SUBSPACE SYSTEM IDENTIFICATION
Theory and applications

Lecture notes

Dr. Ing.
David Di Ruscio

University of South Eastern Norway
Email: david.di.ruscio@usn.no
Porsgrunn, Norway

February 2022

6th edition **February 9, 2022**

University of South Eastern Norway
Kjølnes Ring 56
N-3914 Porsgrunn, Norway

Preface

The material in this report/book is meant to be used as lecture notes in practical and theoretical system identification and the focus is on so called subspace system identification and in particular the DSR algorithms. Some of the material and some chapters are based on published papers on the topic organized in a proper manner.

On central topic is a detailed description of the method for system identification of combined Deterministic and Stochastic systems and Realization (DSR), which is a subspace system identification method which may be used to identify a complete Kalman filter model directly from known input and output data, including the system order. Several special methods and variants of the DSR method may be formulated in order to be used for the identification of special systems, e.g., deterministic systems, stochastic systems, closed loop system identification etc.

Furthermore basic theory as realization theory for dynamic systems based on known markov parameters (impulse response matrices), practical topics as effect of scaling, how to treat trends in the data, selecting one model from different models based on model validation, ordinary least squares regression, principal component regression as well as partial least squares regression, etc.

Parts of the material is used in the Mater Course, SCE2206 System Identification and Optimal Estimation at Telemark University College.

The material is also believed to be useful for students working with main thesis on the subject and some of the material may also be used on a PhD level. The material may also be of interest for the reader interested in system identification of dynamic systems in general.

Contents

1 Preliminaries **1**

1.1 State space model 1

1.2 Inputs and outputs 1

1.3 Data organization 2

1.3.1 Hankel matrix notation 2

1.3.2 Extended data vectors 3

1.3.3 Extended data matrices 4

1.4 Definitions 4

1.5 Extended output equation 5

1.5.1 Extended vector output equation 5

1.5.2 Extended output matrix equation 6

1.6 Observability 6

1.7 Extended state space model 8

1.8 Controllability 9

1.9 ARMAX and extended state space model 10

2 Realization theory **13**

2.1 Introduction 13

2.2 Deterministic case 13

2.2.1 Model structure and problem description 13

2.2.2 Impulse response model 14

2.2.3 Determination of impulse responses 14

2.2.4 Redundant or missing information 17

2.2.5 The Hankel matrices 18

2.2.6 Balanced and normalized realizations 20

2.2.7 Error analysis 22
2.3 Numerical examples 23
2.3.1 Example 1 23
2.4 Concluding remarks 23

3 Combined *D*eterministic and *S*tochastic System Identification and *R*ealization - *DSR*: A subspace approach based on observations [1] 27

3.1 Introduction 28
3.2 Preliminary Definitions 29
3.2.1 System Definition 29
3.2.2 Problem Definition 29
3.2.3 Matrix Definitions 30
3.2.4 Notation 31
3.3 Extended State Space Model 31
3.4 System Identification and Realization 32
3.4.1 Identification and Realization of System Dynamics 32
3.4.2 Realization of the Deterministic Sub-system 37
3.4.3 Realization of the Stochastic Sub-system 39
3.5 Implementation with QR Decomposition 44
3.5.1 Realization of A and D 45
3.5.2 Realization of B and E 46
3.5.3 Realization of C and Δ 47
3.5.4 Special Remarks 49
3.6 Comparison with Existing Algorithms 50
3.6.1 PO-MOESP 51
3.6.2 Canonical Variate Analysis (CVA) 52
3.6.3 N4SID 53
3.6.4 Main Differences and Similarities 54
3.7 Numerical Examples 55
3.7.1 Example 1: Monte Carlo Simulation 55
3.7.2 Example 2 63

[1] ECC95 paper extended with proofs, new results and theoretical comparison with existing subspace identification methods. Also in *Computer Aided Time Series Modeling*, Edited by Masanao Aoki, Springer Verlag, 1997.

