the affine model

$$\mathscr{B} = \left[\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right] + \{ d \mid \left[\begin{smallmatrix} 1 & 0 \end{smallmatrix}\right] d = 0 \}$$

but is not fitted by a linear model of dimension one.

Subtracting the offset c from the data vector d, reduces the affine modeling problem with known offset parameter to an equivalent linear modeling problem. In a realistic data modeling setup, however, the offset parameter is unknown and has to be identified together with the linear model $\mathscr{B}'$. A common heuristic for solving this problem is to replace the offset c by the mean

$$\mathbf{M}(D) := \frac{1}{N} D \mathbf{1}_N = \frac{1}{N}(d_1 + \cdots + d_N) \in \mathbb{R}^{\mathtt{q}},$$

where

$$D = \begin{bmatrix} d_1 & \cdots d_N \end{bmatrix} \quad \text{and} \quad \mathbf{1}_N := \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^\top.$$

This leads to the following two-step procedure for identification of affine models:

1. *pre-processing:* subtract the mean from the data points,
2. *linear identification:* identify a linear model for the pre-processed data.

When the aim is to derive an optimal in some specified sense approximate affine model, the two-step procedure may lead to suboptimal results. Indeed, even if the data centering and linear identification steps are individually optimal with respect to the desired optimality criterion, their composition need not be optimal for the affine modeling problem, *i.e.*, simultaneous subspace fitting and centering. It turns out, however, that in the case of the basic low-rank approximation problem (unstructured approximation in the Frobenius norm) the two-step procedure is optimal. In

the cases of weighted and Hankel structured low-rank approximation problems, the two-step procedure is suboptimal. Methods based on the alternating projections and variable projection algorithms are developed in these cases.

8.1.1 Matrix centering

The matrix centering operation $\mathbf{C}$ subtracts the mean $\mathbf{M}(D)$ from the columns

$$\mathbf{C}(D) := D - \mathbf{M}(D)\mathbf{1}_N^\top = D(I - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^\top).$$

The following proposition justifies the name "matrix centering" for $\mathbf{C}(\cdot)$.

Proposition 8.1 (Matrix centering) *The matrix $\mathbf{C}(D)$ is column centered,* i.e.,

$$\mathbf{M}\big(\mathbf{C}(D)\big) = 0.$$

As shown next, mean computation can be viewed as an optimal modeling problem.

Proposition 8.2 (Mean computation as an optimal modeling) *The mean $\mathbf{M}(D)$ is a solution to the following optimization problem:*

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{D} \text{ and } c \quad \|D - \widehat{D}\|_{\mathrm{F}} \\ \text{subject to} & \widehat{D} = c\mathbf{1}_N^\top. \end{array}$$

Note 8.3 (Intercept). Data fitting with an intercept is a special case of centering when only the output is centered. Intercept requires an input/output partition of the variables and assumes that the input has no offset.

8.1.2 Unweighted low-rank approximation with centering

In this section, we consider the unstructured low-rank approximation problem in the Frobenius norm with centering

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{D} \text{ and } c \quad \|D - c\mathbf{1}_N^\top - \widehat{D}\|_{\mathrm{F}} \\ \text{subject to} & \operatorname{rank}(\widehat{D}) \leq \mathtt{m}. \end{array} \tag{LRA$_c$}$$

The following theorem shows that the two-step procedure yields a solution to (LRA$_c$).

Theorem 8.4 (Optimality of the two-step procedure). *A solution to (LRA$_c$) is the mean of D, $c^* = \mathbf{M}(D)$, and an optimal in a Frobenius norm rank-$\mathtt{m}$ approximation $\widehat{D}^*$ of the centered data matrix $\mathbf{C}(D)$.*

Corollary 8.5 (Nonuniqueness of the solution of (LRA$_c$)). *Let*

$$\widehat{D} = PL, \quad \text{where } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \text{ and } L \in \mathbb{R}^{\mathtt{m}\times N}$$

be a rank revealing factorization of an optimal in a Frobenius norm rank-$\mathtt{m}$ approximation of the centered data matrix $\mathbf{C}(D)$. The solutions of (LRA$_c$) are of the form

$$\begin{array}{rcl} c^*(z) & = & \mathbf{M}(D) + Pz \\ \widehat{D}^*(z) & = & P(L - z\mathbf{1}_N^\top) \end{array} \quad \text{for } z \in \mathbb{R}^{\mathtt{m}}.$$

The same nonuniqueness appears in weighted and structured low-rank approximation problems with centering. It may cause problems in the optimization algorithms. Also solutions produced by different methods can not be compared directly.

8.1.3 Weighted low-rank approximation with centering

Consider the weighted low-rank approximation problem with centering

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{D} \text{ and } c \quad \|D - \widehat{D} - c\mathbf{1}^\top\|_W \\ \text{subject to} & \operatorname{rank}(\widehat{D}) \leq \mathtt{m}, \end{array} \tag{WLRA$_c$}$$

where W is a symmetric positive definite matrix and $\|\cdot\|_W$ is the weighted norm, defined in ($\|\cdot\|_W$). The two-step procedure of computing the mean in a preprocessing step and then the weighted low-rank approximation of the centered data matrix, in general, yields a suboptimal solution to (WLRA$_c$). We present two algorithms for finding a locally optimal solution to (WLRA$_c$). The first one is an alternating projections type method and the second one is a variable projection type method. First, however, we present a special case of (WLRA$_c$) with analytic solution that is more general than the case $W = \alpha I$, with $\alpha \neq 0$ solved in Theorem 8.4.

Theorem 8.6 (Two-sided weighted low-rank approximation with centering). *A solution to (WLRA$_c$), in the case $W = W_{\mathrm{r}} \otimes W_{\mathrm{l}}$, where $W_{\mathrm{l}} \in \mathbb{R}^{\mathtt{q}\times\mathtt{q}}$ and $W_{\mathrm{r}} \in \mathbb{R}^{N\times N}$ with $W_{\mathrm{r}}\mathbf{1}_N = \lambda\mathbf{1}_N$, for some λ, is*

$$c^* = \sqrt{W_{\mathrm{l}}^{-1}}c_{\mathrm{m}}^*/\sqrt{\lambda}, \quad \widehat{D}^* = \sqrt{W_{\mathrm{l}}^{-1}}\widehat{D}_{\mathrm{m}}^*\sqrt{W_{\mathrm{r}}^{-1}},$$

where $(c_{\mathrm{m}}^*, \widehat{D}_{\mathrm{m}}^*)$ is a solution to the unweighted low-rank approximation problem

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{D}_{\mathrm{m}} \text{ and } c_{\mathrm{m}} \quad \|D_{\mathrm{m}} - c_{\mathrm{m}}\mathbf{1}_N^\top - \widehat{D}_{\mathrm{m}}\|_{\mathrm{F}} \\ \text{subject to} & \operatorname{rank}(\widehat{D}_{\mathrm{m}}) \leq \mathtt{m}. \end{array}$$

for the modified data matrix $D_{\mathrm{m}} := \sqrt{W_{\mathrm{l}}}D\sqrt{W_{\mathrm{r}}}$.

Alternating projections algorithm

Following the derivation of the alternating projections method in Section 4.1.4, we use the image representation of the rank constraint

$$\operatorname{rank}(\widehat{D}) \leq \mathtt{m} \quad \Longleftrightarrow \quad \widehat{D} = PL, \quad \text{where } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \text{ and } L \in \mathbb{R}^{\mathtt{m}\times N},$$

in order to obtain an equivalent bilinear problem

$$\text{minimize} \quad \text{over } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}},\ L \in \mathbb{R}^{\mathtt{m}\times N},\ c \in \mathbb{R}^{\mathtt{q}} \quad \|D - PL - c\mathbf{1}_N^\top\|_W \quad (\text{WLRA}_{c,P})$$

to problem (WLRA$_c$). We have

$$\begin{aligned}\|D - c\mathbf{1}_N^\top - PL\|_W &= \left\|\operatorname{vec}(D) - \begin{bmatrix} I_N \otimes P & \mathbf{1}_N \otimes I_{\mathtt{q}}\end{bmatrix}\begin{bmatrix}\operatorname{vec}(L)\\ c\end{bmatrix}\right\|_W \\ &= \left\|\operatorname{vec}(D) - \begin{bmatrix} L^\top \otimes I_{\mathtt{q}} & \mathbf{1}_N \otimes I_{\mathtt{q}}\end{bmatrix}\begin{bmatrix}\operatorname{vec}(P)\\ c\end{bmatrix}\right\|_W,\end{aligned}$$

so that problem (WLRA$_{c,P}$) is linear in c and L as well as in c and P. This suggests a method for weighted low-rank approximation with centering that solves the following two linear least squares problems iteratively,

$$\text{minimize} \quad \text{over } c \text{ and } L \quad \left\|\operatorname{vec}(D) - \begin{bmatrix} I_N \otimes P & \mathbf{1}_N \otimes I_{\mathtt{q}}\end{bmatrix}\begin{bmatrix}\operatorname{vec}(L)\\ c\end{bmatrix}\right\|_W$$

and

$$\text{minimize} \quad \text{over } c \text{ and } P \quad \left\|\operatorname{vec}(D) - \begin{bmatrix} L^\top \otimes I_{\mathtt{q}} & \mathbf{1}_N \otimes I_{\mathtt{q}}\end{bmatrix}\begin{bmatrix}\operatorname{vec}(P)\\ c\end{bmatrix}\right\|_W.$$

The method is summarized in Algorithm 9. The initial approximation $c^{(0)}$, $P^{(0)}$, $L^{(0)}$ is computed by data centering and unweighted low-rank approximation.

Since on each iteration the cost function value is guaranteed to be non increasing and the cost function is bounded from below, the sequence of cost function values generated by the algorithm converges, see Figure 8.1. Moreover, it can be shown that the sequence of the parameters $c^{(k)}$, $P^{(k)}$, $L^{(k)}$ converges to a locally optimal solution to (WLRA$_{c,P}$).

Example 8.7 (Suboptimality of the two-stage procedure for weighted low-rank approximation with centering). In a randomly generated rank-1 weighted approximation problem with $\mathtt{q} = 3$ variables and $N = 6$ data points, the mean of the data matrix and the approximation of the mean, produced by the Algorithm 9 are, respectively

$$c^{(0)} = \begin{bmatrix}0.5017\\ 0.7068\\ 0.3659\end{bmatrix} \quad \text{and} \quad \widehat{c} = \begin{bmatrix}0.4365\\ 0.6738\\ 0.2964\end{bmatrix}.$$

The weighted rank-1 approximation of the matrix $D - c^{(0)}\mathbf{1}_N^\top$ has approximation error 0.1484, while the weighted rank-1 approximation of the matrix $D - \widehat{c}\mathbf{1}_N^\top$ has approximation error 0.1477. This proves the suboptimality of the two-step procedure—data centering, followed by weighted low-rank approximation.

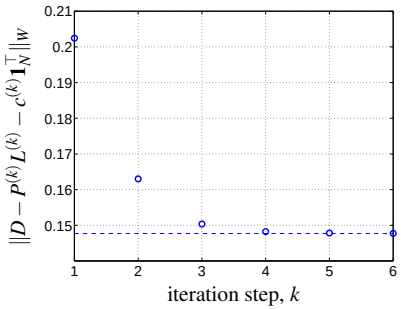


Fig. 8.1: The sequence of the cost function values, produced by Algorithm 9, converges monotonically.

Algorithm 9 Alternating projections algorithm for weighted low-rank approximation with centering.

Input: data matrix $D \in \mathbb{R}^{\mathtt{q}\times N}$, rank constraint $\mathtt{m}$, positive definite weight matrix $W \in \mathbb{R}^{N\mathtt{q}\times N\mathtt{q}}$, and relative convergence tolerance ε.

1: Initial approximation: compute the mean $c^{(0)} := \mathbf{M}(D)$ and the rank-$\mathtt{m}$ approximation $\widehat{D}^{(0)}$ of the centered matrix $D - c^{(0)}\mathbf{1}_N^\top$. Let $\widehat{D}^{(0)} = P^{(0)}L^{(0)}$, with $P^{(0)} \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}}$ and $L^{(0)} \in \mathbb{R}^{\mathtt{m}\times N}$ full rank.
2: $k := 0$.
3: **repeat**
4: Let $\mathbf{P} := \begin{bmatrix} I_N \otimes P^{(k)} & \mathbf{1}_N \otimes I_{\mathtt{q}}\end{bmatrix}$ and

$$\begin{bmatrix}\operatorname{vec}(L^{(k+1)})\\ \widehat{c}^{(k+1)}\end{bmatrix} := \left(\mathbf{P}^\top W \mathbf{P}\right)^{-1}\mathbf{P}^\top W \operatorname{vec}(D).$$

5: Let $\mathbf{L} := \begin{bmatrix} L^{(k+1)\top} \otimes I_{\mathtt{q}} & \mathbf{1}_N \otimes I_{\mathtt{q}}\end{bmatrix}$ and

$$\begin{bmatrix}\operatorname{vec}(P^{(k+1)})\\ \widehat{c}^{(k+1)}\end{bmatrix} := \left(\mathbf{L}^\top W \mathbf{L}\right)^{-1}\mathbf{L}^\top W \operatorname{vec}(D).$$

6: Let $\widehat{D}^{(k+1)} := P^{(k+1)}L^{(k+1)}$.
7: $k = k+1$.
8: **until** $\|\widehat{D}^{(k)} - \widehat{D}^{(k-1)}\|_W / \|\widehat{D}^{(k)}\|_W < \varepsilon$.

Output: Locally optimal solution $\widehat{c} := \widehat{c}^{(k)}$ and $\widehat{D}^* = \widehat{D}^{(k)}$ of (WLRA$_{c,P}$).

Variable projection algorithm

The variable projection approach is based on the observation that (WLRA$_{c,P}$) is a double minimization problem

$$\text{minimize} \quad \text{over } P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}} \quad M(P)$$

where the inner minimization is a weighted least squares problem

$$M(P) := \min_{L\in\mathbb{R}^{\mathtt{m}\times N},\ c\in\mathbb{R}^{\mathtt{q}}} \|D - PL - c\mathbf{1}_N^\top\|_W$$

and therefore can be solved analytically. This reduces the original problem to a nonlinear least squares problem over P only. We have that

$$M(P) = \sqrt{\operatorname{vec}^\top(D)W\mathbf{P}\big(\mathbf{P}^\top W\mathbf{P}\big)^{-1}\mathbf{P}^\top W\operatorname{vec}(D)},$$

where

$$\mathbf{P} := \begin{bmatrix} I_N \otimes P & \mathbf{1}_N \otimes I_{\mathtt{q}} \end{bmatrix}.$$

For the outer minimization a nonlinear (least squares) algorithm is used.

Example 8.8. For the same data, initial approximation, and convergence tolerance as in Example 8.7, the variable projection algorithm, using numerical approximation of the derivatives in combination with quasi-Newton method converges to a locally optimal solution with approximation error 0.1477—the same as the one found by the alternating projections algorithm. The optimal parameters found by the two algorithms are equivalent up to the nonuniqueness of the solution, see Theorem 8.5.

8.1.4 Hankel structured low-rank approximation with centering

Consider the Hankel structured low-rank approximation problem with centering

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{w} \text{ and } c \quad \|w - c - \widehat{w}\|_2 \\ \text{subject to} & \operatorname{rank}\big(\mathscr{H}_{n+1}(\widehat{w})\big) \leq n, \end{array} \qquad \text{(HLRA}_c\text{)}$$

where w is a scalar-valued sequence $\big(w(1),\ldots,w(T)\big)$.

Variable projection algorithm

Using the kernel representation of the rank constraint

$$\operatorname{rank}\big(\mathscr{H}_{n+1}(\widehat{w})\big) \leq r \quad \Longleftrightarrow \quad \begin{array}{c}\text{there is full rank matrix } R \in \mathbb{R}^{1\times\mathtt{n}+1} \\ \text{such that } R\mathscr{H}_{n+1}(w_{\mathrm{d}}) = 0,\end{array}$$

we have

$$R\mathscr{H}_{n+1}(\widehat{w}) = 0 \quad \Longleftrightarrow \quad \mathscr{T}_{T-n}(R)\widehat{w} = 0,$$

where $\mathscr{T}_{T-n}(c)$ is an upper triangular Toeplitz matrix, defined in ($\mathscr{T}$) on page 117. Let $\mathbf{P}$ be a full rank matrix, such that

$$\operatorname{image}(\mathbf{P}) = \ker\big(\mathscr{T}_{T-n}(R)\big).$$

Then the constraint of (HLRA$_c$) can be replaced by "there is v, such that $\widehat{w} = \mathbf{P}v$," which leads to the following problem equivalent to (LRA$_c$)

$$\text{minimize} \quad \text{over } R \quad M(R),$$

where

$$M(R) := \min_{c,v} \left\| w - \begin{bmatrix}\mathbf{1}_N & \mathbf{P}\end{bmatrix}\begin{bmatrix} c \\ v \end{bmatrix}\right\|_2.$$

The evaluation of M for a given R is a standard least squares problem.

Example 8.9 (Suboptimality of the two-stage procedure for structured low-rank approximation with centering.). The data sequence is $w(t) = 0.9^t + 1$, for $t = 1,\ldots,10$. The sequence $(0.9^1,\ldots,0.9^{10})$ satisfies a difference equation $\sigma w = aw$, however, a shifted sequence $w(t) = 0.9^t + c$, with $c \neq 0$, does not satisfy such an equation. The mean of the data is $\mathbf{M}(w) = 1.5862$, so that the centered data $w(t) - \mathbf{M}(w)$ is not a trajectory of a first order autonomous linear time-invariant model. Solving the Hankel structured low-rank approximation problem with centering (HLRA$_c$) yields the exact solution $\widehat{c} = 1$.

Preprocessing by centering the data is a common practice in system identification. Example 8.9 shows that preprocessing may lead to suboptimal results. Therefore, there is a need for methods that combine data preprocessing with the existing identification methods. The algorithm derived in this section is such a method for identification of a scalar autonomous system.

8.2 Approximate low-rank factorization

As shown in the previous chapters, prior knowledge about the true data generating system plays an important role in data modeling. How important? In fact, without prior knowledge, no method can do better than a random guess. Our aim is to develop methods that exploit as effectively as possible the available prior knowledge.

In system identification, prior knowledge is specified via the model class and the approximation criterion. The model class corresponds to the matrix structure and the rank constraint in the low-rank approximation setting. We consider different types of structure in order to incorporate different model classes met in applications, see Table 1.1. The prior knowledge related to the approximation criterion corresponds

to information about the noise, *e.g.*, 2-norm cost function corresponds to Gaussian measurement noise and the weight matrix corresponds to the noise covariance.

In general, either the addition of the structure constraints or the replacement of the Frobenius norm with a weighted norm, makes the modified low-rank approximation problem difficult to solve. A globally optimal solution can no longer be given in terms of the singular values and the resulting optimization problem is nonconvex.

In addition to the prior knowledge expressed as a structure constraint, low-rank constraint, and weighted cost function, section considers prior knowledge expressed in terms of a rank revealing factorization of the true data matrix $\bar{D} = \bar{P}\bar{L}$. In order to make the factorization unique, we normalize the $\bar{P}$ factor as $\bar{P} = \begin{bmatrix} I_{\mathrm{m}} \\ \bar{P}' \end{bmatrix}$. Examples of prior knowledge encountered in bioinformatics are nonnegativity of $\bar{P}'$ and $\bar{L}$, sparsity of $\bar{P}'$, and small variation of the rows of $\bar{L}$. "Using the prior knowledge" means imposing the same structure that is present in the true data $\bar{D}$ also on the approximation $\widehat{D}$. This in turn amounts to adding constraints in the optimization problem.

Next, we present an alternating projections algorithm for low-rank approximation with structured factors. We use the alternating projections approach, because it lends itself naturally to the factorization $\widehat{D} = \widehat{P}\widehat{L}$ and is easier to modify when there are constraints on $\widehat{P}$ and $\widehat{L}$. Certain constrained problems can be treated also using a modification of the variable projection method.

8.2.1 Prior knowledge and problem formulation

In order to make the parameters $\bar{P}$ and $\bar{L}$ unique, we impose the normalization

$$\bar{P} = \begin{bmatrix} I_{\mathrm{m}} \\ \bar{P}' \end{bmatrix}. \tag{A$_1$}$$

In addition, elements specified by a selector matrix S are equal to zero:

$$S\operatorname{vec}(\bar{P}') = 0. \tag{A$_2$}$$

The parameter $\bar{L}$ is periodic with a hyper-parameter the period $l \in \mathbb{Z}_+$:

$$\bar{L} = \mathbf{1}_l^\top \otimes \bar{L}', \tag{A$_3$}$$

where $\bar{L}'$ is nonnegative

$$\bar{L}' \geq 0, \tag{A$_4$}$$

and has smooth rows in the sense that

$$\|\bar{L}'\mathbf{D}\|_{\mathrm{F}}^2 \leq \delta, \tag{A$_5$}$$

with $\mathbf{D}$ being the finite difference matrix

$$\mathbf{D} := \begin{bmatrix} 1 & & & -1 \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}$$

and $\delta > 0$ a smoothness hyper-parameter.

The data D is generated in the errors-in-variables setup (EIV), where the noise $\widetilde{D}$ is assumed to be element-wise uncorrelated with standard deviations Σ_{ij}, $i = 1,\ldots,\mathrm{q}$, $j = 1,\ldots,N$. Then, under assumptions (A$_1$–A$_5$) the maximum likelihood estimator for the parameters $\bar{P}$ and $\bar{L}$ is given by the following optimization problem:

$$\begin{aligned}
&\text{minimize} \quad \text{over } P', L', \text{ and } \widehat{D} && \|D - \widehat{D}\|_\Sigma && \text{(cost function)} && (\mathrm{C}_0)\\
&\text{subject to} && \widehat{D} = PL && \text{(rank constraint)} && \\
& && P = \begin{bmatrix} I_{\mathrm{m}} \\ P' \end{bmatrix} && \text{(normalization of } P) && (\mathrm{C}_1)\\
& && S\operatorname{vec}(P') = 0 && \text{(zero elements of } P') && (\mathrm{C}_2)\\
& && L = \mathbf{1}_l^\top \otimes L' && \text{(periodicity of } L) && (\mathrm{C}_3)\\
& && L' \geq 0 && \text{(nonnegativity of } L') && (\mathrm{C}_4)\\
& && \|L'\mathbf{D}\|_{\mathrm{F}}^2 \leq \delta && \text{(smoothness of } L') && (\mathrm{C}_5)
\end{aligned}$$

8.2.2 Computational algorithm

The alternating projections algorithm, see Algorithm 10, is based on the observation that the cost function (C$_0$) is quadratic and the constraints (C$_1$–C$_5$) are linear in either P or L. Therefore, for a fixed value of P, (C$_0$–C$_5$) is a nonnegativity constrained least squares problem in L and vice verse, for a fixed value of L, (C$_0$–C$_5$) is a constrained least squares problem in P. These problems correspond to, respectively, steps 1 and 2 of the algorithm. Geometrically they are projections. In the unweighted and unconstrained case, the problem on step 1 is the orthogonal projection

$$\widehat{D} = DL^\top(LL^\top)^{-1}L^\top = D\Pi_L$$

of the row of D on $\operatorname{row\,span}(L)$. Similarly, the problem on step 2 is the projection

$$\widehat{D} = P(P^\top P)^{-1}P^\top D = \Pi_P D$$

of the columns of D on $\operatorname{col\,span}(P)$.

Theorem 8.10 (Markovsky and Niranjan (2008)). *Algorithm 10 is globally and monotonically convergent in the* $\|\cdot\|_\Sigma$ *norm,* i.e., *if*

$$\widehat{D}^{(k)} := P^{(k)}L^{(k)}$$

Algorithm 10 Alternating projections algorithm for solving problem (C_0–C_5).

- Find an initial approximation $(P'^{(0)}, L'^{(0)})$ from centering and basic low-rank approximation.
- For $k = 0, 1, \ldots$ till convergence do
  1. $P'^{(k+1)} := \arg\min_{P'} \|D - PL\|_\Sigma$ subject to (C_1–C_2) with $L' = L'^{(k)}$
  2. $L'^{(k+1)} := \arg\min_{L'} \|D - PL\|_\Sigma$ subject to (C_3–C_5) with $P' = P'^{(k+1)}$

is the approximation on the kth step of the algorithm, then

$$f(k) := \|D - \widehat{D}^{(k)}\|_\Sigma^2 \to f^*, \quad \text{as } k \to \infty. \qquad (f(k) \to f^*)$$

*Assuming that there exists a solution to the problem (C_0–C_5) and any (locally optimal) solution is unique (*i.e.*, it is a strict minimum), the sequences $\widehat{D}^{(k)}$, $P^{(k)}$, and $L^{(k)}$ converge element-wise,* i.e.*,*

$$\widehat{D}^{(k)} \to D^*, \quad P^{(k)} \to P^*, \quad \text{and} \quad L^{(k)} \to L^*, \quad \text{as } k \to \infty, \qquad (D^{(k)} \to D^*)$$

where $\widehat{D}^* := P^* L^*$ is a (locally optimal) solution of (C_0–C_5).

8.2.3 Implementation details

The optimization problem on step 1 of the algorithm is computed separately for each row p^i of P. Let d^i be the ith row of D and $\Sigma_{i,:}$ be the ith row of Σ. The problem

$$\text{minimize} \quad \text{over } P \quad \|D - PL\|_\Sigma^2 \quad \text{subject to} \quad (C_1\text{–}C_2)$$

is equivalent to $\mathtt{m}$ problems

$$\text{minimize} \quad \text{over } p^i \qquad \|d^i - p^i L\|_{\Sigma_{i,:}} \quad \text{subject to} \quad (C_1\text{–}C_2). \qquad (*)$$

The optimization problem on step 2 is not separable due to constraint (C_5).

Taking into account constraint (C_1)

Since the first $\mathtt{m}$ rows of P are fixed, we do not solve ($*$) for $i = 1, \ldots, \mathtt{m}$, but define

$$p^i := e_i^\top, \quad \text{for } i = 1, \ldots, \mathtt{m},$$

where e_i is the ith unit vector (the ith column of the identity matrix $I_\mathtt{m}$).

Taking into account constraint (C_2)

Let S_i be a selector matrix for the zeros in the ith row of P

$$S \operatorname{vec}(P') = 0 \quad \Longleftrightarrow \quad p^i S_i = 0, \quad \text{for } i = \mathtt{m}+1, \ldots, \mathtt{q}.$$

The ith problem in ($*$) becomes

$$\text{minimize} \quad \text{over } p^i \qquad \|d^i - p^i L\|_{\Sigma_{i,:}} \quad \text{subject to} \quad p^i S_i = 0. \qquad (**)$$

Let the rows of the matrix N_i form a basis for the left null space of S_i. Then $p^i S_i = 0$ if and only if $p^i = z_i N_i$, for certain z_i, and problem ($**$) becomes

$$\text{minimize} \quad \text{over } z_i \quad \|d^i - z_i N_i L\|_{\Sigma_{i,:}}.$$

Therefore, the solution of ($*$) is

$$p^{i,*} = d^i L^\top N_i^\top (N_i L L^\top N_i^\top)^{-1} N_i.$$

Note 8.11. It is not necessary to explicitly construct the matrices S_i and compute basis N_i for their left null spaces. Since S_i is a selector matrix, it is a submatrix of the identity matrix $I_\mathtt{m}$. The rows of the complementary submatrix of $I_\mathtt{m}$ form a basis for the left null space of S_i. This particular matrix N_i is also a selector matrix, so that the product $N_i L$ need not be computed explicitly.

Taking into account constraint (C_3)

We have,

$$\begin{aligned} D - PL &= D - P(\mathbf{1}_l^\top \otimes L') = \begin{bmatrix} D_1 & \cdots & D_l \end{bmatrix} - P \begin{bmatrix} L' & \cdots & L' \end{bmatrix} \\ &= \begin{bmatrix} D_1 \\ \vdots \\ D_l \end{bmatrix} - \begin{bmatrix} P \\ \vdots \\ P \end{bmatrix} L' =: D' - \underbrace{(\mathbf{1}_l \otimes P)}_{P'} L' = D' - P'L'. \end{aligned}$$

Then the problem

$$\text{minimize} \quad \text{over } L \quad \|D - PL\|_\Sigma^2 \quad \text{subject to} \quad (C_3\text{–}C_5)$$

is equivalent to the problem

$$\text{minimize} \quad \text{over } L' \quad \|D' - P'L'\|_{\Sigma'}^2 \quad \text{subject to} \quad (C_4\text{–}C_5),$$

where $\Sigma' := \operatorname{vec}(\Sigma_{1:l,:})$.

Taking into account constraint (C_4)

Adding the nonnegativity constraint changes the least squares problem to a nonnegative least squares problem. It does not admit an analytical solution but due to convexity it can be solved globally and efficiently. We use an active-set algorithm (Gill et al, 1999), which is implemented in the function `lsqlin` of MATLAB.

Taking into account constraint (C_5)

The problem

$$\text{minimize} \quad \text{over } L \quad \|D-PL\|_\Sigma^2 \quad \text{subject to} \quad \|L\mathbf{D}\|_{\mathrm{F}}^2 \leq \delta$$

is equivalent to a regularized least squares problem

$$\text{minimize} \quad \text{over } L \qquad \|D-PL\|_\Sigma^2 + \gamma\|L\mathbf{D}\|_{\mathrm{F}}^2$$

for certain regularization parameter γ. The latter problem is equivalent to the standard least squares problem

$$\text{minimize} \quad \text{over } L \quad \left\| \begin{bmatrix} \operatorname{diag}\big(\operatorname{vec}(\Sigma)\big)\operatorname{vec}(D) \\ 0 \end{bmatrix} - \begin{bmatrix} \operatorname{diag}\big(\operatorname{vec}(\Sigma)\big)(I\otimes P) \\ \sqrt{\gamma}(\mathbf{D}^\top \otimes I) \end{bmatrix} \operatorname{vec}(L) \right\|.$$

8.3 Complex least squares with constrained phase

The problem considered in this section is defined as follows.

Problem 8.12. Given a complex valued $m \times n$ matrix A and an $m \times 1$ vector b, find a real valued $n \times 1$ vector x and a number ϕ, such that the equation's error or residual of the overdetermined system of linear equations

$$Axe^{\mathbf{i}\phi} \approx b, \quad (\mathbf{i} \text{ is the imaginary unit})$$

is minimized in the least squares sense, *i.e.*,

$$\text{minimize} \quad \text{over } x \in \mathbb{R}^n \text{ and } \phi \in (-\pi, \pi] \quad \|Axe^{\mathbf{i}\phi} - b\|. \qquad \text{(CLS)}$$

Problem (CLS) is a complex linear least squares problem with constraint that all elements of the solution have the same phase.

Problem (CLS) is nonconvex. General purpose local optimization methods can be used for solving it, however, this approach has the usual disadvantages of local optimization methods: need of initial approximation, no guarantee of global optimality, convergence issues, and no insight in the geometry of the solutions set. In (Bydder, 2010) the following closed form solution to (CLS) is derived

$$\widehat{x} = \big(\mathbb{R}_{\mathrm{e}}(A^HA)\big)^+ \mathbb{R}_{\mathrm{e}}(A^Hbe^{-\mathbf{i}\phi}) \qquad \text{(SOL1 } \widehat{x}\text{)}$$

$$\widehat{\phi} = \frac{1}{2}\angle\big((A^Hb)^\top \mathbb{R}_{\mathrm{e}}(A^HA)^+(A^Hb)\big), \qquad \text{(SOL1 } \widehat{\phi}\text{)}$$

where $\mathbb{R}_{\mathrm{e}}(A)/\mathscr{I}_{\mathrm{m}}(A)$ is the real/imaginary part, $\angle(A)$ is the angle, A^H is the complex conjugate transpose, and A^+ is the pseudoinverse of A. Moreover, in the case when a solution of (CLS) is not unique, (SOL1 $\widehat{x}$, SOL1 $\widehat{\phi}$) is a least norm element of the solution set, *i.e.*, a solution (x, ϕ), such that $\|x\|$ is minimized. Expression (SOL1 $\widehat{x}$) is the result of minimizing the cost function $\|Axe^{\mathbf{i}\phi} - b\|$ with respect to x, for a fixed ϕ. This is a linear least squares problems. Then minimization of the cost function with respect to ϕ, for x fixed to its optimal value (SOL1 $\widehat{x}$), leads to (SOL1 $\widehat{\phi}$).

8.3.1 Solution

Problem (CLS) is equivalent[1] to the problem

$$\text{minimize} \quad \text{over } x \in \mathbb{R}^n \text{ and } \phi' \in (-\pi, \pi] \quad \|Ax - be^{\mathbf{i}\phi'}\|_2, \qquad \text{(CLS')}$$

where $\phi' = -\phi$. With

$$y_1 := \mathbb{R}_{\mathrm{e}}(e^{\mathbf{i}\phi'}) = \cos(\phi') = \cos(\phi) \quad \text{and} \quad y_2 := \mathscr{I}_{\mathrm{m}}(e^{\mathbf{i}\phi'}) = \sin(\phi') = -\sin(\phi),$$

we have

$$\begin{bmatrix} \mathbb{R}_{\mathrm{e}}(be^{\mathbf{i}\phi'}) \\ \mathscr{I}_{\mathrm{m}}(be^{\mathbf{i}\phi'}) \end{bmatrix} = \begin{bmatrix} \mathbb{R}_{\mathrm{e}}(b) & -\mathscr{I}_{\mathrm{m}}(b) \\ \mathscr{I}_{\mathrm{m}}(b) & \mathbb{R}_{\mathrm{e}}(b) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}.$$

Then, (CLS') is furthermore equivalent to the problem

$$\begin{aligned} &\text{minimize} \quad \text{over } x \in \mathbb{R}^n \text{ and } y \in \mathbb{R}^2 \quad \left\| \begin{bmatrix} \mathbb{R}_{\mathrm{e}}(A) \\ \mathscr{I}_{\mathrm{m}}(A) \end{bmatrix} x - \begin{bmatrix} \mathbb{R}_{\mathrm{e}}(b) & -\mathscr{I}_{\mathrm{m}}(b) \\ \mathscr{I}_{\mathrm{m}}(b) & \mathbb{R}_{\mathrm{e}}(b) \end{bmatrix} y \right\| \\ &\text{subject to} \quad \|y\|_2 = 1, \end{aligned}$$

or

$$\text{minimize} \quad \text{over } z \in \mathbb{R}^{n+2} \quad z^\top C^\top C z \quad \text{subject to} \quad z^\top D^\top D z = 1, \qquad \text{(CLS'')}$$

with

$$C := \begin{bmatrix} \mathbb{R}_{\mathrm{e}}(A) & \mathbb{R}_{\mathrm{e}}(b) & -\mathscr{I}_{\mathrm{m}}(b) \\ \mathscr{I}_{\mathrm{m}}(A) & \mathscr{I}_{\mathrm{m}}(b) & \mathbb{R}_{\mathrm{e}}(b) \end{bmatrix} \in \mathbb{R}^{2m\times(n+2)} \quad \text{and} \quad D := \begin{bmatrix} 0 & 0 \\ 0 & I_2 \end{bmatrix} \in \mathbb{R}^{(n+2)\times(n+2)}. \qquad (C, D)$$

[1] Two optimization problems are equivalent if the solution of the first can be obtained from the solution of the second by a one-to-one transformation. Of practical interest are equivalent problems for which the transformation is simpler than the original problem.

It is well known that a solution of problem (CLS") can be obtained from the generalized eigenvalue decomposition of the pair of matrices $(C^\top C, D)$. More specifically, the smallest generalized eigenvalue $\lambda_{\min}$ of $(C^\top C, D)$ is equal to the minimum value of (CLS"), *i.e.*,

$$\lambda_{\min} = \|A\widehat{x}e^{\mathbf{i}\widehat{\phi}} - b\|_2^2.$$

If $\lambda_{\min}$ is simple, a corresponding generalized eigenvector $z_{\min}$ is of the form

$$z_{\min} = \alpha \begin{bmatrix} \widehat{x} \\ -\cos(\widehat{\phi}) \\ \sin(\widehat{\phi}) \end{bmatrix},$$

for some $\alpha \in \mathbb{R}$. We have the following result.

Theorem 8.13 (Markovsky (2011)). *Let $\lambda_{\min}$ be the smallest generalized eigenvalue of the pair of matrices $(C^\top C, D)$, defined in (C, D), and let $z_{\min}$ be the corresponding generalized eigenvector. Assuming that $\lambda_{\min}$ is a simple eigenvalue, problem (CLS) has unique solution, given by*

$$\widehat{x} = \frac{1}{\|z_2\|_2} z_1, \quad \widehat{\phi} = \angle(-z_{2,1} + \mathbf{i}z_{2,2}), \quad \text{where } z_{\min} =: \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \begin{matrix} \}n \\ \}2 \end{matrix}. \tag{SOL2}$$

Remarks:

1. *Generalized eigenvalue decomposition vs generalized singular value decomposition* Since the original data are the matrix A and the vector b, the generalized singular value decomposition of the pair (C, D) can be used instead of the generalized eigenvalue decomposition of the pair $(C^\top C, D)$. This avoids "squaring" the data and is recommended from a numerical point of view.
2. *Link to low-rank approximation and total least squares* Problem (CLS") is equivalent to the generalized low-rank approximation problem

$$\begin{array}{lll} \text{minimize} & \text{over } \widehat{C} \in \mathbb{R}^{2m\times(n+2)} & \left\|(C-\widehat{C})D\right\|_{\mathrm{F}} \\ \text{subject to} & \operatorname{rank}(\widehat{C}) \leq n+1 \quad \text{and} & \widehat{C}D^\perp = CD^\perp, \end{array} \tag{GLRA}$$

   where

$$D^\perp = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{(n+2)\times(n+2)}$$

   and $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. Indeed, the constraints of (GLRA) imply that

$$\left\|(C-\widehat{C})D\right\|_{\mathrm{F}} = \|b-\widehat{b}\|_2, \quad \text{where } \widehat{b} = Axe^{\mathbf{i}\phi}.$$

   The normalization (SOL2) is reminiscent to the generic solution of the total least squares problems. The solution of total least squares problems, however, involves a normalization by scaling with the last element of a vector $z_{\min}$ in the approximate kernel of the data matrix C, while the solution of (CLS) involves normalization by scaling with the norm of the last two elements of the vector $z_{\min}$.
3. *Uniqueness of the solution and minimum norm solutions* A solution x of (CLS) is nonunique when A has nontrivial null space. This source of nonuniqueness is fixed in (Bydder, 2010) by choosing from the solutions set a least norm solution. A least norm solution of (CLS), however, may also be nonunique due to possible nonuniquess of ϕ. Consider the following example,

$$A = \begin{bmatrix} 1 & \mathbf{i} \\ -\mathbf{i} & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -\mathbf{i} \end{bmatrix},$$

   which has two least norm solutions

$$\widehat{x}e^{\mathbf{i}\phi_1} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad \widehat{x}'e^{\mathbf{i}\phi_2} = \begin{bmatrix} 0 \\ -\mathbf{i} \end{bmatrix}.$$

   Moreover, there is a trivial nonuniqueness of x and ϕ due to $xe^{\mathbf{i}\phi} = -xe^{\mathbf{i}(\phi\pm\pi)}$ with both ϕ and one of the angles $\phi \pm \pi$ in the interval $(-\pi, \pi]$.

8.3.2 Computational algorithms

Solution (SOL1 $\widehat{x}$, SOL1 $\widehat{\phi}$) gives a straightforward procedure for computing a least norm solution of problem (CLS).

214a ⟨*Complex least squares, solution by (SOL1 $\widehat{x}$, SOL1 $\widehat{\phi}$)* 214a⟩≡

```
function cx = cls1(A, b)
invM = pinv(real(A' * A)); Atb = A' * b;
phi  = 1 / 2 * angle((Atb).' * invM * Atb);
x    = invM * real(Atb * exp(-i * phi));
cx   = x * exp(i * phi);
```

Defines:
cls1, used in chunk 217b.

The computational cost of `cls1` is $O(n^2m + n^3)$.

Theorem 8.13 gives two alternative procedures—one based on the generalized eigenvalue decomposition:

214b ⟨*Complex least squares, solution by generalized eigenvalue decomposition* 214b⟩≡

```
function cx = cls2(A, b)
⟨define C, D, and n 215a⟩
[v, l] = eig(C' * C, D); l = diag(l);
l(find(l < 0)) = inf; % ignore nevative values
[ml, mi] = min(l); z = v(:, mi);
phi = angle(-z(end - 1) + i * z(end));
x   = z(1:(end - 2)) / norm(z((end - 1):end));
cx  = x * exp(i * phi);
```

Defines:
cls2, used in chunk 217b.

215a ⟨*define C, D, and n* 215a⟩≡ (214 215)
```
C = [real(A) real(b) -imag(b);
     imag(A) imag(b)  real(b)];
n = size(A, 2); D = diag([zeros(1, n), 1, 1]);
```

and the other one based on the generalized singular value decomposition:

215b ⟨*Complex least squares, solution by generalized singular value decomposition* 215b⟩≡
```
function cx = cls3(A, b)
⟨define C, D, and n 215a⟩
[u, v] = gsvd(C, D); z = v(:, 1);
phi = angle(-z(end - 1) + i * z(end));
x   = pinv(C(:, 1:n)) * [real(b * exp(- i * phi));
                         imag(b * exp(- i * phi))];
cx  = x * exp(i * phi);
```
Defines:
cls3, used in chunk 217b.

The computational cost of `cls2` is $O\big((n+2)^2m+(n+2)^3\big)$ and of `cls3` is $O\big(m^3+(n+2)^2m^2+(n+2)^2m+(n+2)^3\big)$.

Note, however, that `cls2` and `cls3` compute the full generalized eigenvalue decomposition and generalized singular value decomposition, respectively, while only the smallest generalized eigenvalue/eigenvector or singular value/singular vector pair is needed for solving (CLS). This suggests a way of reducing the computational complexity by a factor of magnitude.

The equivalence between problem (CLS) and the generalized low-rank approximation problem (GLRA), noted in remark 2 above, allows us to use the algorithm from (Golub et al, 1987) for solving problem (CLS). The resulting Algorithm 11 is implemented in the function `cls4`.

215c ⟨*Complex least squares, solution by Algorithm 11* 215c⟩≡
```
function cx = cls4(A, b)
⟨define C, D, and n 215a⟩
R   = triu(qr(C, 0));
[u, s, v] = svd(R((n + 1):(n + 2), (n + 1):end));
phi = angle(v(1, 2) - i * v(2, 2));
x   = R(1:n, 1:n) \ (R(1:n, (n + 1):end) * [v(1, 2); v(2, 2)]);
cx  = x * exp(i * phi);
```
Defines:
cls4, used in chunk 217b.

The computational cost of `cls4` is $O\big((n+2)^2m\big)$.

Table 8.1: Summary of methods for solving the complex least squares problem (CLS).

function	method	computational cost
`cls1`	(SOL1 $\widehat{x}$, SOL1 $\widehat{\phi}$)	$O(n^2m+n^3)$
`cls2`	full generalized eigenvalue decomp.	$O\big((n+2)^2m+(n+2)^3\big)$
`cls3`	full generalized singular value decomp.	$O\big(m^3+(n+2)^2m^2+(n+2)^2m+(n+2)^3\big)$
`cls4`	Algorithm 11	$O\big((n+2)^2m\big)$

Algorithm 11 Solution of (CLS) using generalized low-rank approximation.

Input: $A \in \mathbb{C}^{m\times n}$, $b \in \mathbb{C}^{m\times 1}$

1: QR factorization of C, $QR = C$.

2: Define $R =: \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{matrix} \}n \\ \}2 \end{matrix}$, where $R_{11} \in \mathbb{R}^{n\times n}$.

3: Singular value decomposition of R_{22}, $U\Sigma V^\top = R_{22}$.

4: Let $\widehat{\phi} := \angle(v_{12} - \mathbf{i}v_{22})$ and $\widehat{x} := R_{11}^{-1}R_{12}\begin{bmatrix} v_{12} \\ v_{22} \end{bmatrix}$.

Output: $\widehat{x}e^{\mathbf{i}\widehat{\phi}}$

Numerical examples

Generically, the four solution methods implemented in the functions `cls1`, ..., `cls4` compute the same result, which is equal to the unique solution of problem (CLS). As predicted by the theoretical computation costs, the method based on Algorithm 11 is the fastest of the four methods when both the number of equations and the number of unknowns is growing, see Figure 8.2.

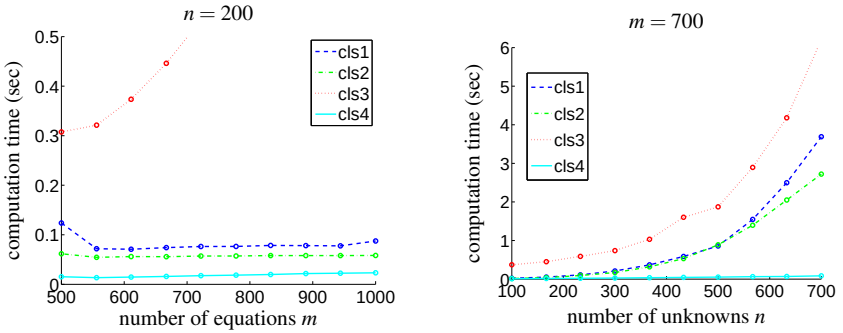


Fig. 8.2: The empirical results of the computation time match the theoretical results of the computational complexity, shown in Table CLS.

The figures are generated by using random test data

216a ⟨*Computation time for* `cls1-4` 216a⟩≡ 216b▷
```
⟨initialize the random number generator 25⟩
mm = 1000; nm = 1000; s = {'b-' 'g-.' 'r:' 'c-'};
Am = rand(mm, nm) + i * rand(mm, nm);
bm = rand(mm, 1 ) + i * rand(mm, 1 );
```

and solving problems with increasing number of equations m

216b ⟨*Computation time for* `cls1-4` 216a⟩+≡ ◁216a 217a▷
```
Nm = 10; M = round(linspace(500, 1000, Nm)); n = 200;
for j = 1:Nm, m = M(j); ⟨call cls1-4 217b⟩ end
k = 1; x = M; ax = [500 1000 0 0.5]; name = 'cls-f1';
⟨plot cls results 217c⟩
```

as well as increasing number of unknowns n

217a ⟨*Computation time for* `cls1-4` 216a⟩+≡ ◁216b
```
Nn = 10; N = round(linspace(100, 700, Nn)); m = 700;
for j = 1:Nn, n = N(j); ⟨call cls1-4 217b⟩ end
k = 2; x = N; ax = [100 700 0 6]; name = 'cls-f2';
⟨plot cls results 217c⟩
```

217b ⟨*call* `cls1-4` 217b⟩≡ (216b 217a)
```
A = Am(1:m, 1:n); b = bm(1:m);
for i = 1:4 % cls1, cls2, cls3, cls4
  eval(sprintf('tic, x = cls%d(A, b); t(%d) = toc;', i, i))
end
T(:, j) = t';
```
Uses `cls1` 214a, `cls2` 214b, `cls3` 215b, and `cls4` 215c.

217c ⟨*plot* `cls` *results* 217c⟩≡ (216b 217a)
```
figure(k), hold on
for i = 1:4 plot(x, T(i, :), s{i}, 'linewidth', 2), end
legend('cls1', 'cls2', 'cls3', 'cls4')
for i = 1:4, plot(x, T(i, :), [s{i}(1) 'o']), end
axis(ax), print_fig(name)
```

8.4 Blind identification with deterministic input model

The blind identification problem aims to find an input/output model from observations of the output only. Without prior knowledge about the input, blind identification is an ill-posed problem, *i.e.*, there are infinitely many solutions that explain the data equally well. Section 8.4.1 defines a blind identification problem with prior knowledge that the input is generated by a given autonomous linear time-invariant model. Analysis of the problem in Section 8.4.2 shows that knowledge of the input model is insufficient. We add an additional assumption that the zeros of the model are a priori known, which makes the problem well posed. Solution methods, based on low-rank approximation, are presented in Section 8.4.3.

8.4.1 Problem formulation

The setup of the problem considered in this section is shown in Figure 8.3. The output y of an unknown linear time-invariant system $\mathscr{B}_y$ to an input u and initial condition $x_{\text{ini},y}$ is observed. The unobserved input u is generated by a known autonomous linear time-invariant model $\mathscr{B}_u$ under initial condition $x_{\text{ini},u}$. The aim is to find $\mathscr{B}_y$ from y, using the prior knowledge that $u \in \mathscr{B}_u$.