3.7.3 Example 3 63

3.7.4 Example 4: Modeling the crude oil prize 66

3.7.5 Example 5: Modeling PH 72

3.8 Conclusions 72

4 On the DSR algorithm 77

4.1 Introduction 77

4.2 BASIC SYSTEM THEORETIC DESCRIPTION 79

4.3 PROBLEM DESCRIPTION 80

4.4 BASIC DEFINITIONS 81

4.5 ORTHOGONAL PROJECTIONS 82

4.6 MATRIX DEFINITIONS 83

4.7 BASIC MATRIX EQUATION IN SUBSPACE IDENTIFICATION 84

4.8 MATRIX EQUATION WITH STATES ELIMINATED 85

4.9 MATRIX OF INSTRUMENTAL VARIABLES W 86

4.10 Subspace identification of O_L: autonomous systems 87

4.11 Subspace identification of O_L: deterministic systems 88

4.12 Subspace identification of O_{L+1}: “general” case 89

4.13 BASIC PROJECTIONS IN THE DSR ALGORITHM 90

4.14 COMPUTING PROJECTIONS BY QR DECOMPOSITION . . 91

4.15 COMPUTING PROJECTIONS WITH PLS 94

4.16 MODEL ORDER SELECTION BY EXAMPLE 95

4.17 PROPERTIES OF THE ALGORITHM 96

4.18 COMPARISON WITH CLASSICAL APPROACHES 97

4.19 INDUSTRIAL APPLICATIONS 98

4.20 SOFTWARE 99

4.21 Analysis and modeling of a scrubber 100

4.22 MONTE CARLO SIMULATION OF MIMO SYSTEM 101

4.23 STATISTICAL ANALYSIS 103

4.24 IDENTIFICATION AND CONTROL OF TMP PROCESS . . . 104

4.25 References 109

5 Subspace Identification for Dynamic Process Analysis and Modeling 115

5.1 Introduction 115

5.2 Preliminary definitions 116

5.2.1 System definition 116

5.2.2 Problem definition 117

5.3 Description of the method 117

5.4 Modeling the power 118

5.4.1 Refiner variables and model 119

5.4.2 Numerical results 120

5.5 Conclusion 121

6 Dynamic Subspace Modeling (DSR) and Static Multivariate Analysis and Regression (PCA,PCR,PLS) 129

6.1 Introduction 129

6.2 System description and data organization 131

6.2.1 Combined deterministic and stochastic description 131

6.2.2 Purely static description 132

6.3 Dependent input or output data 134

6.4 The combined deterministic and stochastic problem 135

6.4.1 Data compression 135

6.4.2 Identification of system dynamics 135

6.5 The static problem 138

6.5.1 Data compression 138

6.5.2 The residual 139

6.5.3 Effective rank analysis 139

6.5.4 Score and loading vector matrices 140

6.5.5 Regression 140

6.6 Concluding remarks 141

7 On Subspace Identification of the Extended Observability Matrix 143

7.1 Introduction 143

7.2 Definitions 143

7.2.1 Notation 143

7.3 Relationship between past and future data matrices 144

7.4 The observability matrix 144
7.5 Standard deviation of the estimates 148
7.6 PLS for computing the projections 148
7.7 Numerical example 150
7.8 Conclusion 151
7.9 References 151
7.10 Appendix: proofs 153
7.10.1 Proof of Equation (7.4) 153
7.10.2 Proof I of Equation (7.10) 154
7.10.3 Proof II of Equation (7.10) 155
7.10.4 Proof of Equation (7.22) 156
7.10.5 Including constraints: regularization 157
7.10.6 Some useful results 158
7.10.7 Proof III of Equation (7.10) 159
7.10.8 Prediction of future outputs 159

8 Subspace identification of the states and the Kalman filter gain161

8.1 Introduction 161
8.2 Notation and definitions 162
8.2.1 System and matrix definitions 162
8.2.2 Hankel matrix notation 163
8.2.3 Projections 164
8.3 Subspace system identification 165
8.3.1 Subspace identification of the states 165
8.3.2 The extended observability matrix 171
8.3.3 Identification of the stochastic subsystem 174
8.3.4 SID of the deterministic subsystem 177
8.4 Closed loop subspace identification 179
8.4.1 Closed loop subspace identification: Using a filter in the feeback loop! 180
8.4.2 Closed loop subspace identification: Using the controller . 182
8.4.3 Closed loop subspace identification: Indirect method . . . 183
8.4.4 Closed loop subspace identification: Direct method 185

8.5 A new subspace identification method for closed and open loop systems . . . 186
8.6 Further remarks . . . 188
8.6.1 Choice of algorithm parameters . . . 188
8.6.2 Choice of input signal . . . 188
8.6.3 N4SID . . . 189
8.7 Numerical examples . . . 189
8.7.1 Example 1 . . . 189
8.7.2 Example 2 . . . 190
8.7.3 Example 3 . . . 192
8.7.4 Example 4 . . . 193
8.7.5 Example 5 . . . 195
8.8 Conclusion . . . 196

9 Effect of Scaling and how to Treat Trends 201
9.1 The data . . . 201
9.2 The data matrices . . . 201
9.3 Scaling the data matrices . . . 202
9.3.1 Proof of Step 3 in Algorithm 9.3.1 . . . 203
9.3.2 Numerical Examples . . . 204
9.4 How to handle trends and drifts . . . 204
9.5 Trends and low frequency dynamics in the data . . . 207
9.6 Time varying trends . . . 208
9.7 Constant trends . . . 209

10 Validation 211
10.1 Model validation and fit . . . 211
10.1.1 Criteria based on the simulated error . . . 211
10.1.2 Criteria based on the prediction error . . . 212

11 Input experiment design 213
11.1 Experiment design for dynamic systems . . . 213

12 Special topics in system identification 219
12.1 Optimal predictions . . . 219

12.2 Constraints in the subspace identification problem 220

12.3 Least squares with equality constraints 220

13 On the Partial Least Squares Algorithm 223

13.1 Notation, basic- and system-definitions 223

13.2 Partial Least Squares 224

13.2.1 Interpretation of the Partial Least Squares algorithm . . . 224

13.2.2 The Partial Least Squares decomposition 235

13.2.3 The Partial Least Squares regression and prediction . . . 237

13.2.4 Computing projections from the PLS decomposition . . . 239

13.2.5 A semi standard PLS implementation 240

13.2.6 The number of components a 240

13.3 Further interpretations of the PLS algorithm 241

13.3.1 General results 241

13.3.2 Special case results 242

13.4 A covariance based PLS implementation 243

13.5 A square root PLS algorithm 244

13.6 Undeflated PLS 245

13.7 Total Least Squares and Truncated Total Least Squares 246

13.7.1 Total Least Squares 246

13.7.2 Truncated Total Least Squares 247

13.8 Performance and PLS 248

13.9 Optimal solution 248

13.10Numerical examples 249

13.11Conclusion 250

13.12Appendix: proofs 251

13.12.1Proof of Equation (13.115) 251

13.12.2The PLS algorithm in terms of the W weight matrix . . . 252

13.13Matlab code: The PLS1 algorithm 254

13.14Matlab code: The PLS2 algorithm 255

13.15Matlab code: The PLS3 algorithm 256

A Proof of the ESSM 257

A.1 Proof of the ESSM 257

A.2 Discussion 258

A.3 Selector matrices 258

A.4 A study of the predictor $Z^d_{J|L+1}$ for $Y_{J|L+1}$. 259

B Linear Algebra and Matrix Calculus 261

B.1 Trace of a matrix 261

B.2 Gradient matrices 261

B.3 Derivatives of vector and quadratic form 262

B.4 Matrix norms 262

B.5 Linearization 263

B.6 Kronecer product matrices 264

C Optimization of the DSR algorithm 265

C.1 Existence of an optimal solution 265

C.2 Optimized algorithm for computing the B and E matrices 266

D D-SR Toolbox for MATLAB 271

D.1 The DSR function for use with MATLAB 272

Chapter 1

Preliminaries

1.1 State space model

Consider a process which can be described by the following linear, discrete time invariant state space model (SSM)

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Cv_k & (1.1) \\ y_k &= Dx_k + Eu_k + Fv_k & (1.2) \end{aligned}$$

where the integer $k \geq 0$ is discrete time, $x_k \in \mathbb{R}^n$ is the state vector, $u_k \in \mathbb{R}^r$ is the input vector, $v_k \in \mathbb{R}^l$ is an external input vector and $y_k \in \mathbb{R}^m$ is the output vector. The constant matrices in the SSM are of appropriate dimensions. A is the *state transition matrix*, B is the *input* matrix, C is the *external input* matrix, D is the *output matrix* and E is the direct *input* to output matrix and F is the direct *external input to output* matrix.

The following assumptions are stated:

- The pair (D, A) is observable.
- The pair $(A, \left[\begin{array}{cc} B & C \end{array}\right])$ is controllable.

In some cases it is not desired to identify the direct input to output matrix E. If E is apriori known to be zero then usually the other model matrices can be identified with higher accuracy if E is not estimated. For this reason, define the integer structure parameter g as follows

$$g \overset{\text{def}}{=} \left\{ \begin{array}{lll} 1 & \text{when} & E \neq 0_{m \times r} \\ 0 & \text{when} & E = 0_{m \times r} \end{array} \right. \tag{1.3}$$

1.2 Inputs and outputs

First of all note that the definition of an input vector u_k in the system identification theory is different from the vector of manipulable inputs u_k in control theory. For example as quoted by Ljung (1995) page 61 and Ljung (1999) page 523. *Note that the inputs need not at all be control signals: anything measurable, including disturbances, should be treated as input signals.*

Think over the physics and use all significant measurement signals which is available as input signals in order to model the desired output signals. With significant measurement signals we mean input variables that describes and observes some of the effects in the outputs.

For example if one want to model the room temperature, one usually gets better fit when both the heat supply and the outside temperature is used as input signals, than only using the heat supply, which in this case is an manipulable input variable.

An example from the industry of using measured output signals and manipulable variables in the input vector u_k is presented in Di Ruscio (1994).

Hence, input signals in system identification can be manipulable input variables as well as non-manipulable measured signals, including measurements, measured states, measured properties, measured disturbances etc. The measurements signals which are included in the input signal may of course be corrupted with noise. Even feedback signals, i.e., input signals which are a function of the outputs, may be included as inputs.

An important issue is to ask what the model should be used for. Common problems are to:

- Identify models for control.
  Important aspects in this case is to identify the dynamics and behaviour from the manipulable input variables and possibly measured disturbances upon the controlled output variables. The user must be careful to include possibly measured states as input signals in this case. The reason for this is that loss of identifibility of the behaviour from the manipulable inputs to the outputs may be the result.
- Identify models for prediction of property and quality variables.
  Generally speaking, the prediction and fit gets better when more measured signals are included as inputs and worse when more outputs are added.
- Identify filters.
  This problem is concerned with extraction of signals from noisy measurements.

1.3 Data organization

1.3.1 Hankel matrix notation

Hankel matrices are frequently used in realization theory and subspace system identification. The special structure of a Hankel matrix as well as some notations, which are frequently used throughout, are defined in the following.

Definition 1.1 (Hankel matrix) *Given a (vector or matrix) sequence of data*

$$s_t \;\in\; \mathbb{R}^{nr\times nc} \;\forall\; 0,1,2,\ldots,t_0,t_0+1,\ldots, \tag{1.4}$$

where nr is the number of rows in s_t and nc is the number of columns in s_t.

Define integer numbers t_0, L and K and define tne the matrix S_t as follows

$$S_{t_0|L} \overset{def}{=} \begin{bmatrix} s_{t_0} & s_{t_0+1} & s_{t_0+2} & \cdots & s_{t_0+K-1} \\ s_{t_0+1} & s_{t_0+2} & s_{t_0+3} & \cdots & s_{t_0+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{t_0+L-1} & s_{t_0+L} & s_{t_0+L+1} & \cdots & s_{t_0+L+K-2} \end{bmatrix} \in \mathbb{R}^{Lnr\times Knc}. \tag{1.5}$$

which is defined as a Hankel matrix because of the special structure. The integer numbers t_0, L and K are defined as follows:

- *t_0 start index or initial time in the sequence s_{t_0} which is the upper left block in the Hankel matrix.*
- *L is the number of nr-block rows in $S_{t_0|L}$.*
- *K is the number of nc-block columns in $S_{t_0|L}$.*

A Hankel matrix is symmetric and the elements are constant across the anti-diagonals. We are usually working with vector sequences in subspace system identification, i.e., s_t is a vector in this case and hence,$nc = 1$.

Example 1.1 *Given a vector valued sequence of observations*

$$s_t \;\in\; \mathbb{R}^{nr} \;\forall\; 1,2,\ldots,N \tag{1.6}$$

with $N = 10$. Choose parameters $t_0 = 3$, $L = 2$ and use all observations to define the Hankel data matrix $S_{3|2}$. We have

$$S_{3|2} = \begin{bmatrix} s_3 & s_4 & s_5 & s_6 & s_7 & s_8 & s_9 \\ s_4 & s_5 & s_6 & s_7 & s_8 & s_9 & s_{10} \end{bmatrix} \in \mathbb{R}^{Lnr\times 7}. \tag{1.7}$$

The number of columns is in this case $K = N - t_0 = 7$.

1.3.2 Extended data vectors

Given a number L known output vectors and a number $L + g$ known input vectors. The following extended vector definitions can be made

$$y_{k|L} \overset{\text{def}}{=} \begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+L-1} \end{bmatrix} \in \mathbb{R}^{Lm} \tag{1.8}$$

where L is the number of block rows. The extended vector $y_{k|L}$ is defined as the *extended state space* vector. The justification for the name is that this vector satisfy an *extended state space* model which will be defined later.

$$u_{k|L+g} \stackrel{\text{def}}{=} \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+L+g-2} \\ u_{k+L+g-1} \end{bmatrix} \in \mathbb{R}^{(L+g)m} \tag{1.9}$$

where $L+g$ is the number of block rows. The extended vector $u_{k|L+g}$ is defined as the *extended input vector*.