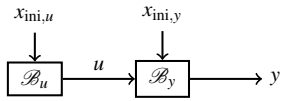


Fig. 8.3: The setup of the blind identification problem is a series connection of an autonomous system $\mathscr{B}_u$ with output u and a system $\mathscr{B}_y$ with input u and output y. The goal is to find $\mathscr{B}_y$, given y and $\mathscr{B}_u$.

Problem 8.14 (Blind identification with deterministic input model). Given a system $\mathscr{B}_u \in \mathscr{L}_{0,\mathtt{n}_u}$, output signal $y_{\mathrm{d}} \in \mathbb{R}^T$, and model class $\mathscr{L}_{1,\mathtt{n}_y}$, find a system $\mathscr{B}_y \in \mathscr{L}_{1,\mathtt{n}_y}$ and an input signal $u_{\mathrm{d}} \in \mathscr{B}_u$, such that $\left[\begin{smallmatrix} u_{\mathrm{d}} \\ y_{\mathrm{d}} \end{smallmatrix}\right] \in \mathscr{B}_y$.

Examples of signals $u \in \mathscr{B}_u$, where $\mathscr{B}_u \in \mathscr{L}^{\mathtt{m}}_{0,\mathtt{n}_u}$ are:

- *constant*
$$u(t) = c, \quad \text{for } \mathscr{B}_u := \{\, u \mid \sigma u = u \,\},$$
- *exponential signal* (with known base λ)
$$u(t) = c\lambda^t, \quad \text{for } \mathscr{B}_u = \{\, u \mid (\sigma - \lambda)u = 0 \,\},$$
- *ramp signal*
$$u(t) = ct, \quad \text{for } \mathscr{B}_u = \{\, u \mid (\sigma - 1)^2 u = 0 \,\},$$
- *sine signal* (with known frequency ω)
$$u(t) = c\sin(\omega t), \quad \text{for } \mathscr{B}_u = \{\, u \mid (\sigma - e^{\mathbf{i}\omega t})(\sigma - e^{-\mathbf{i}\omega t})u = 0 \,\}.$$

The blind system identification problem 8.14 can alternatively be viewed as a special gray-box identification problem, where the data y is generated by a system that is a series connection of a known autonomous system an unknown system, or as an input estimation problem with unknown model dynamics and prior knowledge that the input is a response of a known autonomous model.

The analysis of Problem 8.14 shows that it is ill-posed, *i.e.*, in general, the input u can not be estimated uniquely from y and $\mathscr{B}_u$. Additional information is needed in order to ensure uniqueness of the solution. We add the following assumption.

Assumption 8.15 *The zeros and the DC-gain of $\mathscr{B}_y$ are known.*

The implication of this assumption is that in a transfer function representation of
$$\mathscr{B}_y = \{\, (u,y) \mid q_y(\sigma)u = p_y(\sigma)y \,\}$$
with p_y monic, the polynomial q_y is known. Equivalently, we assume that $\mathscr{B}_y$ has no zeros and the DC-gain is equal to one, *i.e.*, with p_y monic, $q_y = 1$.

8.4.2 Analysis of the problem in the single-input single-output case

In this section, we consider exact (noise free) data. Let

$$\mathscr{B}_u = \{\, u \mid p_u(\sigma)u = 0 \,\} \quad \text{and} \quad \mathscr{B}_y = \{\, (u,y) \mid q_y(\sigma)u = p_y(\sigma)y \,\}.$$

The interconnection of $\mathscr{B}_u$ and $\mathscr{B}_y$ is an autonomous linear time-invariant system $\mathscr{B} \in \mathscr{L}^1_{0,\mathtt{n}}$, of order $\mathtt{n} := \mathtt{n}_u + \mathtt{n}_y$, with representation

$$\mathscr{B} = \{\, y \mid p_y(\sigma)p_u(\sigma)y = 0 \,\}.$$

Therefore, the Z-transform $\mathscr{Z}(y)$ of $y \in \mathscr{B}$ is a rational function $\mathscr{Z}(y) = q/p$, with $p = p_u p_y$. Next, we derive the polynomial q.

For $u \in \mathscr{B}_u$ and $(u,y) \in \mathscr{B}_y$, there is are polynomial $x_{\mathrm{ini},u}$ and $x_{\mathrm{ini},y}$ of degrees $\mathtt{n}_u - 1$ and $\mathtt{n}_y - 1$, respectively, such that $\mathscr{Z}(u) = x_{\mathrm{ini},u}/p_u$ and

$$\mathscr{Z}(y) = \frac{q_y}{p_y}\mathscr{Z}(u) + \frac{x_{\mathrm{ini},y}}{p_y}.$$

Using these relations we obtain

$$\mathscr{Z}(y) = \frac{q_y}{p_y}\frac{x_{\mathrm{ini},u}}{p_u} + \frac{x_{\mathrm{ini},y}}{p_y} = \frac{q_y x_{\mathrm{ini},u} + p_u x_{\mathrm{ini},y}}{p_u p_y}, \qquad (*)$$

so that

$$q = q_y x_{\mathrm{ini},u} + p_u x_{\mathrm{ini},y}.$$

The polynomials p and q can be found from the data y. (This is an exact identification problem.) Then, since p_u is given, finding p_y from $p = p_u p_y$ is a deconvolution problem. Computing q_y and the initial conditions $x_{\mathrm{ini},u}$ and $x_{\mathrm{ini},y}$ from q, however, is an ill-posed problem (it has a nonunique solution). This issue is not a problem of a particular solution method. The nonuniqueness in determining $x_{\mathrm{ini},u}$ and q_y shows that Problem 8.14 is ill-posed.

Using Assumption 8.15, *i.e.*, $q_y = 1$, $(*)$ becomes

$$q = x_{\mathrm{ini},u} + p_u x_{\mathrm{ini},y} = \begin{bmatrix} 1 & p_u \end{bmatrix} \begin{bmatrix} x_{\mathrm{ini},u} \\ x_{\mathrm{ini},y} \end{bmatrix}$$

or, equivalently

$$\left[\begin{bmatrix} I_{\mathtt{n}_u} \\ 0_{\mathtt{n}_y - 1 \times \mathtt{n}_u} \end{bmatrix} \quad \mathscr{M}_{\mathtt{n}_y - 1}(p_u) \right] \begin{bmatrix} x_{\mathrm{ini},u} \\ x_{\mathrm{ini},y} \end{bmatrix} = q. \qquad (**)$$

The system $(**)$ defines a well-posed problem for the estimation of $x_{\mathrm{ini},u}$ and $x_{\mathrm{ini},y}$ from the given p_u and the identified q.

8.4.3 Solution method

The analysis done in the previous section suggest the following solution method:

1. *identification:* compute an input/output representation $\mathscr{B}_{\mathrm{i/o}}(p,q)$ of $\mathscr{B}_{\mathrm{mpum}}(y_{\mathrm{d}})$,
2. *deconvolution:* compute $\mathscr{B}_y$ from $\mathscr{B}_{\mathrm{i/o}}(p,q)$ and $\mathscr{B}_u$, and
3. *estimation:* find the initial condition $(x_{\mathrm{ini},u}, x_{\mathrm{ini},y})$ from $(**)$.

With exact data the identification of $\mathscr{B}_{\mathrm{mpum}}(y_{\mathrm{d}})$ need not exploit the prior knowledge of the $\mathscr{B}_u$ model. In case of inexact data when approximation is involved, however, the prior knowledge should be used. This leads to a grey-box identification problem.

Two methods for solving the grey-box identification problem are presented. The first one is a heuristic subspace methods. The second one is a maximum-likelihood method that leads to a Hankel structured low-rank approximation with a linear equality constraint on the kernel parameter R. Both methods find directly $\mathscr{B}_y$, using Assumption 8.15. This eliminates the need of solving the deconvolution problem on step 2. With knowledge of $\mathscr{B}_y$, the maximum-likelihood estimate of the initial condition $(x_{\mathrm{ini},u}, x_{\mathrm{ini},y})$ is given by a Kalman filter. This is a more robust alternative to the one of solving the system of linear equations $(**)$.

Subspace method

The subspace method is based on the fact that with exact data

$$p_y(\sigma)p_u(\sigma)y_{\mathrm{d}} = 0,$$

so that the signal

$$y'_{\mathrm{d}} := p_u(\sigma)y_{\mathrm{d}},$$

is annihilated by $p_y(\sigma)$ and, therefore,

$$y'_{\mathrm{d}} \in \mathscr{B}'_y := \ker\big(p_y(\sigma)\big).$$

Since y'_{d} is commutable from the given data, assuming that y'_{d} is persistently exciting of order $\mathtt{n}_y$, we can find $\mathscr{B}'_y$ by computing $\mathscr{B}_{\mathrm{mpum}}(y'_{\mathrm{d}})$. Once $\mathscr{B}'_y$ is computed, under Assumption 8.15, the model $\mathscr{B}_y$ is known. In case of noisy data, the computation of an estimate of $\mathscr{B}_y$ involves approximation, see Exercise 4.4.

Maximum-likelihood method

Let the data be generated in the output error setting, *i.e.*,

$$y_{\mathrm{d}} = \bar{y} + \tilde{y}, \qquad \text{where } \tilde{y} \sim \mathrm{N}(0, s^2 I),$$

is the measurement noise and $\bar{y}$ is the true output, *i.e.*, for some $\bar{u} \in \mathscr{B}_u$, $\left[\begin{smallmatrix}\bar{u}\\ \bar{y}\end{smallmatrix}\right] \in \mathscr{B}_y$. Then, the maximum-likelihood estimation problem is

$$\begin{array}{ll}\text{minimize} & \text{over } \widehat{y},\ \widehat{u},\ \text{and } \widehat{\mathscr{B}}_y \quad \|y_{\mathrm{d}} - \widehat{y}\|_2 \\ \text{subject to} & \widehat{u} \in \mathscr{B}_u \quad \text{and} \quad \left[\begin{smallmatrix}\widehat{u}\\ \widehat{y}\end{smallmatrix}\right] \in \mathscr{B}_y \in \mathscr{L}_{1,\mathtt{n}_y}.\end{array} \tag{BLSYSID}$$

The following proposition relates problem (BLSYSID) to a Hankel structured low-rank approximation problem with additional structure of the left kernel.

Proposition 8.16. *Under Assumption 8.15, problem (BLSYSID) is equivalent to the structured low-rank approximation problem*

$$\begin{array}{ll}\text{minimize} & \textit{over } \widehat{y} \textit{ and } \widehat{p}_y \quad \|y_{\mathrm{d}} - \widehat{y}\|_2 \\ \textit{subject to} & \widehat{p}_y \mathscr{T}_{\mathtt{n}_y}(p_u) \mathscr{H}(\widehat{y}) = 0.\end{array} \tag{SLRA}$$

Problem (SLRA) can be solved by the variable projection method of Section 4.4. An implementation of the method is available (Markovsky and Usevich, 2014).

8.5 Notes and references

Nonnegative low-rank approximation

A low-rank approximation problem with element-wise nonnegativity constraint

$$\begin{array}{ll}\text{minimize} & \text{over } \widehat{D} \quad \|D - \widehat{D}\|_{\mathrm{F}} \\ \text{subject to} & \operatorname{rank}(\widehat{D}) \leq \mathtt{m} \quad \text{and} \quad \widehat{D} \geq 0\end{array} \tag{NNLRA}$$

arises in Markov chains (Vanluyten et al, 2006) and image mining (Lee and Seung, 1999). Using the image representation, we obtain the following problem

$$\begin{array}{ll}\text{minimize} & \text{over } \widehat{D},\ P \in \mathbb{R}^{\mathtt{q}\times\mathtt{m}},\ \text{and } L \in \mathbb{R}^{\mathtt{m}\times N} \quad \|D - \widehat{D}\|_{\mathrm{F}} \\ \text{subject to} & \widehat{D} = PL \quad \text{and} \quad P, L \geq 0,\end{array} \tag{NNLRA$_P$}$$

which is a relaxation of problem (NNLRA). The minimal $\mathtt{m}$, for which (NNLRA$_P$) has a solution, is called the *positive rank* of $\widehat{D}$ (Berman and Shaked-Monderer, 2003). In general, the positive rank is less than or equal to the rank.

Note that due to the nonnegativity constraint on $\widehat{D}$, the problem can not be solved using the variable projection method. (There is no closed form solution for the equivalent problem with $\widehat{D}$ eliminated.) The alternating projections algorithm, however, can be used. With the Frobenius norm approximation, the projections are least squares problems with a nonnegativity constraint, see Algorithm 12.

Algorithm 12 Alternating projections algorithm for nonnegative low-rank approximation.

Input: Data matrix D, desired rank $\mathtt{m}$, and convergence tolerance ε.

1: Set $k := 0$ and compute an initial approximation $\widehat{D}^{(0)} := P^{(0)}L^{(0)}$ from the singular value decomposition by setting all negative elements to zero.
2: **repeat**
3: $\quad k := k+1$.
4: $\quad$ Solve: $L^{(k)} := \arg\min_L \|D - P^{(k-1)}L\|_{\mathrm{F}}$ subject to $L \geq 0$.
5: $\quad$ Solve: $P^{(k)} := \arg\min_P \|D - PL^{(k)}\|_{\mathrm{F}}$ subject to $P \geq 0$.
6: **until** $\|P^{(k-1)}L^{(k-1)} - P^{(k)}L^{(k)}\|_{\mathrm{F}} < \varepsilon$

Output: A locally optimal solution $\widehat{D}^* := P^{(k)}L^{(k)}$ to problem (NNLRA$_P$).

Blind identification with deterministic input model

Problem 8.14 is motivated by an application for dynamic measurements in metrology, in which case u is a to-be-measured quantity, $\mathscr{B}_y$ is the measurement device (sensor), and y is the measured value. After calibration, in a steady state $u = y$, so that the to-be-measured quantity is read out from the sensor's output. Due to the transient response of the sensor, however, exact measurement is achieved only asymptotically. Dynamic measurement methods aim to reduce the measurement error due to the transient by taking into account the dynamic properties of the sensor.

In (Markovsky, 2015), the dynamic measurement problem is viewed from a signal processing perspective as an input estimation problem, where the input is an unknown constant. In case of known sensor dynamics ($\mathscr{B}_y$ given), the problem is solved by the classical Kalman filter. In practical situations in metrology, however, $\mathscr{B}_y$ is not a priori known. For example, in mass measurement, the sensor dynamics depends on the to-be-measured mass parameter. In this case, the dynamic measurement problem becomes Problem 8.14 with $\mathscr{B}_u := \{\, u \mid \sigma u = u \,\}$.

Exercises

8.1 (Matrix centering). Prove Proposition 8.1.

8.2 (Mean computation as an optimal modeling). Prove Proposition 8.2.

8.3 (Autonomous system identification with centering, using internal model). Let $\mathscr{A}^{\mathtt{q}}_{0,\ell}$ be the class of bounded complexity *autonomous affine time-invariant models* $c + \mathscr{B}$, with $\mathscr{B} \in \mathscr{L}^{\mathtt{q}}_{0,\ell}$. The problem of finding a model in $\mathscr{A}^{\mathtt{q}}_{0,\ell}$ from data is actually the linear time-invariant identification problem with centering, considered in Section 8.1.4. This exercise explores a solution method using the *internal model principle*. Applied to the problem at hand, the internal model principle is: augment the to-be-found model $\widehat{\mathscr{B}}$ of $\mathscr{B}$ with the known model $\mathscr{B}_{\mathrm{c}} := \{\, c \mid c(t_1) = c(t_2) \text{ for all } t_1, t_2 \,\}$ of the offset c. Derive a kernel representation of the augmented model $\widehat{\mathscr{B}}_{\mathrm{ext}}$ in terms of kernel representations of $\widehat{\mathscr{B}}$ and $\mathscr{B}_{\mathrm{c}}$. Use this result to

re-formulate the 2-norm optimal autonomous system identification with centering problem as an equivalent Hankel low-rank approximation problem with a linear constraint on the kernel. Use the SLRA package to solve this latter problem.

References

Berman A, Shaked-Monderer N (2003) Completely positive matrices. World Scientific Publishing Co

Bydder M (2010) Solution of a complex least squares problem with constrained phase. Linear Algebra Appl 433(11–12):1719–1721

Gill PE, Murray M, Wright MH (1999) Practical Optimization. Academic Press

Gillis N (2011) NMF: Complexity, algorithms and applications. PhD thesis, Université catholique de Louvain

Gillis N, Glineur F (2012) Accelerated multiplicative updates and hierarchical als algorithms for nonnegative matrix factorization. Neural computation 24(4):1085–1105

Gillis N, Vavasis S (2014) Fast and robust recursive algorithmsfor separable nonnegative matrix factorization. IEEE transactions on pattern analysis and machine intelligence 36(4):698–714

Golub G, Hoffman A, Stewart G (1987) A generalization of the Eckart–Young–Mirsky matrix approximation theorem. Linear Algebra Appl 88/89:317–327

Goodwin G, Ramadge P, P C (1979) Ultimate objectives and prior knowledge in system identification. IFAC Proceedings 12(8):1123–1129

Lee D, Seung H (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401:788–791

Markovsky I (2011) On the complex least squares problem with constrained phase. SIAM J Matrix Anal Appl 32(3):987–992

Markovsky I (2015) An application of system identification in metrology. Control Eng Practice 43:85–93

Markovsky I, Niranjan M (2008) Approximate low-rank factorization with structured factors. Comput Statist Data Anal 54:3411–3420

Markovsky I, Usevich K (2014) Software for weighted structured low-rank approximation. J Comput Appl Math 256:278–292

Udell M (1015) Generalized low rank models. PhD thesis, Stanford University

Udell M, Horn C, Zadeh R, Boyd S (2016) Generalized low rank models. Foundations and Trends in Machine Learning 9(1):1–118

Vanluyten B, Willems JC, De Moor B (2006) Matrix factorization and stochastic state representations. In: Proc. 45th IEEE Conf. on Dec. and Control, San Diego, California, pp 4188–4193

Verhaegen M, Verdult V, Bergboer N (2007) Filtering and system identification: An introduction to using matlab software

Appendix A
Total least squares

> *The least squares problem that we are considering here is known by different names in different scientific disciplines. For example, mathematicians may regard the (least squares) problem as finding the closest point in a given subspace to a given point in a function space. ... Statisticians introduce probability distribution into their conception of the problem and use such terms as* regression analysis *to describe this area. Engineers reach this problem by studying such topics as* parameter estimation, filtering, *and* process identification.
>
> *Lawson and Hanson (1987, Page 2)*

Approximate solution of an overdetermined system of linear equations $AX \approx B$ is one of the main topics in (numerical) linear algebra and is covered in any linear algebra textbook, see, *e.g.*, (Strang, 1976, Section 3.3), (Meyer, 2000, Sections 4.6 and 5.14), and (Trefethen and Bau, 1997, Lecture 11). The classical approach is approximate solution in the least squares sense:

$$\text{minimize} \quad \text{over } \widehat{B} \text{ and } X \quad \|B-\widehat{B}\|_{\text{F}} \quad \text{subject to} \quad AX = \widehat{B}, \qquad \text{(LS)}$$

where the matrix B is modified as little as possible in the sense of minimizing the correction size $\|B-\widehat{B}\|_{\text{F}}$, so that the modified system of equations $AX = \widehat{B}$ is compatible. The least squares problem has an analytic solution: assuming that that the matrix A is full column rank, the unique least squares approximate solution is

$$\widehat{X}_{\text{ls}} = (A^{\top}A)^{-1}A^{\top}B \quad \text{and} \quad \widehat{B}_{\text{ls}} = A(A^{\top}A)^{-1}A^{\top}B.$$

In the case when A is rank deficient, the solution is either nonunique or does not exist. Such least squares problems are solved numerically by regularization techniques, see, *e.g.*, (Lawson and Hanson, 1987) and (Björck, 1996, Section 2.7).

There are many variations and generalizations of the least squares method for solving approximately an overdetermined system of equations. Well known ones are methods for recursive least squares approximation (Kailath et al, 2000, Section 2.6), regularized least squares (Hansen, 1997), linear and quadratically constrained least squares problems (Golub and Van Loan, 1996, Section 12.1).

Next, we list generalizations related to the class of the total least squares methods because of their close connection to corresponding low-rank approximation problems. From a data modeling point of view, total least squares is low-rank approximation using an input/output representation of the rank constraint. In all these problems the basic idea is to modify the given data as little as possible, so that the modified data defines a compatible system of equations. In the different methods, however, the correction is done and its size is measured in different ways. This results in different properties of the methods in a stochastic estimation setting and motivates the use of the methods in different practical setups.

Data least squares

The data least squares (Degroat and Dowling, 1991) method is the "reverse" of the least squares method in the sense that A is modified and B is not:

$$\text{minimize} \quad \text{over } \widehat{A} \text{ and } X \quad \|A-\widehat{A}\|_{\text{F}} \quad \text{subject to} \quad \widehat{A}X = B. \qquad \text{(DLS)}$$

The solution of the data least squares problem (DLS) can be found in closed form.

Total least squares

The classical total least squares (Golub, 1973; Golub and Reinsch, 1970; Golub and Van Loan, 1980) method modifies symmetrically the matrices A and B:

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{A}, \widehat{B}, \text{ and } X \quad \left\| \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \right\|_{\text{F}} \\ &\text{subject to} \quad \widehat{A}X = \widehat{B}. \end{aligned} \qquad \text{(TLS)}$$

Conditions for existence and uniqueness of a total least squares approximate solution are given in terms of the singular value decomposition of the augmented data matrix $\begin{bmatrix} A & B \end{bmatrix}$. In the generic case when a unique solution exists, that solution is given in terms of the right singular vectors of $\begin{bmatrix} A & B \end{bmatrix}$ corresponding to the smallest singular values. In this case, the optimal total least squares approximation $\begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix}$ of the data matrix $\begin{bmatrix} A & B \end{bmatrix}$ coincides with the Frobenius norm optimal low-rank approximation of $\begin{bmatrix} A & B \end{bmatrix}$, *i.e.*, in the generic case, the model obtained by the total least squares method coincides with the model obtained by the unstructured low-rank approximation in the Frobenius norm.

Theorem A.1. *Let $\widehat{D}^*$ be a solution to the low-rank approximation problem*

$$\text{minimize} \quad \text{over } \widehat{D} \quad \|D-\widehat{D}\|_{\text{F}} \quad \text{subject to} \quad \operatorname{rank}(\widehat{D}) \leq \mathtt{m}$$

and let $\widehat{\mathscr{B}}^* = \operatorname{image}(\widehat{D}^*)$ be the corresponding optimal linear static model. The parameter $\widehat{X}^*$ of an input/output representation $\widehat{\mathscr{B}}^* = \mathscr{B}_{\text{i/o}}(\widehat{X}^*)$ of the optimal model is a solution to the total least squares problem (TLS) with data matrices

$$\begin{matrix} \mathtt{m} \\ \mathtt{q}-\mathtt{m} \end{matrix} \begin{Bmatrix} \\ \\ \end{Bmatrix} \begin{bmatrix} A^{\top} \\ B^{\top} \end{bmatrix} = D \in \mathbb{R}^{\mathtt{q} \times N}.$$

A total least squares solution $\widehat{X}^*$ exists if and only if an input/output representation $\mathscr{B}_{\text{i/o}}(\widehat{X}^*)$ of $\widehat{\mathscr{B}}^*$ exists and is unique if and only if an optimal model $\widehat{\mathscr{B}}^*$ is unique. In the case of existence and uniqueness of a total least squares solution

$$\widehat{D}^* = \begin{bmatrix} \widehat{A}^* & \widehat{B}^* \end{bmatrix}^\top, \quad \textit{where } \widehat{A}^*\widehat{X}^* = \widehat{B}^*.$$

The theorem makes explicit the link between low-rank approximation and total least squares. From a data modeling point of view,

> total least squares is low-rank approximation of the data matrix $D = \begin{bmatrix} A & B \end{bmatrix}^\top$, followed by input/output representation of the optimal model.

227a ⟨*Total least squares* 227a⟩≡
```
function [x, ah, bh] = tls(a, b)
n = size(a, 2); [r, p, dh] = lra([a b]', n);
⟨low-rank approximation ↦ total least squares solution 227b⟩
```
Uses `lra` 105a.