1.3.3 Extended data matrices

Define the following output data matrix with L block rows and K columns.

$$Y_{k|L} \stackrel{\text{def}}{=} \overbrace{\begin{bmatrix} y_k & y_{k+1} & y_{k+2} & \cdots & y_{k+K-1} \\ y_{k+1} & y_{k+2} & y_{k+3} & \cdots & y_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{k+L-1} & y_{k+L} & y_{k+L+1} & \cdots & y_{k+L+K-2} \end{bmatrix}}^{\textbf{Known} \text{ data matrix of output variables}} \in \mathbb{R}^{Lm\times K} \tag{1.10}$$

Define the following input data matrix with $L+g$ block rows and K columns.

$$U_{k|L+g} \stackrel{\text{def}}{=} \overbrace{\begin{bmatrix} u_k & u_{k+1} & u_{k+2} & \cdots & u_{k+K-1} \\ u_{k+1} & u_{k+2} & u_{k+3} & \cdots & u_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{k+L+g-2} & u_{k+L+g-1} & u_{k+L+g} & \cdots & u_{k+L+K+g-3} \\ u_{k+L+g-1} & u_{k+L+g} & u_{k+L+g+1} & \cdots & u_{k+L+K+g-2} \end{bmatrix}}^{\textbf{Known} \text{ data matrix of input variables}} \in \mathbb{R}^{(L+g)r\times K} \tag{1.11}$$

1.4 Definitions

Associated with the SSM, Equations (1.1) and (1.2), we make the following definitions:

- The *extended observability* matrix ($\mathcal{O}_i$) for the pair (D, A) is defined as

$$\mathcal{O}_i \stackrel{\text{def}}{=} \begin{bmatrix} D \\ DA \\ \vdots \\ DA^{i-1} \end{bmatrix} \in \mathbb{R}^{im\times n} \tag{1.12}$$

  where the subscript i denotes the number of block rows.

- The *reversed extended controllability* matrix $\mathcal{C}_i^d$ for the pair (A, B) is defined as

$$\mathcal{C}_i^d \overset{\text{def}}{=} \left[\begin{array}{cccc} A^{i-1}B & A^{i-2}B & \cdots & B \end{array} \right] \quad \in \mathbb{R}^{n\times ir} \tag{1.13}$$

where the subscript i denotes the number of block columns.

- A *reversed extended controllability* matrix $\mathcal{C}_i^s$ for the pair (A, C) is defined similar to Equation (8.4),

$$\mathcal{C}_i^s \overset{\text{def}}{=} \left[\begin{array}{cccc} A^{i-1}C & A^{i-2}C & \cdots & C \end{array} \right] \quad \in \mathbb{R}^{n\times il} \tag{1.14}$$

i.e., with B substituted with C in Equation (8.4.

- The *lower block triangular Toeplitz* matrix $(\mathcal{H}_i^d)$ for the quadruple matrices (D, A, B, E)

$$\mathcal{H}_i^d \overset{\text{def}}{=} \begin{bmatrix} E & 0 & 0 & \cdots & 0 \\ DB & E & 0 & \cdots & 0 \\ DAB & DB & E & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}B & DA^{i-3}B & DA^{i-4}B & \cdots & E \end{bmatrix} \quad \in \mathbb{R}^{im\times(i+g-1)r} \tag{1.15}$$

where the subscript i denotes the number of block rows and $i + g - 1$ is the number of block columns.

- A *lower block triangular Toeplitz* matrix $\mathcal{H}_i^s$ for the quadruple (D, A, C, F) is defined as

$$\mathcal{H}_i^s \overset{\text{def}}{=} \begin{bmatrix} F & 0 & 0 & \cdots & 0 \\ DC & F & 0 & \cdots & 0 \\ DAC & DC & F & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}C & DA^{i-3}C & DA^{i-4}C & \cdots & F \end{bmatrix} \quad \in \mathbb{R}^{im\times il} \tag{1.16}$$

1.5 Extended output equation

1.5.1 Extended vector output equation

The state space model, Equations (1.1) and (1.2), can in general be written as the following *extended vector* output equation

$$y_{k|L} = \mathcal{O}_L x_k + \mathcal{H}_L^d u_{k|L+g-1} + \mathcal{H}_L^s v_{k|L} \tag{1.17}$$

Proof: Stack L outputs defined by Equation (1.2) and substitute for the state vector given by Equation (1.1). **QED**.

1.5.2 Extended output matrix equation

The extended vector output equation can in general be written as the following *extended output matrix* equation

$$Y_{k|L} = \mathcal{O}_L X_k + \mathcal{H}_L^d U_{k|L+g-1} + \mathcal{H}_L^s E_{k|L} \tag{1.18}$$

where

$$X_k \overset{\text{def}}{=} \begin{bmatrix} x_k & x_{k+1} & \cdots & x_{k+K-1} \end{bmatrix} \quad \in \mathbb{R}^{n \times K} \tag{1.19}$$

Proof: The proof follows from the *extended vector output* equation (1.17). **QED**.

1.6 Observability

Theorem 1.6.1 (Observability)
The system (1.1) and (1.2) is completely observable if the extended observability matrix $\mathcal{O}_L \in \mathbb{R}^{Lm \times n}$ *defined by Equation (8.3) satisfy*

$$rank(\mathcal{O}_L) = n$$

for all numbers L *of block rows given by*

$$L \geq L_{\min} \overset{\text{def}}{=} \begin{cases} n - rank(D) + 1 & when \quad m < n \\ 1 & when \quad m \geq n \end{cases} \tag{1.20}$$

△

Proof:
We have from equation (1.17) that

$$\mathcal{O}_L x_k = y_{k|L} - \mathcal{H}_L^d u_{k|L+g-1} - \mathcal{H}_L^s v_{k|L}$$

Assume that the extended observability matrix has full rank. The state vector can then be computed from

$$x_k = (\mathcal{O}_L^T \mathcal{O}_L)^{-1} \mathcal{O}_L^T (y_{k|L} - \mathcal{H}_L^d u_{k|L+g-1} - \mathcal{H}_L^s v_{k|L})$$

if rank$(\mathcal{O}_L) = n$ and the right hand side is known. If rank$(\mathcal{O}_L) < n$ then only a part of the state vector (x_k) can be observed. **QED**.

We will define $\mathcal{O}_L$ as the *extended observability* matrix when the number of block rows satisfy $L > L_{min}$ and only the sl observability matrix when the number of block rows is $L = L_{min}$.

The row size of the extended observability matrix $\mathcal{O}_L$ is Lm. The minimal number of block rows $L_{\min}$ in the multiple output case follows from an argumentation for when $Lm \geq n$ and for when rank$(\mathcal{O}_L) = n$. In the single output case with ($m = \text{rank}(D) = 1$). Then we must have $L \geq n$ in order for $\mathcal{O}_L$ not to loose rank.

The lower bound for the number of block rows in the observability matrix is the ceiling function of n/m which is defined as the integer rounded towards $+\infty$ of the integer to integer ratio n/m. The upper bound must be n due to the Cayley-Hamilton theorem. Hence, for an observable system the minimum number of block rows in the observability matrix is bounded by

$$\lceil \frac{n}{m} \rceil \leq L_{\text{min}} \leq n \tag{1.21}$$

It can be shown, Kalman, Falb and Arbib (1969), Ch.2, p. 38, that the rank of the matrix $\mathcal{O}_L$ must increase by at least one (maximum m) as each block row is added to the observability matrix, until the maximal rank (n) is reached. If $\text{rank}(D) = m$ then it will be *sufficient* to include at most $n-m$ block rows DA, $\cdots$, DA^{n-m} to see whether the maximal rank of $\mathcal{O}_L$ can reach n. Hence, for an observable system with $\text{rank}(D) = m$ it is sufficient with

$$L_{\text{min}} = n - m + 1$$

block rows in the observability matrix to ensure that $\text{rank}(\mathcal{O}_{L_{\text{min}}}) = n$.

Finally to see that $L_{\text{min}}m \geq n$ consider the multiple output case when $m = \text{rank}(D)$ and $m \leq n$. From Theorem 1.6.1 we must have

$$\begin{gathered} L_{\text{min}}m = (n - m + 1)m \geq n \\ \Downarrow \\ (n - m)(m - 1) \geq 0 \end{gathered}$$

which prove that $L_{min}m \geq n$ when $m \leq n$.

Example 1.2 (Observability)
Given a system with $m = 2$ and $n = 4$

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{1.22}$$

For this system the rank of the observability matrix $\mathcal{O}_3 = [D; DA; DA^2]$ will increase by one each time the block row DA and DA^2 is added until the maximal rank 4 is reached, i.e., $L_{min} = 3$ for this system. Changing elements in A so that $a_{14} = 1$ and $a_{24} = 1$, i.e., for

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{1.23}$$

gives the necessary number of block rows, L, equal to only $\lceil n/m \rceil$=2. This means that rank($\mathcal{O}_2$) = 3 in this last case. $\lhd$

1.7 Extended state space model

An *extended state space model* (ESSM) will be defined in this section.

The importance of the ESSM for subspace system identification is that it gives us an equation where the states is eliminated. It leads us to an algorithm for which the system model can be recovered from the known input and output data. The importance of the ESSM for model predictive control is that it facilitates a simple and general method for building a *prediction model.*

Theorem 1.7.1 (Extended state space model)
The SSM, Equations (1.1) and (1.2), can be described with an equivalent extended state space model (ESSM)

$$y_{k+1|L} = \tilde{A}_L y_{k|L} + \tilde{B}_L u_{k|L+g} + \tilde{C}_L v_{k|L+1} \tag{1.24}$$

where the extended state space vector $y_{k|L}$ *and the extended input vector* $u_{k|L+g}$ *are defined in equations (1.8) and (1.9). The extended external input vector* $v_{k|L+1}$ *is defined similarly.*

The integer number L *which define the number of block rows in the extended state vector must satisfy*

$$L \geq L_{min}$$

where the minimal number of block rows is defined by

$$L_{min} \overset{\text{def}}{=} \begin{cases} n - rank(D) + 1 & when \quad m < n \\ 1 & when \quad m \geq n \end{cases} \tag{1.25}$$

The matrices in the ESSM are given as

$$\begin{aligned}
\tilde{A}_L &\overset{\text{def}}{=} \mathcal{O}_L A (\mathcal{O}_L^T \mathcal{O}_L)^{-1} \mathcal{O}_L^T && \in \mathbb{R}^{Lm \times Lm} && (1.26) \\
\tilde{B}_L &\overset{\text{def}}{=} \begin{bmatrix} \mathcal{O}_L B & \mathcal{H}_L^d \end{bmatrix} - \tilde{A}_L \begin{bmatrix} \mathcal{H}_L^d & 0_{Lm \times r} \end{bmatrix} && \in \mathbb{R}^{Lm \times (L+g)r} && (1.27) \\
\tilde{C}_L &\overset{\text{def}}{=} \begin{bmatrix} \mathcal{O}_L C & \mathcal{H}_L^s \end{bmatrix} - \tilde{A}_L \begin{bmatrix} \mathcal{H}_L^s & 0_{Lm \times l} \end{bmatrix} && \in \mathbb{R}^{Lm \times (L+1)l} && (1.28)
\end{aligned}$$

△

Proof: See Di Ruscio (1994).

The ESSM has some properties:

1. **Minimal ESSM order**. The minimal ESSM order L_{min}, defined in Equation (1.25), defines the minimal *prediction horizon.*
   Proof: Same as proving observability, in this case $(\mathcal{O}_{L_{min}}^T \mathcal{O}_{L_{min}})^{-1}$ is non-singular and the existence of the ESSM can be proved.

2. **Uniqueness**. Define M as a non-singular matrix which transform the state vector x to a new coordinate system. Given two system realizations, (A, B, C, D, F) and $(M^{-1}AM, M^{-1}B, M^{-1}C, DM, F)$. The two

realizations gives the same ESSM matrices for all $L \geq L_{min}$.
Proof: The ESSM matrices are invariant under state (coordinate) transformations in the SSM.

3. **Eigenvalues**. The ESSM transition matrix ($\tilde{A}_L$) have the same (n) eigenvalues as the SSM transition matrix (A) and $Lm - n$ eigenvalues equal to zero.
Proof: From similarity, Equation 8.91.

In the subspace identification algorithm, Di Ruscio (1995), it is shown that the ESSM can be identified directly from a sliding window of known data. However, when the inputs used for identification are poor from a persistent excitation point of view, it is better to only identify the minimal order ESSM as explained in the paper. Note also that L_{min} coincides with the demand for persistent excitation of the inputs, i.e., the inputs must at least be persistent exiting of order L_{min} in order to recover the SSM from known input and output data.