227b ⟨*low-rank approximation ↦ total least squares solution* 227b⟩≡ (227a)
```
x = p2x(p)'; ah = dh(1:n, :)'; bh = dh((n + 1):end, :)';
```

Lack of solution of the total least squares problem (TLS)—a case called *nongeneric total least squares problem*—is caused by lack of existence of an input/output representation of the model. Nongeneric total least squares problems are considered in (Paige and Strakos, 2005; Van Huffel and Vandewalle, 1988, 1991).

Generalized total least squares

The generalized total least squares (Van Huffel and Vandewalle, 1989) method measures the size of the data correction matrix

$$\begin{bmatrix} \Delta A & \Delta B \end{bmatrix} := \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix}$$

after row and column weighting:

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{A}, \widehat{B}, \text{ and } X \quad \left\| W_{\mathrm{l}} \left(\begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \right) W_{\mathrm{r}} \right\|_{\mathrm{F}} \\ \text{subject to} & \widehat{A}X = \widehat{B}. \end{array} \tag{GTLS}$$

Here the matrices are W_{l} and W_{r} are positive semidefinite weight matrices—W_{l} corresponds to weighting of the rows and W_{r} to weighting of the columns of the correction $\begin{bmatrix} \Delta A & \Delta B \end{bmatrix}$. Similarly to the classical total least squares problem, the existence and uniqueness of a generalized total least squares approximate solution is determined from the singular value decomposition. The data least squares and total least squares problems are special cases of the generalized total least squares problem.

Restricted total least squares

The restricted total least squares (Van Huffel and Zha, 1991) method constrains the correction to be in the form

$$\begin{bmatrix} \Delta A & \Delta B \end{bmatrix} = P_{\mathrm{e}} E L_{\mathrm{e}}, \qquad \text{for some } E,$$

i.e., the row and column span of the correction matrix are constrained to be within the given subspaces image(P_{e}) and image$(L_{\mathrm{e}}^\top)$, respectively. The restricted total least squares problem is:

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{A}, \widehat{B}, E, \text{ and } X \quad \|E\|_{\mathrm{F}} \\ \text{subject to} & \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} = P_{\mathrm{e}} E L_{\mathrm{e}} \quad \text{and} \quad \widehat{A}X = \widehat{B}. \end{array} \tag{RTLS}$$

The generalized total least squares problem is a special case of (RTLS).

Procrustes problem

The Procrustes problem: Given $m \times n$ real matrices A and B,

$$\text{minimize over } X \quad \|B - AX\|_{\mathrm{F}} \quad \text{subject to} \quad X^\top X = I_{\mathrm{n}}$$

is a least squares problem with a constraint that the unknown X is an orthogonal matrix. The solution is given by $X = UV^\top$, where $U\Sigma V^\top$ is the singular value decomposition of $AB^\top$, see (Golub and Van Loan, 1996, page 601).

Weighted total least squares

The weighted total least squares (De Moor, 1993, Section 4.3) method generalizes the classical total least squares problem by measuring the correction size by a weighted matrix norm $\|\cdot\|_W$

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{A}, \widehat{B}, \text{ and } X \quad \left\| \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \right\|_W \\ \text{subject to} & \widehat{A}X = \widehat{B}. \end{array} \tag{WTLS}$$

Special weighted total least squares problems correspond to weight matrices W with special structure, *e.g.*, diagonal W corresponds to *element-wise weighted total least squares* (Markovsky et al, 2005). In general, the weighted total least squares problem has no analytic solution in terms of the singular value decomposition, so that contrary to the above listed generalizations, weighted total least squares problems, in general, can not be solved globally and efficiently. Weighted low-rank approximation problems are considered in (Manton et al, 2003; Markovsky and Van Huffel, 2007a; Wentzell et al, 1997).

Regularized total least squares

The regularized total least squares (Beck and Ben-Tal, 2006b; Fierro et al, 1997; Golub et al, 1999; Sima, 2006; Sima et al, 2004) methods is defined as

$$\begin{array}{ll}\text{minimize} & \text{over } \widehat{A}, \widehat{B}, \text{ and } X \quad \left\| \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \right\|_{\mathrm{F}} + \gamma \|DX\|_{\mathrm{F}} \\ \text{subject to} & \widehat{A}X = \widehat{B}. \end{array} \tag{RegTLS}$$

Global and efficient solution methods for solving regularized total least squares problems are derived in (Beck and Ben-Tal, 2006b).

Structured total least squares

The structured total least squares method (Abatzoglou et al, 1991; De Moor, 1993) method is a total least squares method with the additional constraint that the correction should have certain specified structure

$$\begin{array}{ll}\text{minimize} & \text{over } \widehat{A}, \widehat{B}, \text{ and } X \quad \left\| \begin{bmatrix} A & B \end{bmatrix} - \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \right\|_{\mathrm{F}} \\ \text{subject to} & \widehat{A}X = \widehat{B} \quad \text{and} \quad \begin{bmatrix} \widehat{A} & \widehat{B} \end{bmatrix} \text{ has a specified structure.} \end{array} \tag{STLS}$$

Hankel and Toeplitz structured total least squares problems are the most often studied ones due to the their application in signal processing and system theory.

Structured total least norm

The structured total least norm method (Rosen et al, 1996) is the same as the structured total least squares method with a general matrix norm in the approximation criterion instead of the Frobenius norm. For generalizations and applications of the total least squares problem in the periods 1990–1996, 1996–2001, and 2001–2006, see respectively the edited books (Van Huffel, 1997), (Van Huffel and Lemmerling, 2002), and the special issues (Van Huffel et al, 2007a,b). An overview of total least squares is given in (Markovsky and Van Huffel, 2007b; Markovsky et al, 2010).

Exercises

A.1 (Geometric interpretation of the total least squares). Show that the total least squares problem

$$\begin{array}{ll}\text{minimize} & \text{over } x \in \mathbb{R}, \widehat{a} \in \mathbb{R}^N, \text{ and } \widehat{b} \in \mathbb{R}^N \quad \displaystyle\sum_{j=1}^{N} \left\| d_j - \begin{bmatrix} \widehat{a}_j \\ \widehat{b}_j \end{bmatrix} \right\|_2^2 \\ \text{subject to} & \widehat{a}_j x = \widehat{b}_j, \quad \text{for } j = 1, \ldots, N \end{array} \tag{tls}$$

minimizes the sum of the squared orthogonal distances from the data points $d_1, \ldots, d_N$ to the fitting line

$$\mathscr{B} = \{ \operatorname{col}(a,b) \mid xa = b \}$$

over all lines passing through the origin, except for the vertical line.

A.2 (Unconstrained problem, equivalent to the total least squares problem). Show that (tls) is equivalent to the unconstrained optimization problem

$$\text{minimize } f_{\text{tls}}(x), \quad \text{where } f_{\text{tls}}(x) := \frac{\|ax - b\|_2^2}{\|x\|_2^2 + 1}, \tag{tls'}$$

where $a = \begin{bmatrix} a_1 & \cdots & a_N \end{bmatrix}^\top$ and $b = \begin{bmatrix} b_1 & \cdots & b_N \end{bmatrix}^\top$.

A.3 (Lack of total least squares solution). Using the formulation (tls'), derived in Problem 1.3, show that the total least squares line fitting problem (tls) has no solution for the data in Problem 1.1.

References

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

Arablouei R, Dogancay K (2012) Linearly-constrained recursive total least-squares algorithm. IEEE Signal Proc Letters 19(12):821–824

Beck A, Ben-Tal A (2006a) A global solution for the structured total least squares problem with block circulant matrices. SIAM J Matrix Anal Appl 27(1):238–255

Beck A, Ben-Tal A (2006b) On the solution of the Tikhonov regularization of the total least squares. SIAM J Optimization 17(1):98–118

Beck A, Eldar Y (2010) Structured total maximum likelihood: An alternative to structured total least squares. SIAM Journal on Matrix Analysis and Applications 31(5):2623–2649

Björck Å (1996) Numerical Methods for Least Squares Problems. SIAM

Cadzow J (1988) Signal enhancement—A composite property mapping algorithm. IEEE Trans Signal Proc 36:49–62

De Moor B (1993) Structured total least squares and L_2 approximation problems. Linear Algebra Appl 188–189:163–207

Degroat R, Dowling E (1991) The data least squares problem and channel equalization. IEEE Trans Signal Proc 41:407–411

Fierro R, Golub G, Hansen P, O'Leary D (1997) Regularization by truncated total least squares. SIAM J Sci Comp 18(1):1223–1241
Golub G (1973) Some modified matrix eigenvalue problems. SIAM Review 15:318–344
Golub G, Reinsch C (1970) Singular value decomposition and least squares solutions. Numer Math 14:403–420
Golub G, Van Loan C (1980) An analysis of the total least squares problem. SIAM J Numer Anal 17:883–893
Golub G, Van Loan C (1996) Matrix Computations, 3rd edn. Johns Hopkins University Press
Golub G, Hansen P, O'Leary D (1999) Tikhonov regularization and total least squares. SIAM J Matrix Anal Appl 21(1):185–194
Hansen PC (1997) Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion. SIAM
Hnetynkova I, Plesinger M, Sima D (2016) Solvability of the core problem with multiple right-hand sides in the TLS sense. SIAM J Matrix Anal Appl 37(3):861–876
Huang JJ, Dragotti PL (2017) Sparse signal recovery using structured total maximum likelihood. In: Int. Conf. on Sampling Theory and Applications, pp 639–643
Kailath T, Sayed AH, Hassibi B (2000) Linear Estimation. Prentice Hall
Lawson C, Hanson R (1987) Solving Least Squares Problems. Classics in Applied Mathematics, Society for Industrial and Applied Mathematics
Manton J, Mahony R, Hua Y (2003) The geometry of weighted low-rank approximations. IEEE Trans Signal Proc 51(2):500–514
Markovsky I (2008) Structured low-rank approximation and its applications. Automatica 44(4):891–909
Markovsky I (2010) Bibliography on total least squares and related methods. Statistics and Its Interface 3:329–334
Markovsky I, Van Huffel S (2007a) Left vs right representations for solving weighted low rank approximation problems. Linear Algebra Appl 422:540–552
Markovsky I, Van Huffel S (2007b) Overview of total least squares methods. Signal Proc 87:2283–2302
Markovsky I, Rastello M, Premoli A, Kukush A, Van Huffel S (2005) The element-wise weighted total least squares problem. Comput Statist Data Anal 50(1):181–209
Markovsky I, Willems JC, Van Huffel S, De Moor B (2006) Exact and Approximate Modeling of Linear Systems: A Behavioral Approach. SIAM
Markovsky I, Sima D, Van Huffel S (2010) Total least squares methods. Wiley Interdisciplinary Reviews: Comput Stat 2(2):212–217
Meyer C (2000) Matrix Analysis and Applied Linear Algebra. SIAM
Paige C, Strakos Z (2005) Core problems in linear algebraic systems. SIAM J Matrix Anal Appl 27:861–875
Rhode S, Usevich K, Markovsky I, Gauterin F (2014) A recursive restricted total least-squares algorithm. IEEE Trans Signal Process 62(21):5652–5662

Rosen J, Park H, Glick J (1996) Total least norm formulation and solution of structured problems. SIAM J Matrix Anal Appl 17:110–126
Sima D (2006) Regularization techniques in model fitting and parameter estimation. PhD thesis, ESAT, K.U.Leuven
Sima D, Van Huffel S, Golub G (2004) Regularized total least squares based on quadratic eigenvalue problem solvers. BIT 44:793–812
Strang G (1976) Linear Algebra and Its Applications. Academic Press
Trefethen L, Bau D (1997) Numerical Linear Algebra. SIAM
Van Huffel S (ed) (1997) Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modeling. SIAM, Philadelphia
Van Huffel S (2004) Total least squares and errors-in-variables modeling: Bridging the gap between statistics, computational mathematics and engineering. In: Antoch J (ed) Proc. COMPSTAT, Physika-Verlag, Heidelberg, pp 539–555
Van Huffel S, Lemmerling P (eds) (2002) Total Least Squares and Errors-in-Variables Modeling: Analysis, Algorithms and Applications. Kluwer
Van Huffel S, Vandewalle J (1988) Analysis and solution of the nongeneric total least squares problem. SIAM J Matrix Anal Appl 9:360–372
Van Huffel S, Vandewalle J (1989) Analysis and properties of the generalized total least squares problem $AX \approx B$ when some or all columns in A are subject to error. SIAM J Matrix Anal 10(3):294–315
Van Huffel S, Vandewalle J (1991) The total least squares problem: Computational aspects and analysis. SIAM, Philadelphia
Van Huffel S, Zha H (1991) The restricted total least squares problem: Formulation, algorithm and properties. SIAM J Matrix Anal Appl 12(2):292–309
Van Huffel S, Zha H (1993) The total least squares problem. In: Rao C (ed) Handbook of Statistics: Comput. Stat., vol 9, Elsevier, Amsterdam, pp 377–408
Van Huffel S, Cheng CL, Mastronardi N, Paige C, Kukush A (2007a) Editorial: Total least squares and errors-in-variables modeling. Comput Stat Data Anal 52:1076–1079
Van Huffel S, Markovsky I, Vaccaro RJ, Söderström T (2007b) Guest editorial: Total least squares and errors-in-variables modeling. Signal Proc 87(10):2281–2282
Wentzell P, Andrews D, Hamilton D, Faber K, Kowalski B (1997) Maximum likelihood principal component analysis. J Chemometrics 11:339–366
Yeredor A (2004) Multiple delays estimation for chirp signals using structured total least squares. Linear Algebra Appl 391:261–286
Yeredor A, De Moor B (2004) On homogeneous least-squares problems and the inconsistency introduced by mis-constraining. Computational statistics & data analysis 47(3):455–465

Appendix B
Solutions to the exercises

The students will find the solution really interesting if they have made an honest effort, and have the consciousness of having done well. Then they are eager to see what else they could accomplish with that effort, and how they could do equally well another time.

Pólya (1957)

Chapter 1

1.1 (Geometric interpretation of rank-1 approximation). In both problems (lra_R) and (lra_P) the cost function is the sum of the squared distances from the data points d_j to their approximations $\widehat{d}_j$

$$\|D-\widehat{D}\|_{\mathrm{F}}^2 = \sum_{j=1}^{N} \|d_j - \widehat{d}_j\|_2^2.$$

The rank-1 constraint of

$$\widehat{D} = \begin{bmatrix} \widehat{d}_1 & \cdots & \widehat{d}_N \end{bmatrix},$$

is equivalent to the constraint that the approximations $\widehat{d}_j$ lie on a line $\mathscr{B}$ passing through the origin. In (lra_R), $\mathscr{B} = \ker(R)$. In (lra_P), $\mathscr{B} = \text{image}(P)$. By the orthogonality principle, $\widehat{d}_j$ must be the orthogonal projection of d_j on $\mathscr{B}$, so that $\|d-\widehat{d}_j\|_2^2$ is the squared orthogonal distance from d_j to $\mathscr{B}$. Therefore, the rank-1 approximation problems (lra_R) and (lra_P) minimize the sum of the squared orthogonal distances from the data points to the fitting line $\mathscr{B}$ over all lines passing through the origin.

1.2 (Quadratically constrained problem, equivalent to rank-1 approximation). Consider the rank-1 approximation problem (lra_P) and observe that for a fixed parameter $P \in \mathbb{R}^{2\times 1}$, it becomes a least squares problem in the parameter $L \in \mathbb{R}^{1\times N}$

$$\text{minimize} \quad \text{over } L \quad \|D-PL\|_{\mathrm{F}}^2.$$

Assuming that $P \neq 0$, the solution is unique and is given by

$$L^* = (P^\top P)^{-1} P^\top D.$$

Then the minimum $M(P) = \|D-PL^*\|_{\mathrm{F}}^2$ is given by

$$M(P) = \text{trace}\Big(D^\top \big(I_2 - P(P^\top P)^{-1}P^\top\big)D\Big).$$

The function M, however, depends only on the direction of P, *i.e.*,

$$M(P) = M(\alpha P), \quad \text{for all } \alpha \neq 0.$$

Therefore, without loss of generality we can assume that $\|P\|_2 = 1$. This argument and the derivation of M show that problem (lra'_P) is equivalent to problem (lra_P). All solutions of (lra_P) are obtained from a solution P'^* of (lra'_P) by multiplication with a nonzero scalar and vice verse a solution P^* of (lra_P) is reduced to a solution of (lra'_P) by normalization $P^*/\|P^*\|$. A solution to (lra'_P), however, is still not unique because if P'^* is a solution, so is $-P'^*$.

1.3 (Line fitting by rank-1 approximation). The set of vectors $P \in \mathbb{R}^2$, such that $P^\top P = 1$, is parametrized by

$$P(\theta) = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}, \quad \text{where} \quad \theta \in [0, 2\pi).$$

The graph of $M\big(P(\theta)\big)$ (see Figure B.1) shows that the global minimum

$$M\big(P(\theta^{*,1})\big) = M\big(P(\theta^{*,2})\big) = 20$$

is achieved for $\theta^{*,1} = \pi/2$ and $\theta^{*,2} = 3\pi/2$. The corresponding model parameters

$$P^{*,1} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{and} \quad P^{*,2} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \qquad (*)$$

define a vertical line passing through the origin.

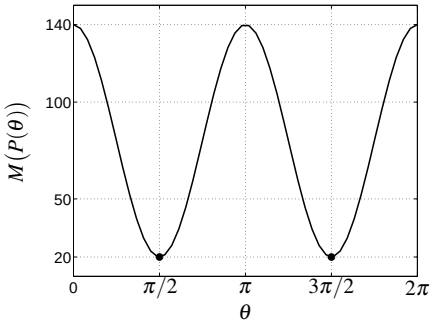


Fig. B.1: The cost function of the rank-1 approximation problem (lra'_P) in Problem 1.3 has two global minima, corresponding to $\theta = \pi/2$ and $\theta = 3\pi/2$.

1.4 (Analytic solution of a rank-1 approximation problem).

$$\begin{aligned} M(P) &= \operatorname{trace}\big(D^\top (I_2 - PP^\top) D\big) \\ &= \operatorname{trace}\big((I_2 - PP^\top) DD^\top\big) \\ &= \cdots \text{ substituting the data and} \\ &\qquad \text{using the constraint } P^\top P = p_1^2 + p_2^2 = 1 \ \cdots \\ &= \operatorname{trace}\left(\begin{bmatrix} p_2^2 & -p_1 p_2 \\ -p_1 p_2 & p_1^2 \end{bmatrix} \begin{bmatrix} 20 & 0 \\ 0 & 140 \end{bmatrix}\right) \\ &= 20 p_2^2 + 140 p_1^2 = 20 \sin^2(\theta) + 140 \cos^2(\theta). \end{aligned}$$

From the analytic expression of M it is easy to see that

$$20 \leq M\big(P(\theta)\big) \leq 140,$$

and the minimum is achieved for $\sin(\theta) = \pm 1$, *cf.*, $(*)$ in Problem 1.3.

1.5 (Literate program for Sylvester matrix construction). The Sylvester matrix constructor function `sylv` has two compulsory inputs—the polynomials p and q—and one output—the Sylvester matrix ($\mathscr{R}$), represented in the code by the variable `S`:

235a ⟨*Sylvester matrix constructor* 235a⟩≡ 235b▷
```
function S = sylv(p, q)
```
Defines:
`sylv`, used in chunk 235d.

After determining the degrees $n = \deg(p)$ and $m = \deg(q)$ from the lengths of `p` and `q`, the matrix `S` is initializing with the $(n+m) \times (n+m)$ zero matrix:

235b ⟨*Sylvester matrix constructor* 235a⟩+≡ ◁235a 235c▷
```
n = length(p) - 1; m = length(q) - 1; S = zeros(n + m);
```

The elements of p and q are assigned to the corresponding elements of `S`:

235c ⟨*Sylvester matrix constructor* 235a⟩+≡ ◁235b
```
for i = 1:m, S(i:i + n, i    ) = p(:); end
for i = 1:n, S(i:i + m, m + i) = q(:); end
```

Using `sylv`, we find that the greatest common factor's degree in the example

235d ⟨*GCD example* 235d⟩≡
```
p = [1 3 5 3]; q = [3 5 3 1];
d = 6 - rank(sylv(p, q))
```
Uses `sylv` 235a.

is $\mathtt{d} = 1$. Indeed,

$$p(z) = (1+z)(3z^2+2z+1) \quad \text{and} \quad q(z) = (1+z)(z^2+2z+3).$$

Chapter 2

2.1 (Relations among $\operatorname{rank}(P)$, $\operatorname{rank}(R)$, and $\dim(\mathscr{B})$, for a linear static model $\mathscr{B}$).

1. From the definition of an image representation $\mathscr{B} = \operatorname{image}(P)$, it follows that the columns of P span $\mathscr{B}$. By definition, $\dim(\mathscr{B})$ is the number of linearly independent vectors that span $\mathscr{B}$. Also, by definition, $\operatorname{rank}(P)$ is the number of linearly independent columns of P. Therefore, $\operatorname{rank}(P) = \dim(\mathscr{B})$.
2. From the definition of a kernel representation $\mathscr{B} = \ker(R)$, it follows that the rows of R span the orthogonal complement $\mathscr{B}^\perp$ of $\mathscr{B}$, defined as

$$\mathscr{B}^\perp := \{ z \in \mathscr{U} \mid z \perp w \text{ for all } w \in \mathscr{B} \}.$$

We have that

$$\operatorname{image}(R^\top) = \mathscr{B}^\perp.$$

Also from part 1 of the problem, we have that $\operatorname{rank}(R) = \dim(\mathscr{B}^\perp)$. Since,

$$\dim(\mathscr{B}) + \dim(\mathscr{B}^\perp) = \mathtt{q},$$

we obtain the relation $\operatorname{rank}(R) = \mathtt{q} - \dim(\mathscr{B})$.

2.2 ($\mathscr{B}_1 \stackrel{?}{=} \mathscr{B}_2$). Let $\mathscr{B}_1 = \ker(R^1) = \operatorname{image}(P^1)$ and $\mathscr{B}_2 = \ker(R^2) = \operatorname{image}(P^2)$. The kernel parameters R^1 and R^2 as well as the image parameters P^1 and P^2 are not unique, so that they can not be compared element-wise. The simplest way to check whether $\mathscr{B}_1 = \mathscr{B}_2$, is to derive the input/output representations $\mathscr{B}_1 = \mathscr{B}_{\mathrm{i/o}}(X^1, \Pi)$ and $\mathscr{B}_2 = \mathscr{B}_{\mathrm{i/o}}(X^2, \Pi)$, with a permutation matrix Π for which both representations exist, and compare the corresponding parameters X^1 and X^2.

A MATLAB function that checks if two models specified by image representations are equal is:

236 ⟨p1_eq_p2 236⟩≡
```
function ans = p1_eq_p2(p1, p2, tol)
if ~exist('tol') || isempty(tol), tol = 1e-12; end
ans = norm(p2x(p1) - p2x(p2)) < tol
```

2.3 (Input/output partitions that are not possible). Let $\mathscr{B}$ be a linear static model with `q` variables and `m` inputs. In order to formulate the solution, first, we to introduce some notation. Consider a set of indeces $\mathscr{I} \subseteq \{1, \ldots, \mathtt{q}\}$ and denote by $\operatorname{card}(\mathscr{I})$ the number of elements of $\mathscr{I}$. The matrix $\Pi_{\mathscr{I}} := I_{\mathscr{I}}$, which consists of the rows of the $\mathtt{q} \times \mathtt{q}$ identity matrix with indeces in $\mathscr{I}$, selects the elements of $d \in \mathbb{R}^{\mathtt{q}}$, which indeces are in $\mathscr{I}$. It is a projection from $\mathbb{R}^{\mathtt{q}}$ onto $\mathbb{R}^{\operatorname{card}(\mathscr{I})}$. Acting on a set $\mathscr{B}$, $\Pi_{\mathscr{I}}$ projects all vectors in the set, *i.e.*, $\Pi_{\mathscr{I}}\mathscr{B} := \{ \Pi_{\mathscr{I}} d \mid d \in \mathscr{B} \}$.

By definition, a set of variables $d_{\mathscr{I}}$ can be inputs in an input/output representation of the model $\mathscr{B}$ if they are free, *i.e.*,

$$\Pi_{\mathscr{I}}\mathscr{B} = \mathbb{R}^{\operatorname{card}(\mathscr{I})}.$$

Note that for a linear static model, $\Pi_i\mathscr{B}$ is either $\{0\}$ or $\mathbb{R}$. Which is the case ($\{0\}$ or $\mathbb{R}$) can be checked from a given representation of the model. For example, let $\mathscr{B} = \text{image}(P)$, then $\Pi_i\mathscr{B} = \mathbb{R}$ if and only if the ith row of P is zero.

Example B.1. The linear static model with three variables and one input

$$\mathscr{B} = \text{image}\left(\begin{bmatrix}1\\0\\0\end{bmatrix}\right) = \ker\left(\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\right)$$

has only two possible input/output partitions: $u = w_1$, $y = \left[\begin{smallmatrix}w_2\\ w_3\end{smallmatrix}\right]$ and $u = w_1$, $y = \left[\begin{smallmatrix}w_3\\ w_2\end{smallmatrix}\right]$. Indeed, the function `r2io`

237a ⟨*Test* `r2io` 237a⟩≡
```
r2io([0 0 1; 0 1 0])
```
computes the input/output partitionings from the parameter R in a kernel representation of the model

```
ans =

     1     2     3
     1     3     2
```

2.4 (Initial conditions specification by trajectory). First, we consider the map $w_\text{p} \mapsto x(0)$. For a trajectory $w_\text{p} = \left[\begin{smallmatrix}u_\text{p}\\ y_\text{p}\end{smallmatrix}\right]$ of the system $\mathscr{B}$, we have that

$$y_\text{p} = \underbrace{\begin{bmatrix}C\\ CA\\ \vdots\\ CA^{\ell-1}\end{bmatrix}}_{\mathscr{O}} x(-\ell+1) + \underbrace{\begin{bmatrix}H(0) & & & \\ H(1) & H(0) & & \\ \vdots & \ddots & \ddots & \\ H(\ell-1) & \cdots & H(1) & H(0)\end{bmatrix}}_{\mathscr{T}} u_\text{p} \qquad (*)$$

for some $x(-\ell+1)$. Since the state space representation is minimal, $\mathscr{O}$ is full column rank. Therefore, a solution $x(-\ell+1)$ of $(*)$ is unique and the unique $x(0)$ is

$$x(0) = A^{\ell-1}x(-\ell+1) + \underbrace{\begin{bmatrix}A^{\ell-2}B & \cdots & BA^0 & 0\end{bmatrix}}_{\mathscr{C}} u_\text{p}.$$

The resulting function is:

237b ⟨`wp2x0` 237b⟩≡
```
function x0 = wp2x0(wp, sys)
[p, m, n] = size(sys); ell = ceil(n / p);
xini  = obsv(sys) \ (wp(:, 2) - lsim(sys, wp(:, 1)));
x0 = sys.a ^ (ell - 1) * xini + obsv(sys) * wp(:, 1);
```

Next, we consider the map $w_\text{p} \mapsto x(0)$. In order to set $x(0)$, we append a past trajectory w_p to w_f, *i.e.*, we consider the extended trajectory $w = w_\text{p} \wedge w_\text{f}$. The past trajectory w_p is determined as a solution of the system

$$x(0) = \begin{bmatrix}\mathscr{C} - A^{\ell-1}\mathscr{O}^+\mathscr{T} & A^{\ell-1}\mathscr{O}^+\end{bmatrix}\begin{bmatrix}u_\text{p}\\ y_\text{p}\end{bmatrix}.$$

The resulting function is:

238a ⟨`x02wp` 238a⟩≡
```
function wp = x02wp(x0, sys)
[p, m, n] = size(sys); ell = ceil(n / p);
⟨construct 𝒞 238b⟩
⟨construct 𝒯 238c⟩
O  = obsv(sys);
AO = sys.a ^ (ell -1) * pinv(O);
wp = pinv([C - AO * T , AO]) * x0;
wp = reshape(wp, ell, 2);
```
where

238b ⟨*construct* $\mathscr{C}$ 238b⟩≡ (238a)
```
C = [sys.b zeros(ell, 1)];
for i = 1:(ell - 2)
  C = [sys.a * C(:, 1) C];
end
```
and

238c ⟨*construct* $\mathscr{T}$ 238c⟩≡ (238a)
```
h = impulse(sys, ell - 1);
T = toeplitz(h, [h(1) zeros(1, ell - 1)]);
```

Finally, we test the functions `wp2x0` and `x02wp` on a simulation example.

238d ⟨*simulate data* 238d⟩≡ (238e)
```
n = 2; sys = drss(n);
T = 20; u = rand(T, 1); xini = rand(n, 1);
[y, t, x] = lsim(sys, u, [], xini); w = [u y];
```

The results of the computation

238e ⟨*test* 238e⟩≡
```
⟨simulate data 238d⟩
wp = w(end - n + 1:end, :); x0 = x(end, :)';
wp2x0(wp, sys) - x0                % == 0
wp2x0(x02wp(x0, sys), sys) - x0 % == 0
```
are of the order of the machine precision, which empirically confirms that the derivation is correct and the functions `wp2x0`, `x02wp` are correctly implemented.

2.5 $(w \overset{?}{\in} \mathscr{B} = \ker\left(R(\sigma)\right))$**.** Consider a trajectory $w \in \ker\left(R(\sigma)\right)$. We have that

$$R(\sigma)w = 0 \iff R_0w(t) + R_1w(t+1) + \cdots + R_\ell w(t+\ell) = 0, \text{ for } t = 1, \ldots, T-\ell$$
$$\iff \underbrace{\begin{bmatrix}R_0 & R_1 & \cdots & R_\ell & & \\ & R_0 & R_1 & \cdots & R_\ell & \\ & & \ddots & \ddots & & \ddots \\ & & & R_0 & R_1 & \cdots & R_\ell\end{bmatrix}}_{\mathscr{M}_{T-\ell}(R) \in \mathbb{R}^{\mathsf{p}(T-\ell)\times \mathsf{q}T}} \underbrace{\begin{bmatrix}w(1)\\ w(2)\\ \vdots\\ w(T)\end{bmatrix}}_{\text{vec}(w)} = 0.$$

The last equation holds if and only if $w \in \ker\big(R(\sigma)\big)$. Therefore, we can check if $w \in \ker\big(R(\sigma)\big)$ by checking $\|\mathscr{M}_{T-\ell}(R)\,\mathrm{vec}(w)\| < \varepsilon$, where ε is a tolerance.

An alternative way for checking if $w \in \ker\big(R(\sigma)\big)$ is derived from

$$\begin{aligned} w \in \ker\big(R(\sigma)\big) &\iff \mathscr{M}_{T-\ell}(R)\,\mathrm{vec}(w) = 0 \\ &\iff R\mathscr{H}_{\ell+1}(w) = 0, \end{aligned}$$

where

$$\underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix}}_{R} \underbrace{\begin{bmatrix} w(1) & w(2) & \cdots & w(T-\ell) \\ w(2) & w(3) & \cdots & \\ \vdots & \vdots & & \vdots \\ w(\ell+1) & w(\ell+2) & \cdots & w(T) \end{bmatrix}}_{\mathscr{H}_{\ell+1}(w)\in\mathbb{R}^{q(\ell+1)\times(T-\ell)}} = 0.$$

In this case the numerical test is $\|R\mathscr{H}_{\ell+1}(w)\| < \varepsilon$.

The second method is implemented in in the function `w_in_ker`.

239a ⟨`w_in_ker` 239a⟩≡
```
function a = w_in_ker(w, r, ell)
a = norm(r * blkhank(w, ell + 1)) < 1e-8;
```
Uses `blkhank` 26a.

Applying it on the data

239b ⟨*Test trajectory* 239b⟩≡
```
w = [0 0 0 0; 1 1 1 1];
r = [1 -1 -1 1]; ell = 1;
w_in_ker(w, r, 1)
```

it gives positive answer, so that w is a trajectory of $\mathscr{B}$.

2.6 **($\ker\big(R(\sigma)\big) \leftrightarrow \mathscr{B}_{\mathrm{i/o}}(p,q)$).** In the single-input single-output case, with input/output partitioning $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ and minimal representations

$$\mathscr{B} = \ker\big(R(\sigma)\big) = \mathscr{B}_{\mathrm{i/o}}(p,q),$$

the link between $R(z)$ and $\big(p(z), q(z)\big)$ is given by

$$R(z) = \begin{bmatrix} -q(z) & p(z) \end{bmatrix}. \qquad ((p,q) \mapsto R)$$

Indeed,

$$R(\sigma)w = \begin{bmatrix} -q(\sigma) & p(\sigma) \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} \iff q(\sigma)u = p(\sigma)y.$$

MATLAB implementation: We represent $R(z)$ by a vector and $\mathscr{B}_{\mathrm{i/o}}(p,q)$ by a `tf` object (from the Control System Toolbox of MATLAB). Our convention of representing a polynomial is ordering the coefficients in *ascending* degrees, *i.e.*,

$$R(z) = R_0 + R_1 z + \cdots + R_\ell z^\ell \quad \leftrightarrow \quad R = \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix}.$$

MATLAB and the Control System Toolbox, on the other hand, use the convention of ordering the coefficients in *descending* degrees. This requires interchanging the order when making the transitions from R to (p,q) and back.

240a ⟨*(TF)* $\mapsto$ *R(z)* 240a⟩≡
```
[q, p] = tfdata(tf(sys), 'v'); R = zeros(1, 2 * length(p));
R(1:2:end) = - fliplr([q zeros(length(p) - length(q))]);
R(2:2:end) =   fliplr(p);
```

and

240b ⟨*R(z)* $\mapsto$ *(TF)* 240b⟩≡ (240e)
```
q = - fliplr(R(1:2:end));
p = fliplr(R(2:2:end));
sys = tf(q, p, -1);
```

2.7 **(image$\big(P(\sigma)\big) \leftrightarrow \mathscr{B}_{\mathrm{i/o}}(p,q)$).** Consider the single-input single-output case, with an input/output partitioning $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$ and minimal representations

$$\mathscr{B} = \mathrm{image}\big(P(\sigma)\big) = \mathscr{B}_{\mathrm{i/o}}(p,q).$$

From ($(p,q) \mapsto R$) and the identity $R(z)P(z) = 0$, we have

$$P(z) = \begin{bmatrix} p(z) \\ q(z) \end{bmatrix}. \qquad ((p,q) \mapsto P)$$

Therefore, the transition $P \mapsto (p,q)$ is $p(z) = P_1(z)$ and $q(z) = P_2(z)$.

MATLAB implementation:

240c ⟨*(TF)* $\mapsto$ *P(z)* 240c⟩≡ (118a 146a 147b 240e)
```
[q, p] = tfdata(tf(sys), 'v'); P = zeros(2, length(p));
P(1, :) = fliplr(p);
P(2, :) = fliplr([q zeros(length(p) - length(q))]);
```

and

240d ⟨*P(z)* $\mapsto$ *(TF)* 240d⟩≡ (119)
```
p = fliplr(P(1, :)); q = fliplr(P(2, :)); sys = tf(q, p, -1);
```

An indirect way of implementing the transition $R(z) \mapsto P(z)$ is two go via $\mathscr{B}_{\mathrm{i/o}}(p,q)$:

240e ⟨*R(z)* $\mapsto$ *P(z)* 240e⟩≡ (118c)
```
⟨R(z) ↦ (TF) 240b⟩
⟨(TF) ↦ P(z) 240c⟩
```

Chapter 3

3.1 **(The most powerful unfalsified model in $\mathscr{L}_0$).** By definition, a model $\mathscr{B}$ is exact for data $\mathscr{D}$ if it contains the data, *i.e.*, $\mathscr{D} \subset \mathscr{B}$. Since the model is linear, it must contain also all linear combinations of the data, *i.e.*, $\mathrm{span}(\mathscr{D}) \subset \mathscr{B}$. Among all exact linear static models for $\mathscr{D}$, the most powerful one is the smallest one, so that

$$\mathscr{B}_{\mathrm{mpum}}(\mathscr{D}) = \mathrm{span}(\mathscr{D}). \qquad (*)$$

Next, we consider the question of computing $\mathscr{B}_{\text{mpum}}(\mathscr{D})$. Since $\mathscr{B}_{\text{mpum}}(\mathscr{D})$ is a set. First, we need to choose how to represent it. Then, the question of computing $\mathscr{B}_{\text{mpum}}(\mathscr{D})$ becomes a question of computing the corresponding model parameter.

Kernel, image, and input/output representations of $\mathscr{B}_{\text{mpum}}(\mathscr{D})$ are candidates for defining parameter estimation problems equivalent to the original problem of computing $\mathscr{B}_{\text{mpum}}(\mathscr{D})$. Consider an image representation image(P) of $\mathscr{B}_{\text{mpum}}(\mathscr{D})$.

From $(*)$ it follows that the columns of P span the image of the data matrix $D = \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix}$. We can impose the stronger requirement that the columns of P are a basis of image(D). The problem of computing $\mathscr{B}_{\text{mpum}}(\mathscr{D})$ from $\mathscr{D}$, then becomes the standard linear algebra problem of computing a basis for the image of a matrix. This problem can be solved by existing algorithms. Therefore, the problem of computing $\mathscr{B}_{\text{mpum}}(\mathscr{D})$ in $\mathscr{L}_0$ is solved by reduction to an already solved problem.

Optional continuation of Exercise 3.1

> Can you derive the result differently? We prefer, of course, a short and intuitive argument to a long and heavy one: Can you see it at a glance?
>
> Pólya (1957)

1. Implement and test the method for computing $\mathscr{B}_{\text{mpum}}(\mathscr{D}) = \text{image}(P)$.
2. Explore the approaches based on the kernel and input/output representations.
3. Compare the different solution methods with respect to the following criteria.

- **Theory:** which method was easiest/shortest to derive?
- **Insight:** which method did you find most insightful in understating the map

$$\mathscr{D} \mapsto \mathscr{B}_{\text{mpum}}(\mathscr{D})?$$

- **Algorithms:** what are the algorithms being used and which one is fastest?

3.2 (The most powerful unfalsified model in $\mathscr{L}^1_{0,\ell}$). We are looking for an autonomous linear time-invariant model $\mathscr{B} = \ker\big(p(\sigma)\big)$ with lag at most ℓ, such that $y_{\text{d}} \in \mathscr{B}$. Therefore, the system of linear equations

$$p_0 y_{\text{d}}(t) + p_1 y_{\text{d}}(t+1) + \cdots + p_\ell y_{\text{d}}(t+\ell) = 0, \quad \text{for } t = 1, \ldots, T-\ell.$$

must hold true. Written in a matrix form the system of equations is

$$\underbrace{\begin{bmatrix} p_0 & p_1 & \cdots & p_\ell \end{bmatrix}}_{p} \mathscr{H}_{\ell+1}(y_{\text{d}}) = 0. \tag{$*$}$$

Since $p \neq 0$, we obtain an equivalent rank condition $\operatorname{rank}\big(\mathscr{H}_{\ell+1}(y_{\text{d}})\big) \leq \ell$ that depends on the data y_{d} and the lag ℓ only and is verifiable by existing methods.

3.3 ($y_{\text{d}} \mapsto \mathscr{B}_{\text{mpum}}(y_{\text{d}}) = \ker\big(p(\sigma)\big)$). Equation $(*)$ suggests a method for computing the parameter vector p from the left kernel of the matrix $\mathscr{H}_{\ell+1}(y_{\text{d}})$, *i.e.*, p can

be computed by constructing the Hankel matrix $\mathscr{H}_{\ell+1}(y_{\text{d}})$ and computing its left kernel. Then, any vector in the left kernel defines an exact model for y_{d}.

So far we assumed that ℓ is given. In case when the ℓ is unknown, we can detect the smallest ℓ, for which there is an exact model by doing the rank test for $\ell = 1, 2, \ldots$

242 ⟨*Finding the lag* 242⟩≡

```
for ell = 1:ell_max
   if (rank(H(y, ell + 1)) == ell)
      break
   end
end
```

In the example, $\ell = 3$ and p can be chosen as $\begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix}$.

Note that the method, described above, for the computation of a kernel representation $\ker\big(p(\sigma)\big) = \mathscr{B}_{\text{mpum}}(y_{\text{d}})$ generalizes trivially to the model class $\mathscr{L}^{\mathtt{q}}_{\mathtt{m},\ell}$, *i.e.*, general multivariable linear time-invariant systems. Indeed, the only difference with the scalar autonomous case is that the kernel of the Hankel matrix $\mathscr{H}_{\ell+1}(w_{\text{d}})$, constructed from the data w_{d}, has dimension $\mathtt{p}$. A basis for the kernel contains the parameters $R_0, R_1, \ldots, R_\ell$ of a (in general nonminimal) kernel representation

$$\ker\big(R(\sigma)\big) = \mathscr{B}_{\text{mpum}}(w_{\text{d}}).$$

3.4 ($y_{\text{d}} \mapsto \mathscr{B}_{\text{mpum}}(y_{\text{d}}) = \mathscr{B}(A,c)$). Finding the most powerful unfalsified model in the model class $\mathscr{L}^1_{0,\ell}$ is equivalent to the realization problem. Indeed, the impulse response of $\mathscr{B}_{\text{i/s/o}}(A, x_{\text{ini}}, c, 0)$ and the output of $\mathscr{B}_{\text{ss}}(A,c)$ due to initial condition $x(1) = x_{\text{ini}}$ are identical. Therefore, realization algorithms, *e.g.*, Kung's method (`h2ss`), can be used for computing a state-space representation of $\mathscr{B}_{\text{mpum}}(y_{\text{d}})$.

3.5 (Data-driven step response estimation). A data-driven step response estimation method can be obtained mutatis mutandis from the impulse response estimation method, see Section 3.2. Under the assumptions of Lemma 3.7, there is g_k such that

$$w_k = \mathscr{H}_{\ell+t}(w_{\text{d}}) g_k, \quad \text{for } k = 1, \ldots, \mathtt{m},$$

where ℓ is the lag of the system, t is the number of to-be-estimated samples of the step response, and $w_k = \left[\begin{smallmatrix} u_k \\ y_k \end{smallmatrix}\right]$, with

$$u_k = \big(\underbrace{0, \ldots, 0}_{\ell}, \underbrace{e_k, \ldots, e_k}_{t}\big) \quad \text{and} \quad y_k = \big(\underbrace{0, \ldots, 0}_{\ell}, \underbrace{s_k(1), \ldots, s_k(t)}_{t}\big).$$

By construction, the $s_k(1), \ldots, s_k(t)$'s are the first t samples of the step response due to the zero initial conditions (see Exercise 2.4) and step on the kth input.

Reordering the equations, we obtain the following relations

$$\begin{aligned} \mathbf{H}_{\text{f},y} g_k &= s_k, \\ \begin{bmatrix} \mathbf{H}_{\text{p}} \\ \mathbf{H}_{\text{f},u} \end{bmatrix} g_k &= \begin{bmatrix} 0_{\mathtt{q}\ell \times 1} \\ \mathbf{1}_t \otimes e_k \end{bmatrix}, \quad \text{for } k = 1, \ldots, \mathtt{m}. \end{aligned}$$

Defining $G := \begin{bmatrix} g_1 & \cdots & g_{\mathtt{m}} \end{bmatrix}$ and $S := \begin{bmatrix} s_1 & \cdots & s_{\mathtt{m}} \end{bmatrix}$,

$$\mathbf{H}_{\mathrm{f},y} G = S,$$
$$\begin{bmatrix} \mathbf{H}_{\mathrm{p}} \\ \mathbf{H}_{\mathrm{f},u} \end{bmatrix} G = \begin{bmatrix} 0_{\mathtt{q}\ell\times\mathtt{m}} \\ \mathbf{1}_t \otimes I \end{bmatrix}.$$

By construction the second equation has a solution and any solution G satisfies the first equation. Taking the least-norm solution, we obtain the following closed-form expression for the first t samples of the impulse response

$$S = \mathbf{H}_{\mathrm{f},y} \begin{bmatrix} \mathbf{H}_{\mathrm{p}} \\ \mathbf{H}_{\mathrm{f},u} \end{bmatrix}^{+} \begin{bmatrix} 0_{\mathtt{q}\ell\times\mathtt{m}} \\ \mathbf{1}_t \otimes I \end{bmatrix}.$$

This gives us a data-driven algorithm for step response estimation.

For implementation and testing, see the solution of Exercise 6.2.

Chapter 4

4.1 (Distance from a data point to a linear model).

1. The distance computation is equivalent to the standard least squares problem

$$\operatorname{dist}(d,\mathscr{B}) := \min \|d-\widehat{d}\|_2 \quad \text{subject to} \quad \widehat{d} = P\ell.$$

Since P is full column rank (minimal representation), the best approximation is

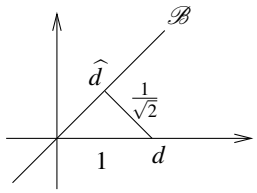
$$\widehat{d}^* = P(P^\top P)^{-1}P^\top d =: \Pi_P d \qquad (d^*)$$

and the distance of d to $\mathscr{B}$ is

$$\operatorname{dist}(d,\mathscr{B}) = \|d-\widehat{d}^*\|_2 = \sqrt{d^\top(I-\Pi_P)d}. \qquad (\operatorname{dist}_P)$$

Substituting $d = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$ and $P = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$ in (dist_P), we have

$$\operatorname{dist}\left(\begin{bmatrix}1\\0\end{bmatrix}, \operatorname{image}\left(\begin{bmatrix}1\\1\end{bmatrix}\right)\right) = \cdots = 1/\sqrt{2}.$$



2. Using a minimal kernel representation $\ker(R)$ of the model $\mathscr{B}$, the distance computation problem is equivalent to the problem

$$\operatorname{dist}(d,\mathscr{B}) := \min_{\widehat{d}} \|d-\widehat{d}\|_2 \quad \text{subject to} \quad R\widehat{d} = 0.$$

This is not a standard least squares problem. A change of variables $\Delta d := d - \widehat{d}$, however, leads to an equivalent standard least norm problem

$$\operatorname{dist}(d,\mathscr{B}) := \min_{\widehat{d}} \|\Delta d\|_2 \quad \text{subject to} \quad R\Delta d = Rd.$$

Since R is full row rank (minimal representation),

$$\Delta d^* = R^\top(RR^\top)^{-1}Rd = \Pi_{R^\top} d$$

and

$$\operatorname{dist}(d,\mathscr{B}) = \|\Delta d^*\|_2 = \sqrt{d^\top \Pi_{R^\top} d}. \qquad (\operatorname{dist}_R)$$

3. As shown in part 1, $\widehat{d}^*$ is unique (and can be computed by, *e.g.*, (d^*) and (dist_P)).
4. A vector Δd is orthogonal to the model $\mathscr{B}$ if and only if Δd is orthogonal to all vectors in $\mathscr{B}$. Using (d^*) and the basis P for $\mathscr{B}$, we have

$$\Delta d^{*\top} P = (d-\widehat{d}^*)^\top P = d^\top(I-\Pi_P)P = 0,$$

which shows that is Δd^* is orthogonal to $\mathscr{B}$.
The converse statement "$\Delta d = d - \widehat{d}$ being orthogonal to $\mathscr{B}$ implies that $\widehat{d}$ is the closest point in $\mathscr{B}$ to d" is also true. It completes the proof of what is known as the *orthogonality principle*—$\widehat{d}$ is an optimal approximation of a point d in a model $\mathscr{B}$ if and only if the approximation error $d-\widehat{d}$ is orthogonal to $\mathscr{B}$.

4.2 (Distance from a data point to an affine model).

1. The problem of computing $\operatorname{dist}(d,\mathscr{B})$ reduces to an equivalent problem of computing the distance of a point to a subspace by the change of variables

$$d' := d - c.$$

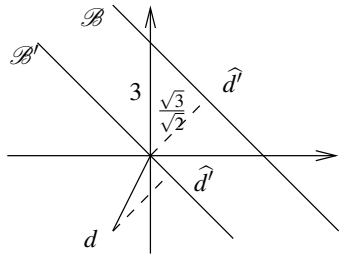
We have

$$\operatorname{dist}(d,\mathscr{B}) = \min_{\widehat{d}\in\mathscr{B}} \|d-\widehat{d}\|_2 = \min_{\widehat{d}'\in\mathscr{B}'} \|d'-\widehat{d}'\|_2 = \operatorname{dist}(d',\mathscr{B}').$$

2. Using the change of variables argument, we have

$$\operatorname{dist}\left(\begin{bmatrix}0\\0\end{bmatrix}, \ker(\begin{bmatrix}1 & 1\end{bmatrix}) + \begin{bmatrix}1\\2\end{bmatrix}\right) = \operatorname{dist}\left(-\begin{bmatrix}1\\2\end{bmatrix}, \ker(\begin{bmatrix}1 & 1\end{bmatrix})\right).$$

Then using (dist_R) we have $\operatorname{dist}\left(-\left[\begin{smallmatrix}1\\2\end{smallmatrix}\right], \ker(\begin{bmatrix}1 & 1\end{bmatrix})\right) = \cdots = \sqrt{9/2}$.