Assume that a minimal order ESSM is given and defined as follows

$$y_{k+1|L_{min}} = \tilde{A}_{L_{min}} y_{k|L_{min}} + \tilde{B}_{L_{min}} u_{k|L_{min}} + \tilde{C}_{L_{min}} v_{k|L_{min}+1} \tag{1.29}$$

If the *prediction horizon* is chosen greater than L_{min} it is trivial to construct the following ESSM from the minimal ESSM.

By using $y_{k+1|L-L_{min}} = y_{k+1|L-L_{min}}$ in addition to Equation (1.29) we obtain the following ESSM model matrices

$$\tilde{A}_L = \begin{bmatrix} 0_{(L-L_{min})m\times m} & I_{(L-L_{min})m\times(L-1)m} \\ 0_{L_{min}m\times(L-L_{min})m} & \tilde{A}_{L_{min}} \end{bmatrix} \tag{1.30}$$

$$\tilde{B}_L = \begin{bmatrix} 0_{(L-L_{min})m\times Lr} & \\ 0_{L_{min}m\times(L-L_{min})r} & \tilde{B}_{L_{min}} \end{bmatrix} \tag{1.31}$$

$$\tilde{C}_L = \begin{bmatrix} 0_{(L-L_{min})m\times(L+1)l} & \\ 0_{L_{min}m\times(L-L_{min})l} & \tilde{C}_{L_{min}} \end{bmatrix} \tag{1.32}$$

This last formulations of the ESSM matrices is attractive from an identification point of view.

It will also have some properties with respect to noise retention compared to using the matrices defined by Equations (8.91) to (8.93) directly in the model predictive control algorithms which will be presented in a later chapter.

1.8 Controllability

Theorem 1.8.1 (Controllability)
The linear system (1.1) described by the pair (A, B) is completely controllable if the extended controllability matrix $\mathcal{C}_J^d \in \mathbb{R}^{n\times Jr}$ defined by Equation (8.4) satisfy

$$rank(\mathcal{C}_J^d) = n,$$

for all numbers, J, of block rows given by

$$J \geq J_{\min} \overset{\text{def}}{=} \begin{cases} n - \mathit{rank}(B) + 1 & \mathit{when} \quad r < n, \\ 1 & \mathit{when} \quad r \geq n. \end{cases} \tag{1.33}$$

$\triangle$

1.9 ARMAX and extended state space model

Given a polynomial (ARMAX) model in the discrete time domain as follows

$$y_{k+L|1} = \tilde{A}_1 y_{k|L} + \tilde{B}_1 u_{k|L+g} + \tilde{C}_1 v_{k|L+1} \tag{1.34}$$

where the polynomial (ARMAX) model matrices are given by

$$\begin{aligned} \tilde{A}_1 &\overset{\text{def}}{=} \begin{bmatrix} A_1 & A_2 & \cdots & A_L \end{bmatrix} && \in \mathbb{R}^{m \times Lm}, \\ \tilde{B}_1 &\overset{\text{def}}{=} \begin{bmatrix} B_0 & B_1 & \cdots & B_{L-1+g} & B_{L+g} \end{bmatrix} && \in \mathbb{R}^{m \times (L+g)r}, \\ \tilde{C}_1 &\overset{\text{def}}{=} \begin{bmatrix} C_0 & C_1 & \cdots & C_L & C_{L+1} \end{bmatrix} && \in \mathbb{R}^{m \times (L+1)m}. \end{aligned} \tag{1.35}$$

where $A_i \in \mathbb{R}^{m \times m}$ for $i = 1, \ldots, L$, $B_i \in \mathbb{R}^{m \times r}$ for $i = 0, \ldots, L+g$, $C_i \in \mathbb{R}^{m \times m}$ for $i = 0, \ldots, L+1$. This model can be formulated directly as an ESSM which have relations to the SSM matrices. We have

$$\tilde{A}_L = \begin{bmatrix} 0 & I & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & I \\ A_1 & A_2 & \cdots & A_{L-1} & A_L \end{bmatrix} \in \mathbb{R}^{Lm \times Lm} \tag{1.36}$$

$$\tilde{B}_L = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ B_0 & B_1 & \cdots & B_{L+g-2} & B_{L+g-1} & B_{L+g} \end{bmatrix} \in \mathbb{R}^{Lm \times (L+g)r} \tag{1.37}$$

$$\tilde{C}_L = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ C_0 & C_0 & \cdots & C_{L-1} & C_L & C_{L+1} \end{bmatrix} \in \mathbb{R}^{Lm \times (L+1)m} \tag{1.38}$$

Example 1.3 (Conversion of first-order ARMAX to ESSM)
Consider the ARMAX model

$$y_k = a y_{k-1} + b u_{k-1} + f e_k + c e_{k-1} \tag{1.39}$$

An ESSM with $L = 3$ can be constructed as follows.

$$\begin{aligned}
y_{k+1} &= y_{k+1} \\
y_{k+2} &= y_{k+2} \\
y_{k+3} &= ay_{k+2} + bu_{k+2} + fe_{k+3} + ce_{k+2}
\end{aligned}$$

which gives

$$\underbrace{\begin{bmatrix} y_{k+1} \\ y_{k+2} \\ y_{k+3} \end{bmatrix}}_{y_{k+1|3}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & a \end{bmatrix}}_{\tilde{A}_3} \underbrace{\begin{bmatrix} y_k \\ y_{k+1} \\ y_{k+2} \end{bmatrix}}_{y_{k|3}} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & b \end{bmatrix}}_{\tilde{B}_3} \underbrace{\begin{bmatrix} u_k \\ u_{k+1} \\ u_{k+2} \end{bmatrix}}_{u_{k|3}} + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & c & f \end{bmatrix}}_{\tilde{C}_3} \underbrace{\begin{bmatrix} e_k \\ e_{k+1} \\ e_{k+2} \\ e_{k+3} \end{bmatrix}}_{e_{k|L+1}} \tag{1.40}$$

The model is proper, i.e. $g = 0$. This can also be seen from the ARMAX model. There is no direct feed-through term from the input u_k to the output y_k.