4.3 (Two-sided weighted low-rank approximation). Define

$$D_{\mathrm{m}} := \sqrt{W_{\mathrm{l}}} D \sqrt{W_{\mathrm{r}}} \quad \text{and} \quad \widehat{D}_{\mathrm{m}} := \sqrt{W_{\mathrm{l}}} \widehat{D} \sqrt{W_{\mathrm{r}}}.$$

Since W_{l} and W_{r} are nonsingular,

$$\operatorname{rank}(\widehat{D}) = \operatorname{rank}(\widehat{D}_{\mathrm{m}}).$$

Then, from (WLRA2), we obtain the equivalent problem

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{D}_{\mathrm{m}} \quad \|D_{\mathrm{m}} - \widehat{D}_{\mathrm{m}}\|_{\mathrm{F}} \\ &\text{subject to} && \operatorname{rank}(\widehat{D}_{\mathrm{m}}) \leq \mathtt{m}, \end{aligned} \tag{WLRA2'}$$

which is an unweighted low-rank approximation.

MATLAB implementation:

245a ⟨*Two-sided weighted low-rank approximation* 245a⟩≡
```
function [r, p, dh] = glra(d, m, wl, wr)
swl = sqrtm(wl);
swr = sqrtm(wr);
d = swl * d  * swr';
[r, p, dh] = lra(d, m);
r = r * swl;
p = pinv(swl) * p;
dh = swl \ dh / swr';
```
Uses lra 105a.

4.4 (A simple method for approximate system identification). A trivial way of obtaining an approximate identification method from a corresponding exact identification method is to replace exact computations by corresponding approximate computations. In the case of the method described in the solution of Exercise 3.3, the computation is for a basis of the left kernel of the Hankel matrix $\mathscr{H}_{\ell+1}(w_{\mathrm{d}})$. An approximate version of this computation is unstructured low-rank approximation (lra). This gives us the following approximate identification method:

245b ⟨w2r 245b⟩≡
```
function R = w2r(w, m, ell)
⟨reshape w and define q, T 27a⟩
R = lra(blkhank(w, ell + 1), q * ell + m);
```
Uses blkhank 26a and lra 105a.

4.5 (Misfit computation using state space representation). A finite sequence $w = \big(w(1), \dots, w(T)\big)$, with an input/output partition $w = \left[\begin{smallmatrix} u \\ y \end{smallmatrix}\right]$, is a trajectory of $\mathscr{B} = \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$, if and only if there is an initial condition $x_{\mathrm{ini}} \in \mathbb{R}^{\mathtt{n}}$, such that

$$y = \underbrace{\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{T-1} \end{bmatrix}}_{\mathscr{O}} x_{\mathrm{ini}} + \underbrace{\begin{bmatrix} D & & & & \\ CB & D & & & \\ CAB & CB & D & & \\ \vdots & \ddots & \ddots & \ddots & \\ CA^{T-1}B & \cdots & CAB & CB & D \end{bmatrix}}_{\mathscr{T}} u = \mathscr{O} x_{\mathrm{ini}} + \mathscr{T} u. \qquad (\mathscr{O}, \mathscr{T})$$

Substituting the last expression for y in the definition (misfit $\mathscr{L}_{\mathtt{m},\ell}$) of the misfit, we obtain an ordinary least squares problem

$$\min_{\widehat{x}_{\mathrm{ini}}, \widehat{u}} \left\| \begin{bmatrix} u_{\mathrm{d}} \\ y_{\mathrm{d}} \end{bmatrix} - \begin{bmatrix} 0 & I \\ \mathscr{O} & \mathscr{T} \end{bmatrix} \begin{bmatrix} \widehat{x}_{\mathrm{ini}} \\ \widehat{u} \end{bmatrix} \right\|.$$

The solution gives us the initial condition $\widehat{x}_{\mathrm{ini}}$ and the input $\widehat{u}$ of the optimal approximation $\widehat{w}$ of w_{d} in $\mathscr{B}$. The output $\widehat{y}$ is $\mathscr{O}\widehat{x}_{\mathrm{ini}} + \mathscr{T}\widehat{u}$. Knowing $\widehat{w}$, we can evaluate the misfit as

$$\operatorname{dist}\big(w_{\mathrm{d}}, \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)\big)\big) = \|w_{\mathrm{d}} - \widehat{w}\|.$$

Note B.2 (Fast computation of the misfit using errors-in-varibales Kalman filter). Solving the least squares problem by general purpose methods leads to cubic in the number of samples T computational cost. The matrices $\mathscr{O}$ and $\mathscr{T}$, however, have structured that allows fast computations. A recursive algorithm, known as the *errors-in-variables Kalman filter*, has linear in T computational complexity.

Optional continuation of Exercise 4.5

As in the misfit computation in the static case, see Exercise 4.1, you can continue this exercise by considering alternative solutions that use different representations of the model. Different solutions of the problem give a better perspective and allow us to choose the most robust, efficient, simple to implement, *etc* method.

1. Implement and test the method for computing (misfit $\mathscr{L}_{\mathtt{m},\ell}$), based on the input/state/output representation of the model.
2. Explore the approaches based on the kernel and image representations.
3. Compare the different solution methods with respect to the following criteria.

   - **Theory:** which method was easiest/shortest to derive?
   - **Insight:** which method did you find most insightful in understating the misfit computational problem?
   - **Algorithms:** which one is most efficient computationally?

Chapter 5

5.1 (Matrix representation of polynomial multiplication).

1. The equivalence of the polynomial product $c(z) = a(z)b(z)$ and the matrix-vector product $c = \mathscr{M}_n(a)b$ can be proven by direct inspection of the two expression. From $c(z) = a(z)b(z)$, we have that

$$c_i = \sum_{k=0}^{m} a_k b_{i-k}. \qquad (*)$$

   On the other hand, from $c = \mathscr{M}_n(a)b$, we have that c_i is also given by $(*)$.
2. The statement follows from part 1 and the fact that convolution $c = a \star b$ of sequences is equivalent to polynomial multiplication $c(z) = a(z)b(z)$. Indeed, $(*)$ is the definition of the discrete convolution $c = a \star b$.
3. As in part 1, the equivalence of $c = a(\sigma)b$ and $\mathscr{M}^{\top}_{m-n}(a)b$ can be proven by inspection. By definition, from $c = a(\sigma)b$, we have

$$c_i = \sum_{k=0}^{m} a_k b_{i+k}. \qquad (**)$$

   On the other hand, from $c = \mathscr{M}^{\top}_n(a)b$, we have that c_i is also given by $(**)$.

5.2 (Computing approximate common factor with `slra`). First we define the reduced Sylvester matrix structure ($\mathscr{R}_{\mathrm{d}}$), using the `red_sylv` function:

247a ⟨*Define the reduced Sylvester matrix structure* `tts` 247a⟩≡ (247e)
```
n = length(p) - 1;
tts = red_sylv(1:n + 1, n + 2:2 * (n + 1), d)';
```
Uses `red_sylv` 151.

The structure parameter vector is concatenation of the coefficients of p and q:

247b ⟨*Call* `slra` 247b⟩≡ (247e)
```
[z, phqh, info] = slra(tts [p(:); q(:)], size(tts, 1) - 1);
```
Uses `slra` 116c.

The approximating polynomials $\widehat{p}$ and $\widehat{q}$ are extracted from the parameter `phqh`:

247c ⟨*Extract* $\widehat{p}$ *and* $\widehat{q}$ *from the* `slra` *parameter vector* 247c⟩≡ (247e)
```
ph = phqh(1:n + 1); qh = phqh(n + 2:end);
```

In order to find the approximate common factor c, we use $((u,v) \mapsto c)$,

247d ⟨*Compute the approximate common factor from the* `slra` *kernel parameter* 247d⟩≡ (247e)
```
v = z(1:n - d + 1); u = - z(n - d + 2:end);
c = [blktoep(u, d + 1)'; blktoep(v, d + 1)'] \ [p(:); q(:)];
```
Uses `blktoep` 117a.

Putting everything together, we obtain the following function:

247e ⟨*Approximate common factor, using* `slra` 247e⟩≡
```
function [c, ph, qh, M, info] = agcd_slra(p, q, d)
⟨Define the reduced Sylvester matrix structure tts 247a⟩
⟨Call slra 247b⟩
⟨Extract p̂ and q̂ from the slra parameter vector 247c⟩
⟨Compute the approximate common factor from the slra kernel parameter 247d⟩
M = norm([p(:); q(:)] - phqh);
```
Defines:
`agcd_slra`, used in chunk 248.

5.3 (Approximate common factor numerical examples). Algorithm 8 converges in 4 iteration steps and the `slra` algorithm (called from `agcd_slra`) converges in 11 iteration steps. Both methods find the approximate common factor

$$c(z) = 3.9830 + 1.9998z + 1.0000z^2,$$

to which correspond approximating polynomials

$$\widehat{p}(z) = 20.0500 + 18.0332z + 9.0337z^2 + 2.0001z^3$$
$$\widehat{q}(z) = 20.0392 + 14.0178z + 7.0176z^2 + 0.9933z^3.$$

The approximation error is $M(c) = 0.0126$.

248 ⟨*Approximate common factor numerical examples* 248⟩≡
```
p = conv([1 2 4], [2 5]) + [0 0.04 0.03 0.05];
q = conv([1 2 4], [1 5]) + [0 0.01 0.02 0.04];
d = 2;
[c, ph, qh, M, info] = agcd_slra(p, q, d);
```
Uses `agcd_slra` 247e.

Chapter 6

6.1 (Missing values completion with given model). Two solutions are presented:

- using an image representation of the model, and
- using an input/state/output representation.

The solutions are constructive and lead to algorithms for solving the problem. Both algorithms complete the sequence or prove that the sequence can not be completed.

Using image representation

Assuming that the model $\mathscr{B}$ is controllable, it has an image representation $\mathscr{B} = \operatorname{image}\big(P(\sigma)\big)$. For any $w \in \mathscr{B}|_T$, there is latent variable v, such that $w := \mathscr{T}(P)v$. Therefore, the sequence $w_{\mathrm{d}} \in (\mathbb{R}^{\mathtt{q}}_{\mathtt{e}})^T$ with missing values $w_{\mathrm{d},\mathscr{I}_{\mathrm{m}}} = \mathtt{NaN}$ and given values $w_{\mathrm{d},\mathscr{I}_{\mathrm{g}}}$ is a trajectory of $\mathscr{B}$ if and only if the system of linear equations

$$w_{\mathrm{d},\mathscr{I}_{\mathrm{g}}} = \mathscr{T}_{\mathscr{I}_{\mathrm{g}}}(P)v \qquad (v)$$

has a solution $\widehat{v}$, in which case a completion of the missing values is given by

$$\widehat{w}_{\mathscr{I}_{\mathrm{m}}} = \mathscr{T}_{\mathscr{I}_{\mathrm{m}}}(P)\widehat{v}. \qquad (\widehat{w})$$

The solution presented is summarized in Algorithm 13.

Algorithm 13 Missing values completion with model $\mathscr{B} = \text{image}\big(P(\sigma)\big)$.

Input: Polynomial matrix $P(z)$ defining an image representation $\mathscr{B} = \text{image}\big(P(\sigma)\big)$ of the system and a sequence $w_\mathrm{d} \in (\mathbb{R}^\mathtt{q}_\mathrm{e})^T$.
1: **if** (v) has a solution $\widehat{v}$ **then**
2: $w_\mathrm{d} \in \mathscr{B}|_T$, with a certificate $(\widehat{w})$.
3: **else**
4: $w_\mathrm{d} \notin \mathscr{B}|_T$, with a certificate $w_{\mathrm{d},\mathscr{I}_\mathrm{g}} \notin \mathscr{B}_{\mathscr{I}_\mathrm{g}}$.
5: **end if**
Output: The completed sequence $\widehat{w}$ or a certificate that $w_{\mathrm{d},\mathscr{I}_\mathrm{g}} \notin \mathscr{B}_{\mathscr{I}_\mathrm{g}}$.

Using input/state/output representation

In the general case of a possibly uncontrollable system $\mathscr{B}$, the problem can be approached using an input/state/output representation $\mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$. The solution is again a solvability test for a system of linear equations with a coefficients matrix

$$P := \begin{bmatrix} 0 & I \\ \mathscr{O} & \mathscr{T}(H) \end{bmatrix} \in \mathbb{R}^{\mathtt{q}T \times (\mathtt{n}+\mathtt{m}T)},$$

which consists of the extended observability matrix $\mathscr{O}$ and the low-triangular block-Toeplitz matrix $\mathscr{T}$ of the Markov parameters, see $(\mathscr{O}, \mathscr{T})$. For $w \in \mathscr{B}|_T$, we have

$$w = \begin{bmatrix} u \\ y \end{bmatrix} = \begin{bmatrix} 0 & I \\ \mathscr{O} & \mathscr{T} \end{bmatrix} \begin{bmatrix} x_\mathrm{ini} \\ u \end{bmatrix} =: Pv'.$$

Therefore, the sequence $w_\mathrm{d} \in (\mathbb{R}^\mathtt{q}_\mathrm{e})^T$ with missing values $w_{\mathrm{d},\mathscr{I}_\mathrm{m}} = \mathtt{NaN}$ and given values $w_{\mathrm{d},\mathscr{I}_\mathrm{g}}$ is a trajectory of $\mathscr{B}$ if and only if the system of linear equations

$$w_{\mathrm{d},\mathscr{I}_\mathrm{g}} = P_{\mathscr{I}_\mathrm{g}}(M)v' \qquad (v')$$

has a solution $\widehat{v}'$, in which case a completion of the missing values is given by

$$\widehat{w}_{\mathscr{I}_\mathrm{m}} = P_{\mathscr{I}_\mathrm{m}}\widehat{v}'. \qquad (\widehat{w}')$$

The solution method presented is summarized in Algorithm 14.

Note B.3 (Computational cost). Solving (v) and (v') by general purpose methods that do not exploit the structure of the matrices $\mathscr{T}$ and $\mathscr{O}$ requires $O(T^3)$ operations.

Note B.4 (Approximate solution). If $w_\mathrm{d} \notin \mathscr{B}|_T$, (v) and (v') have no solution. Computing a least squares approximate solution leads to a "smoothed sequence" $\widehat{w}$, which simultaneously approximates the given and completes the missing values.

Algorithm 14 Missing values completion with a model $\mathscr{B} = \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$.

Input: Matrices A,B,C,D defining an input/state/output representation $\mathscr{B} = \mathscr{B}_{\mathrm{i/s/o}}(A,B,C,D)$ of the system and a sequence $w_\mathrm{d} \in (\mathbb{R}^\mathtt{q}_\mathrm{e})^T$.
1: **if** (v') has a solution $\widehat{v}'$ **then**
2: $w_\mathrm{d} \in \mathscr{B}|_T$, with a certificate $(\widehat{w}')$.
3: **else**
4: $w_\mathrm{d} \notin \mathscr{B}|_T$, with a certificate $w_{\mathrm{d},\mathscr{I}_\mathrm{g}} \notin \mathscr{B}_{\mathscr{I}_\mathrm{g}}$.
5: **end if**
Output: The completed sequence $\widehat{w}$ or a certificate that $w_{\mathrm{d},\mathscr{I}_\mathrm{g}} \notin \mathscr{B}_{\mathscr{I}_\mathrm{g}}$.

Note B.5 (Online filtering with missing data). Finding a recursive online algorithm for solving (v) and (v') approximately in the least squares sense is an open problem. Such an algorithm would generalize the Kalman filter to missing data estimation.

6.2 (Subspace method for data-driven step response simulation). MATLAB implementation of the method, described in the solution of Exercise 3.5:

250a ⟨*Data-driven step response simulation, subspace method* 250a⟩≡ 250d▷
```
function sh = dd_step_subspace(wd, n, Tf)
ud = wd(:, 1); yd = wd(:, 2); T = length(ud);
H = [blkhank(ud, n + Tf); blkhank(yd, n + Tf)];
u = [zeros(n, 1); ones(Tf, 1)]; yp = zeros(n, 1);
sh = H(end-Tf+1:end, :) * pinv(H(1:end-Tf, :)) * [u; yp];
```
Uses blkhank 26a.

In order to study empirically the estimation accuracy in the present of noise, we simulate data in the errors-in-variables setup.

250b ⟨*Test data-driven step response simulation* 250b⟩≡ 250c▷
```
n = 2; T = 50; Tf = 5; sn = 0.01;

sys0 = drss(n); u0 = rand(T, 1); y0 = lsim(sys0, u0);
w0 = [u0 y0]; wt = randn(T, 2);
wd = w0 + sn * norm(w0) * wt / norm(wt);
```

Let $\bar{s}$ be the step response of the true system and $\widehat{s}$ the estimated step response. The estimation accuracy is evaluated by the relative error

$$e := \|\bar{s} - \widehat{s}\|_2 / \|\bar{s}\|_2.$$

250c ⟨*Test data-driven step response simulation* 250b⟩+≡ ◁250b 251a▷
```
s0 = step(sys0, Tf); s0 = s0(1:Tf);
e = @(sh) norm(s0 - sh) / norm(s0);
sh_subspace = dd_step_subspace(wd, n, Tf); e(sh_subspace)
```

6.3 (Data-driven step response simulation, using the SLRA package). The following function solves (DD SIM), in the case of a step response estimation, using the SLRA package:

250d ⟨*Data-driven step response simulation, subspace method* 250a⟩+≡ ◁250a
```
function [sh, info] = dd_step_slra(wd, n, Tf)
uf = ones(Tf, 1); yf = NaN * ones(Tf, 1); wf = [uf yf];
opt.exct = {[], 1}; opt.wini = {0, 0}; opt.solver = 'm';
```

```
[sysh, info, wh] = ident({wd wf}, 1, n, opt);
sh = wh{2}(:, 2);
```

The estimate obtained with `dd_step_slra` is compared with the estimate obtained with the subspace method `dd_step_subspace` on the simulation setup of Example 6.2

251a ⟨*Test data-driven step response simulation* 250b⟩+≡ ◁250c

```
sh_slra = dd_step_slra(wd, n, Tf); e(sh_slra)
```

Chapter 7

7.1 (Exact line fitting). The points $d_1,\dots,d_N$, $d_i = \left[\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right]$ lie on line if and only if there is a nonzero vector $\begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix} \neq 0$, such that

$$R_1 x_i + R_2 y_i + R_3 = 0, \quad \text{for } i = 1,\dots,N.$$

This system of linear equations can be written as a matrix equation

$$\underbrace{\begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix}}_{R} \underbrace{\begin{bmatrix} x_1 & \cdots & x_N \\ y_1 & \cdots & y_N \\ 1 & \cdots & 1 \end{bmatrix}}_{D_{\text{ext}}} = 0.$$

Since the parameter vector R is nonzero, we have that, by construction,

$$\operatorname{rank}(D_{\text{ext}}) \leq 2. \tag{$*$}$$

We have shown that the points $d_i = \left[\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right]$, $i = 1,\dots,N$ lie on a line if and only if a matrix constructed from the data points is rank deficient. Moreover, the model parameters can be computed from a nonzero vector in the left kernel of this matrix.

The condition for exact line fitting ($*$) can be checked in MATLAB as follows:

251b ⟨*Check if the data can be fitted by a line* 251b⟩≡

```
N = size(d, 2); dext = [d; ones(1, N)];
if (rank(dext) < 3)
   disp('exact fit exists')
else
   disp('exact fit does not exist')
end
```

where the columns of `d` are the data points. When an exact line that fits the points exists, its parameter vector R can be computed by the `null` function:

251c ⟨*Find an exact fit of data by a line* 251c⟩≡

```
r = null(dext')';
```

7.2 (Approximate line fitting). Observing that

$$\sqrt{\textstyle\sum_{j=1}^{N} \left\| d_j - \widehat{d}_j \right\|_2^2} = \left\| \begin{bmatrix} d_1 & \cdots & d_N \end{bmatrix} - \begin{bmatrix} \widehat{d}_1 & \cdots & \widehat{d}_N \end{bmatrix} \right\|_{\text{F}},$$

the geometric line fitting problem can be rewritten as

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{D} \quad \|D - \widehat{D}\|_F \\ &\text{subject to} && \widehat{d}_1,\dots,\widehat{d}_N \in \widehat{\mathscr{B}} \text{ and } \widehat{\mathscr{B}} \text{ is a line in } \mathbb{R}^2. \end{aligned}$$

Using the result of Exercise 7.1, we replace the constraint by the rank constraint ($*$). This gives us the low-rank approximation problem:

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{D} \quad \|D - \widehat{D}\|_F \\ &\text{subject to} && \operatorname{rank}\left(\begin{bmatrix} \widehat{D} \\ \mathbf{1}_N^\top \end{bmatrix} \right) \leq 2. \end{aligned}$$

Note that due to the fixed row of ones, this problem is not unstructured low-rank approximation. Using the truncated singular value decomposition (see Theorem 4.5) in general does not preserve the structure of the extended data matrix D_{ext}. Ignoring the structure leads to a suboptimal solution. Optimal solution, based on the singular value decomposition, is developed in (Golub et al, 1987), see also (Zha, 1991).

7.3 (Exact conic section fitting). The points $d_1,\dots,d_N$, $d_i = \left[\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right]$ lie on a conic section if and only if there is a symmetric matrix $A = A^\top$, a vector b, and a scalar c, at least one of which nonzero, such that

$$d_i^\top A d_i + b^\top d_i + c = 0, \quad \text{for } i = 1,\dots,N. \tag{$**$}$$

Equation ($**$) is linear in the parameters A, b, c and has a matrix representation

$$\underbrace{\begin{bmatrix} a_{11} & 2a_{12} & b_1 & a_{22} & b_2 & c \end{bmatrix}}_{R} \underbrace{\begin{bmatrix} x_1^2 & \cdots & x_N^2 \\ x_1 y_1 & \cdots & x_N y_N \\ x_1 & \cdots & a_N \\ y_1^2 & \cdots & y_N^2 \\ y_1 & \cdots & y_N \\ 1 & \cdots & 1 \end{bmatrix}}_{D\text{ext}} = 0.$$

Since, by assumption, the parameter vector R is nonzero, we have that

$$\operatorname{rank}(D_{\text{ext}}) \leq 5. \tag{$*$}$$

We have shown that the points $d_i = \left[\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right]$, $i = 1,\dots,N$ lie on a conic section if and only if a matrix constructed from the data is rank deficient. Moreover, the model parameters can be computed from a nonzero vector in the left kernel of the matrix.

For the MATLAB implementation of the exact conic section fitting method, we define a function that constructs a column of the D_{ext} matrix

252a ⟨*Bivariate quadratic function* 252a⟩≡ (253c)

```
f = @(x, y) [x .^ 2; x .* y; x; y .^ 2; y; ones(size(x))];
```

The exact model's parameter vector is then computed as

252b ⟨*Find an exact fit of data by a conic section* 252b⟩≡

```
R = null(f(d(1, :), d(2, :))')';
```

The obtained conic section model can be plotted by the function

253a ⟨*Plot a conic section* 253a⟩≡
```
function H = plot_model(th, f, ax, c)
H = ezplot(@(a, b) th * f(a, b), ax);
for h = H', set(h, 'color', c, 'linewidth', 2); end
```
Uses plot_model 187c.

Applying the method on the data in the problem:

253b ⟨*Find and plot exact conic sections fitting the data* 253b⟩≡
```
d = [-1  1 1 -1;
     -1 -1 1  1];
plot(d(1, :), d(2, :), 'o', 'markersize', 12)
ax = 2 * axis;
R = null(f(d(1, :), d(2, :))')';
for i = 1:size(R, 1)
  hold on, plot_model(R(i, :), f, ax, c(i));
end
```
Uses plot_model 187c.

we obtained the two conic sections, shown in Figure B.2

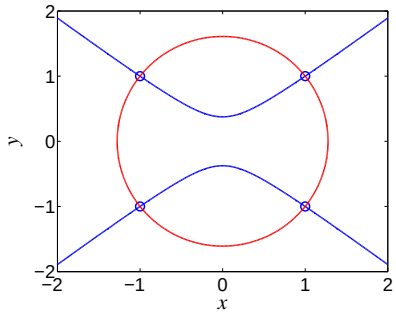


Fig. B.2: For the data points given in Problem 7.3 there are infinitely many exact fitting conic section models. Two of them are shown in the figure.

Note that when $\operatorname{rank}(D) < 5$, *i.e.*, the dimension of the left kernel of D is higher than one, the rows of R define different exact conic sections that fit the data. In this case, there are infinitely many exact conic section fitting models. Their parameter vectors can be obtained as linear combinations of the rows of the matrix R.

7.4 (Approximate conic section fitting). The algebraic fitting method involves the construction of the augmented data matrix dext and doing unstructured low-rank approximation by calling the function lra:

253c ⟨*Algebraic conic section fitting* 253c⟩≡
```
function R = function als(d)
⟨Bivariate quadratic function 252a⟩
```



```
R = lra(f(d(1, :), d(2, :)), 5);
```
Uses lra 105a.

The result R defines a kernel representation of the optimal algebraic fitting model $\widehat{\mathscr{B}}$. It can be plotted using the function plot_model.

Chapter 8

8.1 (Matrix centering).

$$\begin{aligned}\mathbf{E}\left(\mathbf{C}(D)\right) &= \mathbf{E}\left(D - \mathbf{E}(D)\mathbf{1}_N^\top\right)\\ &= \frac{1}{N}\left(D - \frac{1}{N}D\mathbf{1}_N\mathbf{1}_N^\top\right)\mathbf{1}_N\\ &= \frac{1}{N}D\mathbf{1}_N - \frac{1}{N^2}D\mathbf{1}_N\underbrace{\mathbf{1}_N^\top\mathbf{1}_N}_{N} = 0.\end{aligned}$$

8.2 (Mean computation as an optimal modeling). The optimization problem is a linear least squares problem and its solution is

$$\widehat{c} = D\mathbf{1}_N(\mathbf{1}_N^\top\mathbf{1}_N)^{-1} = \frac{1}{N}D\mathbf{1}_N = \mathbf{E}(D).$$

8.3 (Autonomous system identification with centering, using internal model). Let

$$\mathscr{B} = \ker\left(P(\sigma)\right) \quad \text{and} \quad \mathscr{B}_{\mathrm{c}} = \ker\left(P_{\mathrm{c}}(\sigma)\right)$$

be minimal kernel representations. The model $\mathscr{B}_{\mathrm{c}}$ for the constant has parameter

$$P_{\mathrm{c}}(z) := 1 - z.$$

Therefore, the augmented system $\mathscr{B}_{\mathrm{ext}}$ has a kernel representation

$$\mathscr{B}_{\mathrm{ext}} = \ker\left(P_{\mathrm{ext}}(\sigma)\right), \quad \text{with } P_{\mathrm{ext}}(z) = P(z)P_{\mathrm{c}}(z) = P(z)(1-z).$$

The autonomous system identification problem with centering becomes then

$$\begin{aligned}&\text{minimize} && \text{over } \widehat{y} \text{ and } P \in \mathbb{R}[z],\ \deg(P) = \ell && \|y_{\mathrm{d}} - \widehat{y}\|\\ &\text{subject to} && P_{\mathrm{c}}(\sigma)(1-\sigma)\widehat{y} = 0.\end{aligned}$$

Using Lemma 5.10 and the matrix representation of the polynomial product (see Exercise 5.1), we have an equivalent Hankel low-rank approximation problem

$$\begin{aligned} &\text{minimize} && \text{over } \widehat{y} \text{ and } P \in \mathbb{R}^{1\times(\ell+1)} && \|y_{\mathrm{d}} - \widehat{y}\| \\ &\text{subject to} && P \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{bmatrix} && \mathscr{H}_{\ell+2}(\widehat{y}) = 0, \end{aligned}$$

with structured kernel. The SLRA package allows specification of linear constraints on the kernel parameter R, so that the autonomous system identification problem with centering can be solved by the SLRA package.

255 ⟨*Autonomous system identification with centering, using internal model* 255⟩≡

```
function [sys_aug, yh, info] = aut_ident_center(y, ell, opt)
s.m = ell + 2; opt.psi = blktoep([1 -1], ell + 1);
[yh, info] = slra(y, s, ell + 1, opt);
sysh = h2ss(yh, ell + 1); xini_aug = sysh.b;
sysh_aug = ss(sysh.a, [], sysh.c, [], -1);
```

Uses `blktoep` 117a, `h2ss` 74c, and `slra` 116c.

Chapter A

A.1 (Geometric interpretation of the total least squares). The constraint

$$\widehat{a}_j x = \widehat{b}_j, \quad \text{for } j = 1, \ldots, N$$

is equivalent to the constraint that the points

$$\widehat{d}_1 := \begin{bmatrix} \widehat{a}_1 \\ \widehat{b}_1 \end{bmatrix}, \ \ldots, \ \widehat{d}_N := \begin{bmatrix} \widehat{a}_N \\ \widehat{b}_N \end{bmatrix}$$

lie on the line

$$\mathscr{B}_{\mathrm{i/o}}(x) := \{ d = \left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right] \in \mathbb{R}^2 \mid ax = b \}.$$

Therefore, problem (tls) can be written as

$$\begin{aligned} &\text{minimize} && \text{over } x \text{ and } \widehat{d}_1, \ldots, \widehat{d}_N && \textstyle\sum_{j=1}^N \left\| d_j - \widehat{d}_j \right\|_2^2 \\ &\text{subject to} && \widehat{d}_j \in \mathscr{B}_{\mathrm{i/o}}(x), && \text{for } j = 1, \ldots, N. \end{aligned} \tag{tls'}$$

In turn, problem (tls') is equivalent to minimization of $f_{\mathrm{tls}} : \mathbb{R} \to \mathbb{R}$ defined by

$$f_{\mathrm{tls}}(x) := \min_{\widehat{d}_1, \ldots, \widehat{d}_N} \sum_{j=1}^N \left\| d_j - \widehat{d}_j \right\|_2^2 \quad \text{subject to} \quad \widehat{d}_j \in \mathscr{B}_{\mathrm{i/o}}(x), \ \text{for } j = 1, \ldots, N. \tag{tls"}$$

The minimization problem in (tls") is separable in the variables $\widehat{d}_1, \ldots, \widehat{d}_N$, *i.e.*, (tls") decouples into N independent problems

$$f_{\mathrm{tls},j}(x) = \min_{\widehat{d}_j} \left\| d_j - \widehat{d}_j \right\|_2^2 \quad \text{subject to} \quad \widehat{d}_j \in \mathscr{B}_{\mathrm{i/o}}(x), \quad \text{for } i = 1, \ldots, N.$$

By the orthogonality principle, $f_{\mathrm{tls},j}(x)$ is the squared orthogonal distance from d_j to the line $\mathscr{B}_{\mathrm{i/o}}(x)$. Subsequently,

$$f_{\mathrm{tls}}(x) = \textstyle\sum_{j=1}^N f_{\mathrm{tls},j}(x)$$

is the sum of squared orthogonal distances from the data points to the line $\mathscr{B}_{\mathrm{i/o}}(x)$.

For any $x \in \mathbb{R}$, $\mathscr{B}_{\mathrm{i/o}}(x)$ is a line passing through the origin and any line passing through the origin, except for the vertical line, corresponds to a set $\mathscr{B}_{\mathrm{i/o}}(x)$, for some $x \in \mathbb{R}$. Therefore, the total least squares problem $\min_{x\in\mathbb{R}} f_{\mathrm{tls}}(x)$ minimizes the sum of squared orthogonal distances from the data points to a line, over all lines passing through the origin, except for the vertical line.

A.2 (Unconstrained problem, equivalent to the total least squares problem). The total least squares approximation problem (tls) is $\min_x f_{\mathrm{tls}}(x)$, where

$$f_{\mathrm{tls}}(x) = \min_{\widehat{a},\widehat{b}} \left\| \begin{bmatrix} a & b \end{bmatrix} - \begin{bmatrix} \widehat{a} & \widehat{b} \end{bmatrix} \right\|_{\mathrm{F}}^2 \quad \text{subject to} \quad \widehat{a}x = \widehat{b} \tag{f_{tls}}$$

or with the change of variables $\Delta a := a - \widehat{a}$ and $\Delta b := b - \widehat{b}$,

$$f_{\mathrm{tls}}(x) = \min_{\Delta a, \Delta b} \left\| \begin{bmatrix} \Delta a & \Delta b \end{bmatrix} \right\|_{\mathrm{F}}^2 \quad \text{subject to} \quad ax - b = \Delta a x - \Delta b. \tag{f'_{tls}}$$

Define

$$\Delta b := ax - b, \quad \Delta D := \begin{bmatrix} \Delta a & \Delta b \end{bmatrix}^\top, \quad \text{and} \quad R = \begin{bmatrix} x^\top & -1 \end{bmatrix}$$

in order to write (f'_{tls}) as a standard linear least norm problem

$$\min_{\Delta D} \left\| \Delta D \right\|_{\mathrm{F}}^2 \quad \text{subject to} \quad R \Delta D = \Delta b^\top.$$

The least norm solution for ΔD is

$$\Delta D^* = \frac{1}{RR^\top} R^\top \Delta b,$$

so that, we have

$$f_{\mathrm{tls}}(x) = \|\Delta D^*\|_{\mathrm{F}}^2 = \operatorname{trace}\left((\Delta D^*)^\top D^* \right) = \frac{\Delta b^\top \Delta b}{RR^\top} = \frac{\|ax - b\|^2}{\|x\|^2 + 1}.$$

From Problem 1.2 and the derivation of f_{tls}, we see that $f_{\mathrm{tls}}(x)$ is the sum of squared orthogonal distances from the data points to the model $\mathscr{B}_{\mathrm{i/o}}(x)$, defined by x.

A.3 (Lack of total least squares solution). The total least squares line fitting method, applied to the data in Problem 1.1 leads to the overdetermined system

$$\underbrace{\mathrm{col}(-2,-1,0,1,2,2,1,0,-1,-2)}_{\mathbf{a}}x = \underbrace{\mathrm{col}(1,4,6,4,1,-1,-4,-6,-4,-1)}_{\mathbf{b}}.$$

Therefore, using the (tls') formulation, the problem is to minimize the function

$$f_{\mathrm{tls}}(x) = \frac{(\mathbf{a}x-\mathbf{b})^\top(\mathbf{a}x-\mathbf{b})}{x^2+1} = \cdots \begin{array}{c}\text{substituting } \mathbf{a} \text{ and } \mathbf{b} \text{ with}\\ \text{their numerical values}\end{array} \cdots = 20\frac{x^2+7}{x^2+1}.$$

The first derivative of f_{tls} is

$$\frac{\mathrm{d}}{\mathrm{d}x}f_{\mathrm{tls}}(x) = -\frac{240x}{(x^2+1)^2},$$

so that f_{tls} has a unique stationary point at $x=0$. The second derivative of f_{tls} at $x=0$ is negative, so that the stationary point is a maximum. This proves that f_{tls} has no minimum and the total least squares problem has no solution.

Figure B.3 shows the plot of f_{tls} over the interval $[-6.3, 6.3]$. It can be verified that the infimum of f_{tls} is 20 and f_{tls} has asymptotes $f_{\mathrm{tls}}(x) \to 20$ for $x \to \pm\infty$, *i.e.*, the infimum is achieved asymptotically as x tends to infinity or minus infinity.

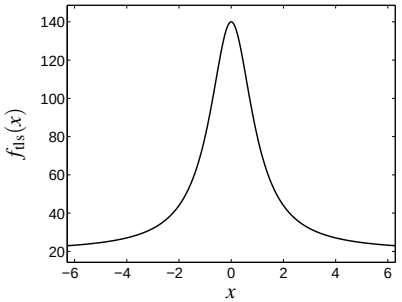


Fig. B.3: Cost function of the total least squares problem (tls') in Problem 1.4.

References

Golub G, Hoffman A, Stewart G (1987) A generalization of the Eckart–Young–Mirsky matrix approximation theorem. Linear Algebra Appl 88/89:317–327

Pólya G (1957) How to Solve It. Doubleday

Zha H (1991) The restricted singular value decomposition of matrix triplets. SIAM J Matrix Anal Appl 12(1):172–194

Appendix C
Proofs

It is time to dispel a popular misconception. The goal of mathematics is discovery, *not proof.*

Chover (1972)

The proof is meaningful when it answers the students doubts, when it proves what is not obvious. Intuition may fly the student to a conclusion but where doubt remains he may then be asked to call upon plodding logic to show the overland route to the same goal.

Kline (1974)

Emphasize the difference between "seeing" and "proving": Can you see clearly that the step is correct? But can you also prove that the step is correct?

Pólya (1957)

Chapter 4

Proof of Proposition 4.4

The probability density function of the observation vector $\operatorname{vec}(D)$ is

$$p_{\widehat{\mathscr{B}},\widehat{D}}\big(\operatorname{vec}(D)\big) = \begin{cases} \text{const} \cdot \exp\left(-\frac{1}{2s^2}\|\operatorname{vec}(D) - \operatorname{vec}(\widehat{D})\|^2_{V^{-1}}\right), \\ \quad \text{if } \operatorname{image}(\widehat{D}) \subset \widehat{\mathscr{B}} \text{ and } \dim(\widehat{\mathscr{B}}) \leq \mathtt{m} \\ 0, \quad \text{otherwise.} \end{cases}$$

"const" is a term that does not depend on $\widehat{D}$ and $\widehat{\mathscr{B}}$. The log-likelihood function is

$$\ell(\widehat{\mathscr{B}}, \widehat{D}) = \begin{cases} -\text{const} \cdot \frac{1}{2s^2}\|\operatorname{vec}(D) - \operatorname{vec}(\widehat{D})\|^2_{V^{-1}}, \\ \quad \text{if } \operatorname{image}(\widehat{D}) \subset \widehat{\mathscr{B}} \text{ and } \dim(\widehat{\mathscr{B}}) \leq \mathtt{m} \\ -\infty, \quad \text{otherwise,} \end{cases}$$

and the maximum likelihood estimation problem is

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{\mathscr{B}} \text{ and } \widehat{D} \quad \dfrac{1}{2s^2}\|\operatorname{vec}(D) - \operatorname{vec}(\widehat{D})\|^2_{V^{-1}} \\ \text{subject to} & \operatorname{image}(\widehat{D}) \subset \widehat{\mathscr{B}} \text{ and } \dim(\widehat{\mathscr{B}}) \leq \mathtt{m}. \end{array}$$

This problem is equivalent to Problem 4.2 with $\|\cdot\| = \|\cdot\|_{V^{-1}}$.

Proof of Proposition 4.5

The proof is given in (Vanluyten et al, 2006). Let $\widehat{D}^*$ be a solution to

$$\widehat{D}^* := \arg\min_{\widehat{D}} \|D - \widehat{D}\| \quad \text{subject to} \quad \operatorname{rank}(\widehat{D}) \leq \mathtt{m}. \tag{LRA}$$

and let

$$\widehat{D}^* := U^* \Sigma^* (V^*)^\top$$

be a singular value decomposition of $\widehat{D}^*$. By the unitary invariance of the Frobenius norm, we have that

$$\|D - \widehat{D}^*\|_{\mathrm{F}} = \|(U^*)^\top (D - \widehat{D}^*) V^*\|_{\mathrm{F}} = \|\underbrace{(U^*)^\top D V^*}_{\widehat{D}} - \Sigma^*\|_{\mathrm{F}},$$

which shows that Σ^* is an optimal approximation of $\widehat{D}$. Partition

$$\widehat{D} =: \begin{bmatrix} \widehat{D}_{11} & \widehat{D}_{12} \\ \widehat{D}_{21} & \widehat{D}_{22} \end{bmatrix}$$

conformably with $\Sigma^* =: \left[\begin{smallmatrix} \Sigma_1^* & 0 \\ 0 & 0 \end{smallmatrix}\right]$ and observe that

$$\operatorname{rank}\left(\begin{bmatrix} \Sigma_1^* & \widehat{D}_{12} \\ 0 & 0 \end{bmatrix}\right) \leq \mathtt{m} \quad \text{and} \quad \widehat{D}_{12} \neq 0 \implies \left\|\widehat{D} - \begin{bmatrix} \Sigma_1^* & \widehat{D}_{12} \\ 0 & 0 \end{bmatrix}\right\|_{\mathrm{F}} < \left\|\widehat{D} - \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & 0 \end{bmatrix}\right\|_{\mathrm{F}},$$

so that $\widehat{D}_{12} = 0$. Similarly $\widehat{D}_{21} = 0$. Observe also that

$$\operatorname{rank}\left(\begin{bmatrix} \widehat{D}_{11} & 0 \\ 0 & 0 \end{bmatrix}\right) \leq \mathtt{m} \quad \text{and} \quad \widehat{D}_{11} \neq \Sigma_1^* \implies \left\|\widehat{D} - \begin{bmatrix} \widehat{D}_{11} & 0 \\ 0 & 0 \end{bmatrix}\right\|_{\mathrm{F}} < \left\|\widehat{D} - \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & 0 \end{bmatrix}\right\|_{\mathrm{F}},$$

so that $\widehat{D}_{11} = \Sigma_1^*$. Therefore,

$$\widehat{D} = \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & \widehat{D}_{22} \end{bmatrix}.$$

Let

$$\widehat{D}_{22} = U_{22} \Sigma_{22} V_{22}^\top$$

be the singular value decomposition of $\widehat{D}_{22}$. Then the matrix

$$\begin{bmatrix} I & 0 \\ 0 & U_{22}^\top \end{bmatrix} \widehat{D} \begin{bmatrix} I & 0 \\ 0 & V_{22} \end{bmatrix} = \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & \Sigma_{22} \end{bmatrix}$$

has optimal rank-$\mathtt{m}$ approximation $\Sigma^* =: \left[\begin{smallmatrix} \Sigma_1^* & 0 \\ 0 & 0 \end{smallmatrix}\right]$, so that

$$\min\left(\operatorname{diag}(\Sigma_1^*)\right) > \max\left(\operatorname{diag}(\Sigma_{22})\right)$$

Therefore,

$$D = U^* \begin{bmatrix} I & 0 \\ 0 & U_{22} \end{bmatrix} \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & \Sigma_{22} \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & V_{22}^\top \end{bmatrix} (V^*)^\top$$

is a singular value decomposition of D.

Then, if $\sigma_{\mathtt{m}} > \sigma_{\mathtt{m}+1}$, the rank-$\mathtt{m}$ truncated singular value decomposition

$$\widehat{D}^* = U^* \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & 0 \end{bmatrix} (V^*)^\top = U^* \begin{bmatrix} I & 0 \\ 0 & U_{22} \end{bmatrix} \begin{bmatrix} \Sigma_1^* & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & V_{22}^\top \end{bmatrix} (V^*)^\top$$

is unique and $\widehat{D}^*$ is the unique solution of (LRA). Moreover, $\widehat{D}^*$ is simultaneously optimal in any unitarily invariant norm.

Proof of Theorem 4.18

Defining

$$\Delta p_{\mathscr{I}_{\mathrm{g}}} := p_{\mathscr{I}_{\mathrm{g}}} - \widehat{p}_{\mathscr{I}_{\mathrm{g}}}$$

and using the identity

$$R\mathscr{S}(\widehat{p}) = 0 \quad \Longleftrightarrow \quad G\widehat{p} = -\operatorname{vec}(RS_0),$$

we have

$$R\mathscr{S}(\widehat{p}) = 0 \quad \Longleftrightarrow \quad \begin{bmatrix} G_{:,\mathscr{I}_{\mathrm{g}}} & G_{:,\mathscr{I}_{\mathrm{m}}} \end{bmatrix} \begin{bmatrix} p_{\mathscr{I}_{\mathrm{g}}} - \Delta p_{\mathscr{I}_{\mathrm{g}}} \\ \widehat{p}_{\mathscr{I}_{\mathrm{m}}} \end{bmatrix} = -\operatorname{vec}(RS_0).$$

Therefore, (M) is equivalent to

$$M(R) := \min_{\Delta p_{\mathscr{I}_{\mathrm{g}}} \in \mathbb{R}^{n_{\mathrm{g}}},\ \widehat{p}_{\mathscr{I}_{\mathrm{m}}} \in \mathbb{R}^{n_{\mathrm{m}}}} \|\Delta p_{\mathscr{I}_{\mathrm{g}}}\|_2^2 \quad \text{subject to}$$

$$\begin{bmatrix} G_{:,\mathscr{I}_{\mathrm{g}}} & G_{:,\mathscr{I}_{\mathrm{m}}} \end{bmatrix} \begin{bmatrix} \Delta p_{\mathscr{I}_{\mathrm{g}}} \\ -\widehat{p}_{\mathscr{I}_{\mathrm{m}}} \end{bmatrix} = G_{:,\mathscr{I}_{\mathrm{g}}} p_{\mathscr{I}_{\mathrm{g}}} + \operatorname{vec}(RS_0),$$

which is a generalized linear least norm problem. The solution follows from Lemma 4.19.

Proof of Theorem 4.19

Under assumption 1, B has a nontrivial left kernel of dimension $m - n_y$. Therefore, for the nonsingular matrix $T = \begin{bmatrix} B^+ \\ B^\perp \end{bmatrix} \in \mathbb{R}^{m \times m}$, we have that

$$TB = \begin{bmatrix} B^+ \\ B^\perp \end{bmatrix} B = \begin{bmatrix} B^+B \\ B^\perp B \end{bmatrix} = \begin{bmatrix} I_{n_y} \\ 0 \end{bmatrix}.$$

Pre-multiplying both sides of the constraint of (GLN) by T, we have the following equivalent constraint

$$\begin{bmatrix} B^+Ax \\ B^\perp Ax \end{bmatrix} + \begin{bmatrix} y \\ 0 \end{bmatrix} = \begin{bmatrix} B^+c \\ B^\perp c \end{bmatrix}.$$

The first equation

$$y = B^+(c - Ax)$$

uniquely determines y, given x. The second equation

$$B^\perp Ax = B^\perp c \qquad (*)$$

defines a linear constraint for x only. By assumption 2, it is an underdetermined system of linear equations. Therefore, (GLN) is equivalent to the following standard least norm problem

$$f = \min_x \|x\|_2^2 \quad \text{subject to} \quad B^\perp Ax = B^\perp c. \qquad \text{(GLN')}$$

By assumption 3, the solution is unique and is given by (SOL).

Proof of Theorem 4.20

We call a set $\mathscr{R} \subset \mathbb{R}^{d \times m}$ a "homogeneous set" if for all $R \in \mathscr{R}$ and for all nonsingular matrices $T \in \mathbb{R}^{d \times d}$, $TR \in \mathscr{R}$. Let R be a solution to (SLRAC''_R) with the constraint $R \in \mathscr{R}$, where $\mathscr{R}$ is a homogeneous set. We will show that

$$\|RR^\top - I_{m-r}\|_{\mathrm{F}}^2 = m - r - \operatorname{rank}(R). \qquad (*)$$

There exists an orthogonal matrix U diagonalizing $RR^\top$. We have,

$$\begin{aligned} \|RR^\top - I_{m-r}\|_{\mathrm{F}}^2 &= \|URR^\top U^\top - I_{m-r}\|_{\mathrm{F}}^2 \\ &= \|\operatorname{diag}(a_1, \ldots, a_{\operatorname{rank}(R)}, 0, \ldots, 0) - I_{m-r}\|_{\mathrm{F}}^2, \quad \text{where } a_i > 0 \\ &= \sum_{i=1}^{\operatorname{rank}(R)} (a_i - 1)^2 + m - r - \operatorname{rank}(R). \end{aligned}$$

Suppose that $a_i \neq 1$ for some i. The matrix

$$R' = \operatorname{diag}(1, \ldots, 1, 1/\sqrt{a_i}, 1, \ldots, 1)R$$

has the same row span and rank as R, so that by the homogeneity property of M, $M(R) = M(R')$. However, we have

$$\|RR^\top - I_{m-r}\|_{\mathrm{F}}^2 > \|R'R'^\top - I_{m-r}\|_{\mathrm{F}}^2,$$

so that $R' \in \mathscr{R}$ achieves smaller value of the cost function of (SLRAC$''_R$) than R. This is a contradiction. Therefore, $a_i = 1$ for all i. This concludes the proof of ($*$).