Chapter 2

Realization theory

2.1 Introduction

A method for the realization of linear state space models from known system input and output time series is studied. The input-output time series are usually obtained from a number of independently input experiments to the system.

2.2 Deterministic case

2.2.1 Model structure and problem description

Assume that a system can be illustrated as shown in Figure (2.2.1). $y \in \Re^{ny}$

Figure 2.1: Dynamic system with inputs and outputs

is the system outputs which are measured, $u \in \Re^{nu}$ is the system inputs which can be manipulated, $\bar{v} \in \Re^{nv}$ is unknown system disturbances, and $\bar{w} \in \Re^{ny}$ is unknown output (measurement) disturbances.

Assume that the system can be described by the discrete state space model

$$x_{i+1} = Ax_i + Bu_i + f_i(\bar{v}_i) \tag{2.1}$$

$$y_i = Dx_i + g_i(\bar{w}_i) \tag{2.2}$$

where i is discrete time, ie. an integer, $x \in \Re^{nx}$ is the state vector, and x_0 is the initial state. A, B and D are time invariant matrices of appropriate dimensions, where (A, B) is a controllable pair and (D, A) is an observable pair.

The problem investigated in this work is to estimate the state space model matrices A, B and D from known input and output time series. The input-output time series are usually obtained from a number of independently input experiments to the system. $f(\cdot)$ and $g(\cdot)$ are non-linear functions.

The derivation of the method will be based on the assumption that the functions f and g are linear, ie.

$$f_i(\bar{v}_i) = C\bar{v}_i \qquad\qquad g_i(\bar{w}_i) = \bar{w}_i \tag{2.3}$$

This is not a restriction, but a choice, which make it possible to apply stochastic realization theory to estimate the noise covariance matrices, in addition to A, B and D.

2.2.2 Impulse response model

A linear state space model $x_{x+1} = Ax_k + Bu_k$ and $y_k = Dx_k + Eu_k$ with the initial state x_0 given, can be described as the following impulse response model

$$y_k = DA^k x_0 + \sum_{i=1}^{k} DA^{k-i}Bu_{i-1} + Eu_k. \tag{2.4}$$

We define the matrix

$$H_{k-i+1} = DA^{k-i}B \ \in \ \mathbb{R}^{ny\times nu}, \tag{2.5}$$

as the impulse response matrix at time instant $k-i+1$. hence, the output, y_k, at time instant k is defined in terms of the impulse response matrices $H_1 = DB$, $H_2 = DAB$, …, $H_k = DA^{k-1}B$.

2.2.3 Determination of impulse responses

To determine the impulse responses a set of process experiments have to be performed. u_i^j is defined as the control input at time instant i for experiment number j, and y_i^j is the corresponding process output. The derivation is based on information from $mk = nu+1$ experiments, however, this is not a restriction.

The derivation is based on the assumption that the noise vectors f and g are the same for each experiment. In this case, exact impulse responses are the result of this method.

In most cases, the noise vectors are different for each experiment, for example when the disturbances are white noise. In this case errors are introduced, and a crude estimate may occur. However, these errors may be reduced.

Time instant $i = 1$

$$\begin{aligned} x_1 &= Ax_0 + C\bar{v}_0 + Bu_0 \qquad (2.6)\\ y_1 &= D(Ax_0 + C\bar{v}_0) + \bar{w}_1 + DBu_0 \qquad (2.7)\end{aligned}$$

or in matrix form

$$y_1 = \begin{bmatrix} z_1 & H_1 \end{bmatrix} \begin{bmatrix} 1 \\ u_0 \end{bmatrix} \tag{2.8}$$

where

$$\begin{aligned} H_1 &= DB && (2.9) \\ z_1 &= D(Ax_0 + C\bar{v}_0) + \bar{w}_1 && (2.10) \end{aligned}$$

For $nu+1$ experiments

$$Y_1 = \begin{bmatrix} z_1 & H_1 \end{bmatrix} \begin{bmatrix} ones(1, nu+1) \\ U_0 \end{bmatrix} \tag{2.11}$$

where

$$Y_1 = \begin{bmatrix} y_1^1 & y_1^2 & \cdots & y_1^{nu+1} \end{bmatrix} \quad U_0 = \begin{bmatrix} u_0^1 & u_0^2 & \cdots & u_0^{nu+1} \end{bmatrix} \tag{2.12}$$

Time instant $i = 2$

$$\begin{aligned} x_2 &= A(Ax_0 + C\bar{v}_0) + C\bar{v}_1 + ABu_0 + Bu_1 && (2.13) \\ y_2 &= D(A^2x_0 + AC\bar{v}_0 + C\bar{v}_1) + \bar{w}_2 + DABu_0 + DBu_1 && (2.14) \end{aligned}$$

or in matrix form

$$y_2 = \begin{bmatrix} z_2 & H_2 & H_1 \end{bmatrix} \begin{bmatrix} 1 \\ u_0 \\ u_1 \end{bmatrix} \tag{2.15}$$

where

$$\begin{aligned} H_2 &= DAB && (2.16) \\ z_2 &= D(A^2x_0 + AC\bar{v}_0 + C\bar{v}_1) + \bar{w}_2 && (2.17) \end{aligned}$$

For $nu+1$ experiments

$$Y_2 = \begin{bmatrix} z_2 & H_2 & H_1 \end{bmatrix} \begin{bmatrix} ones(1, nu+1) \\ U_0 \\ U_1 \end{bmatrix} \tag{2.18}$$

where

$$Y_2 = \begin{bmatrix} y_2^1 & y_2^2 & \cdots & y_2^{nu+1} \end{bmatrix} \quad U_1 = \begin{bmatrix} u_1^1 & u_1^2 & \cdots & u_1^{nu+1} \end{bmatrix} \tag{2.19}$$

Time instants $i = 1, ..., 2n$

The results, for time instants 1 to $2n$, may be stacked in the following matrix system, which have a suitable structure for the determination of the unknown matrices H_i and the vectors z_i. Note that the information from time instant $i = 0$ is not necessary at this point.