So far we showed that minimization of the cost function in (SLRAC$''_R$) on homogeneous sets is equivalent to minimization of

$$M(R) + \gamma\big(m - r - \operatorname{rank}(R)\big). \tag{M''}$$

The set of full row rank matrices

$$\mathscr{R}_f := \{\, R \in \mathbb{R}^{(m-r)\times m} \mid \operatorname{rank}(R) = m - r \,\}$$

and the set of rank-deficient matrices

$$\mathscr{R}_d := \{\, R \in \mathbb{R}^{(m-r)\times m} \mid \operatorname{rank}(R) < m - r \,\}$$

are homogeneous. Denote the solutions of (SLRAC$''_R$) on these sets as

$$M_f^* := \inf_{R\in\mathscr{R}_f} M(R) + \gamma\|RR^\top - I_{m-r}\|_{\mathrm{F}}^2 \overset{(*)}{=} \inf_{R\in\mathscr{R}_f} M(R) < \gamma,$$

$$M_d^* := \inf_{R\in\mathscr{R}_d} M(R) + \gamma\|RR^\top - I_{m-r}\|_{\mathrm{F}}^2 \overset{(*)}{=} \inf_{R\in\mathscr{R}_d} \underbrace{M(R)}_{\geq 0} + \underbrace{\gamma\big(m - r - \operatorname{rank}(R)\big)}_{\geq \gamma}.$$

Then, $M_f^* < \gamma \leq M_d^*$ and

$$M^* := \inf_{R\in\mathbb{R}^{(m-r)\times m}} M(R) + \gamma\|RR^\top - I_{m-r}\|_{\mathrm{F}}^2 = M_f^*.$$

In addition, by the homogeneity of M, the minimum of (SLRAC$'_R$) coincides with M_f^*. Therefore, the solutions of (SLRAC$''_R$) and (SLRAC$'_R$) coincide if they exist.

Chapter 5

Proof of Lemma 5.10

($\Longrightarrow$) Assume that

$$w|_{T-\ell} \in \mathscr{B}|_{T-\ell} \quad \text{and} \quad \mathscr{B} \in \mathscr{L}_{\mathtt{m},\ell}^{\mathtt{q}} \tag{$w \in \mathscr{B}$}$$

holds and let $\ker(R)$, with $R(z) = \sum_{i=0}^{\ell} z^i R_i \in \mathbb{R}^{g\times \mathtt{q}}[z]$ full row rank, be a kernel representation of $\mathscr{B}$. The assumption $\mathscr{B} \in \mathscr{L}_{\mathtt{m},\ell}$ implies that $g \geq \mathtt{p} := \mathtt{q} - \mathtt{m}$. From $w \in \mathscr{B}|_T$, we have that

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \mathscr{H}_{\ell+1}(w) = 0, \tag{repr}$$

which implies that

$$\operatorname{rank}\big(\mathscr{H}_{\ell+1}(w)\big) \leq \mathtt{m}(\ell+1) + (\mathtt{q} - \mathtt{m})\ell. \tag{rank}$$

holds.

($\Longleftarrow$) Assume that (rank) holds. Then, there is a full row rank matrix

$$R := \begin{bmatrix} R_0 & R_1 & \cdots & R_\ell \end{bmatrix} \in \mathbb{R}^{\mathtt{p}\times \mathtt{q}(\ell+1)},$$

such that (repr) holds. Define the polynomial matrix $R(z) = \sum_{i=0}^{\ell} z^i R_i$. Then the system $\mathscr{B}$ induced by $R(z)$ via the kernel representation $\ker\big(R(\sigma)\big)$ is such that ($w \in \mathscr{B}$) holds.

Proof of Proposition 5.20

Assume that p and q have a common factor c of degree d. Then, there are polynomials u and v, such that $p = cu$ and $q = cv$. Therefore, $pv = qu$. Written in a matrix form, this equation is $\mathscr{T}^\top(p)v = \mathscr{T}^\top(q)u$ or

$$\mathscr{R}_{\mathtt{d}}(p,q) \begin{bmatrix} v \\ -u \end{bmatrix} = 0.$$

Since $z := \left[\begin{smallmatrix} v \\ -u \end{smallmatrix}\right] \neq 0$ and $\mathscr{R}_{\mathtt{d}}(p,q)$ has less columns than rows, we've proved that $\mathscr{R}_{\mathtt{d}}(p,q)$ is rank deficient.

Assume now that $\mathscr{R}_{\mathtt{d}}(p,q)$ is rank deficient. Then there is a nonzero vector z in the null space, *i.e.*, $\mathscr{R}_{\mathtt{d}}(p,q)z = 0$. Partitioning z as $z =: \left[\begin{smallmatrix} v \\ -u \end{smallmatrix}\right]$, we obtain polynomials u and v, such that $p = cu$ and $q = cv$, with c of degree d.

Proof of Theorem 5.22

The polynomial equations $p = uc$ and $q = vc$ are equivalent to the following systems of algebraic equations

$$\begin{bmatrix} \widehat{p}_0 \\ \widehat{p}_1 \\ \vdots \\ \widehat{p}_n \end{bmatrix} = \mathscr{T}_{\mathtt{d}+1}^\top(u) \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{\mathtt{d}} \end{bmatrix}, \quad \begin{bmatrix} \widehat{q}_0 \\ \widehat{q}_1 \\ \vdots \\ \widehat{q}_n \end{bmatrix} = \mathscr{T}_{\mathtt{d}+1}^\top(v) \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{\mathtt{d}} \end{bmatrix},$$

where the Toeplitz matrix constructor $\mathscr{T}$ is defined in ($\mathscr{T}$) on page 117. Rewriting and combining the above equations, we have that a polynomial c is a common factor of $\widehat{p}$ and $\widehat{q}$ with $\operatorname{degree}(c) \leq \mathtt{d}$ if and only if the system of equations

$$\begin{bmatrix} \widehat{p}_0 & \widehat{q}_0 \\ \widehat{p}_1 & \widehat{q}_1 \\ \vdots & \vdots \\ \widehat{p}_n & \widehat{q}_n \end{bmatrix} = \mathscr{T}_{n-\mathtt{d}+1}^{\top}(c) \begin{bmatrix} u_0 & v_0 \\ u_1 & v_1 \\ \vdots & \vdots \\ u_{n-\mathtt{d}} & v_{n-\mathtt{d}} \end{bmatrix}$$

has a solution.

The condition degree$(c) = \mathtt{d}$ implies that the highest power coefficient $c_{\mathtt{d}}$ of c is different from 0. Since c is determined up to a scaling factor, we can impose the normalization $c_{\mathtt{d}} = 1$. Conversely, imposing the constraint $c_{\mathtt{d}} = 1$ in the optimization problem to be solved ensures that degree$(c) = \mathtt{d}$. Therefore, Problem 5.17 is equivalent to

$$\begin{aligned} \text{minimize} \quad & \text{over } \widehat{p}, \widehat{q} \in \mathbb{R}^{n+1}, u, v \in \mathbb{R}^{n-\mathtt{d}+1}, \text{ and } c_0, \ldots, c_{\mathtt{d}-1} \in \mathbb{R} \\ & \qquad \text{trace}\left(\left(\begin{bmatrix} p & q \end{bmatrix} - \begin{bmatrix} \widehat{p} & \widehat{q} \end{bmatrix} \right)^{\top} \left(\begin{bmatrix} p & q \end{bmatrix} - \begin{bmatrix} \widehat{p} & \widehat{q} \end{bmatrix} \right) \right) \\ \text{subject to} \quad & \begin{bmatrix} \widehat{p} & \widehat{q} \end{bmatrix} = \mathscr{T}_{n-\mathtt{d}+1}^{\top}(c) \begin{bmatrix} u & v \end{bmatrix}. \end{aligned}$$

Substituting $\begin{bmatrix} \widehat{p} & \widehat{q} \end{bmatrix}$ in the cost function and minimizing with respect to $\begin{bmatrix} u & v \end{bmatrix}$ by solving a least squares problem gives the equivalent problem (ACF').

Chapter 8

Proof of Theorem 8.4

Using a kernel representation of the rank constraint

$$\text{rank}(\widehat{D}) \leq \mathtt{m} \quad \Longleftrightarrow \quad \text{there is full rank matrix } R \in \mathbb{R}^{(\mathtt{q}-\mathtt{m})\times \mathtt{q}}, \text{ such that } R\widehat{D} = 0,$$

we have the following equivalent problem to (LRA$_c$)

$$\begin{aligned} \text{minimize} \quad & \text{over } \widehat{D}, c, \text{ and } R \in \mathbb{R}^{(\mathtt{q}-\mathtt{m})\times \mathtt{q}} \quad \|D - c\mathbf{1}_N^{\top} - \widehat{D}\|_{\mathrm{F}}^2 \\ \text{subject to} \quad & R\widehat{D} = 0 \quad \text{and} \quad RR^{\top} = I_{\mathtt{q}-\mathtt{m}}. \end{aligned} \tag{LRA$_{c,R}$}$$

The Lagrangian of (LRA$_{c,R}$) is

$$L(\widehat{D}, c, R, \Lambda, \Xi) := \sum_{i=1}^{\mathtt{q}} \sum_{j=1}^{N} (d_{ij} - c_i - \widehat{d}_{ij})^2 + 2\,\text{trace}(R\widehat{D}\Lambda) + \text{trace}\left(\Xi(I - RR^{\top})\right).$$

Setting the partial derivatives of L to zero, we obtain the necessary optimality conditions

$$\begin{aligned} \partial L/\partial \widehat{D} = 0 \quad &\Longrightarrow \quad D - c\mathbf{1}_N^{\top} - \widehat{D} = R^{\top}\Lambda^{\top}, && \text{(L1)} \\ \partial L/\partial c = 0 \quad &\Longrightarrow \quad Nc = (D - \widehat{D})\mathbf{1}_N, && \text{(L2)} \\ \partial L/\partial R = 0 \quad &\Longrightarrow \quad \widehat{D}\Lambda = R^{\top}\Xi, && \text{(L3)} \\ \partial L/\partial \Lambda = 0 \quad &\Longrightarrow \quad R\widehat{D} = 0, && \text{(L4)} \\ \partial L/\partial \Xi = 0 \quad &\Longrightarrow \quad RR^{\top} = I. && \text{(L5)} \end{aligned}$$

The theorem follows from the system of equations (L1–L5). Next we list the derivation steps.

From (L3), (L4), and (L5), it follows that $\Xi = 0$ and from (L1), we obtain

$$D - \widehat{D} = c\mathbf{1}_N^{\top} + R^{\top}\Lambda^{\top}.$$

Substituting the last identity in (L2), we have

$$\begin{aligned} Nc = (c\mathbf{1}_N^{\top} + R^{\top}\Lambda^{\top})\mathbf{1}_N = Nc + R^{\top}\Lambda^{\top}\mathbf{1}_N \quad &\Longrightarrow \quad R^{\top}\Lambda^{\top}\mathbf{1}_N = 0 \\ &\Longrightarrow \quad \Lambda^{\top}\mathbf{1}_N = 0. \end{aligned}$$

Multiplying (L1) from the left by R and using (L4) and (L5), we have

$$R(D - c\mathbf{1}_N^{\top}) = \Lambda^{\top}. \tag{$*$}$$

Now, multiplication of the last identity from the right by $\mathbf{1}_N$ and use of $\Lambda^{\top}\mathbf{1}_N = 0$, shows that c is the row mean of the data matrix D,

$$R(D\mathbf{1}_N - Nc) = 0 \quad \Longrightarrow \quad c = \frac{1}{N} D\mathbf{1}_N.$$

Next, we show that $\widehat{D}$ is an optimal in a Frobenius norm rank-$\mathtt{m}$ approximation of $D - c\mathbf{1}_N^{\top}$. Multiplying (L1) from the right by Λ and using $\widehat{D}\Lambda = 0$, we have

$$(D - c\mathbf{1}_N^{\top})\Lambda = R^{\top}\Lambda^{\top}\Lambda. \tag{$**$}$$

Defining

$$\Sigma := \sqrt{\Lambda^{\top}\Lambda} \quad \text{and} \quad V := \Lambda\Sigma^{-1},$$

($*$) and ($**$) become

$$\begin{aligned} R(D - c\mathbf{1}_N^{\top}) = \Sigma V^{\top}, \quad V^{\top}V = I \\ (D - c\mathbf{1}_N^{\top})V = R^{\top}\Sigma, \quad RR^{\top} = I. \end{aligned}$$

The above equations show that the rows of R and the columns of V span, respectively, left and right $\mathtt{m}$-dimensional singular subspaces of the centered data matrix $D - c\mathbf{1}_N^{\top}$. The optimization criterion is minimization of

$$\|D - \widehat{D} - c\mathbf{1}_N^{\top}\|_{\mathrm{F}} = \|R^{\top}\Lambda^{\top}\|_{\mathrm{F}} = \sqrt{\text{trace}(\Lambda\Lambda^{\top})} = \text{trace}(\Sigma).$$

Therefore, a minimum is achieved when the rows of R and the columns of V span the, respectively left and right $\mathtt{m}$-dimensional singular subspaces of the centered data matrix $D - c\mathbf{1}_N^\top$, corresponding to the $\mathtt{m}$ smallest singular values. The solution is unique if and only if the $\mathtt{m}$th singular value is strictly bigger than the $(\mathtt{m}+1)$st singular value. Therefore, $\widehat{D}$ is a Frobenius norm optimal rank-$\mathtt{m}$ approximation of the centered data matrix $D - c\mathbf{1}_N^\top$, where $c = D\mathbf{1}_N/N$.

Proof of Corollary 8.5

$$\begin{aligned} c\mathbf{1}_N^\top + \widehat{D} &= c\mathbf{1}_N^\top + PL \\ &= c\mathbf{1}_N^\top + Pz\mathbf{1}_N^\top + PL - Pz\mathbf{1}_N^\top \\ &= \underbrace{(c+Pz)}_{c'}\mathbf{1}_N^\top + P\underbrace{(L - z\mathbf{1}_N^\top)}_{L'} = c'\mathbf{1}_N^\top + \widehat{D}' \end{aligned}$$

Therefore, if $(c, \widehat{D})$ is a solution, then $(c', \widehat{D}')$ is also a solution. From Theorem 8.4, it follows that $c = \mathbf{M}(D)$, $\widehat{D} = PL$ is a solution.

Proof of Theorem 8.6

Using the property $W_\mathrm{r}\mathbf{1}_N = \lambda\mathbf{1}_N$ of W_r, we have

$$\begin{aligned} \|D - \widehat{D} - c\mathbf{1}^\top\|_W &= \|\sqrt{W_\mathrm{l}}(D - \widehat{D} - c\mathbf{1}^\top)\sqrt{W_\mathrm{r}}\|_\mathrm{F} \\ &= \|D_\mathrm{m} - \widehat{D}_\mathrm{m} - c_\mathrm{m}\mathbf{1}^\top\|_\mathrm{F} \end{aligned}$$

where

$$D_\mathrm{m} = \sqrt{W_\mathrm{l}}D\sqrt{W_\mathrm{r}}, \quad \widehat{D}_\mathrm{m} = \sqrt{W_\mathrm{l}}\widehat{D}\sqrt{W_\mathrm{r}}, \quad \text{and} \quad c_\mathrm{m} = \sqrt{W_\mathrm{l}}c\sqrt{\lambda}.$$

Therefore, the considered problem is equivalent to the low-rank approximation problem (LRA_c) for the modified data matrix D_m.

Proof of Theorem 8.10

First, we show that the sequence $\widehat{D}^{(0)}, \widehat{D}^{(1)}, \ldots, \widehat{D}^{(k)}, \ldots$ converges monotonically in the Σ-weighted norm $\|\cdot\|_\Sigma$. On each iteration, Algorithm 10 solves two optimization problems (steps 1 and 2), which cost functions and constraints coincide with the ones of problem (C_0–C_5). Therefore, the cost function $\|D - \widehat{D}^{(k)}\|_\Sigma^2$ is monotonically nonincreasing. The cost function is bounded from below, so that the sequence

$$\|D - \widehat{D}^{(1)}\|_\Sigma^2, \quad \|D - \widehat{D}^{(2)}\|_\Sigma^2, \quad \ldots$$

is convergent. This proves ($f(k) \to f^*$).

Although, $\widehat{D}^{(k)}$ converges in norm, it may not converge element-wise. A sufficient condition for element-wise convergence is that the underlying optimization problem has a solution and this solution is unique, see (Kiers, 2002, Theorem 5). The element-wise convergence of $\widehat{D}^{(k)}$ and the uniqueness (due to the normalization condition (A_1)) of the factors $P^{(k)}$ and $L^{(k)}$, implies element-wise convergence of the factor sequences $P^{(k)}$ and $L^{(k)}$ as well. This proves ($D^{(k)} \to D^*$).

In order to show that the algorithm convergence to a minimum point of (C_0–C_5), we need to verify that the first order optimality conditions for (C_0–C_5) are satisfied at a cluster point of the algorithm. The algorithm converges to a cluster point if and only if the union of the first order optimality conditions for the problems on steps 1 and 2 are satisfied. Then

$$P'^{(k-1)} = P'^{(k)} =: P'^* \quad \text{and} \quad L'^{(k-1)} = L'^{(k)} =: L'^*.$$

From the above conditions for a stationary point and the Lagrangians of the problems of steps 1 and 2 and (C_0–C_5), it is easy to see that the union of the first order optimality conditions for the problems on steps 1 and 2 coincides with the first order optimality conditions of (C_0–C_5).

References

Chover J (1972) The green book of calculus. WA Benjamin

Kiers H (2002) Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems. Comput Stat Data Anal 41:157–170

Kline M (1974) Why Johnny Can't Add: The Failure of the New Math. Random House Inc

Pólya G (1957) How to Solve It. Doubleday

Vanluyten B, Willems JC, De Moor B (2006) Matrix factorization and stochastic state representations. In: Proc. 45th IEEE Conf. on Dec. and Control, San Diego, California, pp 4188–4193

Notation

Symbolism can serve three purposes. It can communicate ideas effectively; it can conceal ideas; and it can conceal the absence of ideas.

M. Kline

Sets of numbers

$\mathbb{R}$	the set of real numbers
$\mathbb{Z}$, $\mathbb{Z}_+$	the sets of integers and positive integers (natural numbers)

Norms and extreme eigen/singular values

$\|x\| = \|x\|_2,\ x \in \mathbb{R}^n$	vector 2-norm
$\|w\|,\ w \in (\mathbb{R}^{\mathtt{q}})^T$	signal 2-norm
$\|A\|,\ A \in \mathbb{R}^{m\times n}$	matrix induced 2-norm
$\|A\|_{\mathrm{F}},\ A \in \mathbb{R}^{m\times n}$	matrix Frobenius norm
$\|A\|_W, W \geq 0$	matrix weighted norm
$\|A\|_*$	nuclear norm
$\lambda(A), A \in \mathbb{R}^{m\times m}$	spectrum (set of eigenvalues)
$\lambda_{\min}(A)$, $\lambda_{\max}(A)$	minimum, maximum eigenvalue of a symmetric matrix
$\sigma_{\min}(A)$, $\sigma_{\max}(A)$	minimum, maximum singular value of a matrix

Matrix operations

$A^+, A^\top$	pseudoinverse, transpose
$\mathrm{vec}(A)$	column-wise vectorization
vec^{-1}	operator reconstructing the matrix A back from $\mathrm{vec}(A)$
$\mathrm{col}(a,b)$	the column vector $\left[\begin{smallmatrix} a \\ b \end{smallmatrix}\right]$
$\mathrm{col\,dim}(A)$	the number of block columns of A
$\mathrm{row\,dim}(A)$	the number of block rows of A
$\mathrm{image}(A)$	the span of the columns of A (the image or range of A)
$\ker(A)$	the null space of A (kernel of the function defined by A)
$\mathrm{diag}(v),\ v \in \mathbb{R}^n$	the diagonal matrix $\mathrm{diag}(v_1,\ldots,v_n)$
$\otimes$	Kronecker product $A \otimes B := [a_{ij}B]$
$\odot$	element-wise (Hadamard) product $A \odot B := [a_{ij}b_{ij}]$

Expectation, covariance, and normal distribution

$\mathbf{E}$, cov	expectation, covariance operator
$x \sim \mathrm{N}(m,V)$	x is normally distributed with mean m and covariance V

Fixed symbols

$\mathscr{B}$, $\mathscr{M}$	model, model class
$\mathscr{S}$	structure specification
$\mathscr{H}_i(w)$	Hankel matrix with *i block* rows, see ($\mathscr{H}_i$) on page 11
$\mathscr{T}_i(c)$	upper triangular Toeplitz matrix with *i block* rows, see ($\mathscr{T}$) on page 117
$\mathscr{R}(p,q)$	Sylvester matrix for the pair of polynomials p and q, see ($\mathscr{R}$) on page 11
$\mathscr{O}_i(A,C)$	extended observability matrix with i block-rows, see ($\mathscr{O}$) on page 52
$\mathscr{C}_i(A,B)$	extended controllability matrix with i block-columns, see ($\mathscr{C}$) on page 52

Linear time-invariant model class

$\mathtt{m}(\mathscr{B})$, $\mathtt{p}(\mathscr{B})$	number of inputs, outputs of $\mathscr{B}$
$\ell(\mathscr{B})$, $\mathtt{n}(\mathscr{B})$	lag, order of $\mathscr{B}$
$w\|_T$, $\mathscr{B}\|_T$	restriction of w, $\mathscr{B}$ to the interval $[1,T]$, see ($\mathscr{B}\|_T$) on page 53

$$\mathscr{L}^{\mathtt{q},\mathtt{n}}_{\mathtt{m},\ell} := \{\, \mathscr{B} \subset (\mathbb{R}^{\mathtt{q}})^{\mathbb{Z}} \mid \mathscr{B} \text{ is linear time-invariant with } \mathtt{m}(\mathscr{B}) \leq \mathtt{m},\ \ell(\mathscr{B}) \leq \ell, \text{ and } \mathtt{n}(\mathscr{B}) \leq \mathtt{n} \,\}$$

If $\mathtt{m}$, ℓ, or $\mathtt{n}$ are not specified, the corresponding invariants are not bounded.

Miscellaneous

$:= / =:$	left (right) hand side is defined by the right (left) hand side
$:\iff$	left-hand side is defined by the right-hand side
$\iff:$	right-hand side is defined by the left-hand side
σ^τ	the shift operator $(\sigma^\tau f)(t) = f(t+\tau)$
$\mathbf{i}$	imaginary unit
δ	Kronecker delta, $\delta_0 = 1$ and $\delta_t = 0$ for all $t \neq 0$
$\mathbf{1}_n = \left[\begin{smallmatrix} 1 \\ \vdots \\ 1 \end{smallmatrix}\right]$	vector with n elements that are all ones
$W \succ 0$	W is positive definite
$\lceil a \rceil$	rounding to the nearest integer greater than or equal to a

With some abuse of notation, the discrete-time signal, vector, and polynomial

$$\big(w(1),\ldots,w(T)\big) \quad\leftrightarrow\quad \mathrm{col}\big(w(1),\ldots,w(T)\big) \quad\leftrightarrow\quad z^1 w(1) + \cdots + z^T w(T)$$

are all denoted by w. The intended meaning is understood from the context.

Index

Acknowledgements

A number of individuals and funding agencies supported me during the preparation of the book. Oliver Jackson—Springer's editor engineering—encouraged me to embark on the project and to prepare this second edition of the book. My colleagues at ESAT/STADIUS (formally SISTA), K.U.Leuven, ECS/VLC (formerly ISIS), University of Southampton, and the department ELEC, Vrije Universiteit Brussels created the right environment for developing the ideas in the book. In particular, I am in debt to Jan C. Willems for his personal guidance and example of critical thinking. The behavioral approach that Jan initiated in the early 1980's is present in this book.

M. De Vos, D. Sima, K. Usevich, and J. C. Willems proofread chapters of the first edition and suggested improvements. I gratefully acknowledge funding from:

- the European Research Council (ERC) under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC Grant agreement number 258581 "Structured low-rank approximation: Theory, algorithms, and applications";
- the Fund for Scientific Research (FWO-Vlaanderen), FWO projects G028015N "Decoupling multivariate polynomials in nonlinear system identification" and G090117N "Block-oriented nonlinear identification using Volterra series";
- the Belgian Science Policy Office Interuniversity Attraction Poles Programme, network on "Dynamical Systems, Control, and Optimization" (DYSCO); and
- the FWO/FNRS Excellence of Science project "Structured low-rank matrix / tensor approximation: numerical optimization-based algorithms and applications".