$$\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_{2n} \end{bmatrix} = \begin{bmatrix} z_1 & H_1 & 0 & 0 & \cdots & 0 \\ z_2 & H_2 & H_1 & 0 & & 0 \\ z_3 & H_3 & H_2 & H_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ z_{2n} & H_{2n} & \cdots & H_3 & H_2 & H_1 \end{bmatrix} \begin{bmatrix} ones(1, nu+1) \\ U_0 \\ U_1 \\ U_2 \\ \vdots \\ U_{2n-1} \end{bmatrix} \tag{2.20}$$

This linear matrix equation may be solved recursively. Start with $i = 1$ and solve for $[z_1 \ H_1]$. Then successively solve the i th equation for $[z_i \ H_i]$ for $i = 1, ..., 2n$. We have

$$\begin{bmatrix} z_i & H_i \end{bmatrix} \begin{bmatrix} ones(1, nu+1) \\ U_0 \end{bmatrix} = (Y_i - \sum_{k=1}^{i-1} H_{i-k} U_k) \tag{2.21}$$

or

$$\begin{bmatrix} z_i & H_i \end{bmatrix} = (Y_i - \sum_{k=1}^{i-1} H_{i-k} U_k) \begin{bmatrix} ones(1, nu+1) \\ U_0 \end{bmatrix}^{-1} \tag{2.22}$$

because the matrix is non singular by the definition of the input experiments $u^j \ j = 1, ..., nu+1$.

We have now determined the impulse responses H_i and the vectors z_i for all $i = 1, ..., 2n$ which contain information of noise and initial values. H_i and z_i can be expressed by

$$H_i = DA^{i-1}B \tag{2.23}$$

$$z_i = DA^i x_0 + \sum_{k=1}^{i} DA^{i-k} C \bar{v}_{k-1} + \bar{w}_i \tag{2.24}$$

Observe that the sequence $z_i \ \forall \ i = 1, ..., 2n$ is generated by the following state space model.

$$s_{i+1} = As_i + C\bar{v}_i \quad s_0 = x_0 \tag{2.25}$$

$$z_i = Ds_i + \bar{w}_i \tag{2.26}$$

where for the moment, z_0, is undefined. Define $e = x - s$, then

$$e_{i+1} = Ae_i + Bu_i \quad e_0 = 0 \tag{2.27}$$

$$y_i = De_i + z_i \tag{2.28}$$

The problem is now reduced to estimate the matrices (A, B, D), which satisfy the model, Equations (2.27) and (2.28), from known impulse respenses H_i. In addition, initial values and noise statistics can be estimated from the model given by Equations (2.25) and (2.26) where z_i and the matrices A, and D are known.

z_0 must be defined before continuing. At time instant $i = 0$ we have $y_0 = Dx_0 + \bar{w}_0$. For $nu + 1$ experiments

$$Y_0 = Z_0 \tag{2.29}$$

$$Y_0 = \begin{bmatrix} y_0^1 & y_0^2 & \cdots & y_0^{nu+1} \end{bmatrix} \quad Z_0 = \begin{bmatrix} z_0^1 & z_0^2 & \cdots & z_0^{nu+1} \end{bmatrix} \tag{2.30}$$

We have assumed that the initial values are equal for all experiments. In this case z_0 is chosen as one of the columns in Y_0. z_0 can even be computed from

$$\begin{bmatrix} z_0 & 0 \end{bmatrix} = Y_0 \begin{bmatrix} ones(1, nu+1) \\ U_0 \end{bmatrix}^{-1} \tag{2.31}$$

However, this last method can have numerical disadvantages. It is a method for constructing a right inverse for $ones(1, nu+1)$, and is nothing but the mean of the columns of the matrix Y_0.

If the initial values are different for some or all $nu + 1$ experiments, then, there are obviously many choices for z_0. One choice is the mean of the columns of the matrix Y_0, which is consistent with the case of equal initial values, ie.

$$z_0 = \frac{1}{nu+1} \sum_{j=1}^{nu+1} y_0^j \tag{2.32}$$

2.2.4 Redundant or missing information

Assume that the number of experiments, m, are different from $nu + 1$. In this case we have

$$\begin{bmatrix} z_i & H_i \end{bmatrix} \tilde{U}_0 = (Y_i - \sum_{k=1}^{i-1} H_{i-k} U_k) \tag{2.33}$$

Assume that $m \geq nu+1$ and $rank(\tilde{U}_0 \tilde{U}_0^T) = m$, then, the solution of Equation (2.33) with respect to H_i and z_i is given by

$$\begin{bmatrix} z_i & H_i \end{bmatrix} = (Y_i - \sum_{k=1}^{i-1} H_{i-k} U_k) \tilde{U}_0^T (\tilde{U}_0 \tilde{U}_0^T)^{-1} \tag{2.34}$$

Assume that $m < nu+1$ or $rank(\tilde{U}_0 \tilde{U}_0^T) < m$. An approximate solution is determined by

$$\begin{bmatrix} z_i & H_i \end{bmatrix} = (Y_i - \sum_{k=1}^{i-1} H_{i-k} U_k) \tilde{V} \tilde{S}^{+} \tilde{U}^T \tag{2.35}$$

where

$$\tilde{U}_0 = \tilde{U} \tilde{S} \tilde{V}^T \tag{2.36}$$

is the singular value decomposition. In any cases the solutions discussed above minimizes the 2 - norm of the error residual.

Note the connection to the theory of multivariate calibration, Martens and Næss (1989), where usually a steady state model of the form $y = Hu + z$ is chosen to fit the data. The gain H is computed in the same way as outlined above. Our method can thus be viewed as a dynamic extension of the theory of multivariate calibration.

2.2.5 The Hankel matrices

The (block) Hankel matrices, which contains information of the system matrices, can now be constructed from the impulse responses $H_i \ \forall \ i = 1, ..., L + J$.

$$\mathbf{H} = \begin{bmatrix} H_1 & H_2 & H_3 & \cdots & H_J \\ H_2 & H_3 & H_4 & & H_{J+1} \\ H_3 & H_4 & H_5 & \cdots & H_{J+2} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ H_{L+1} & H_{L+2} & \cdots & \cdots & H_{L+J} \end{bmatrix} \quad \Re^{ny(L+1)\times nu\cdot J} \tag{2.37}$$

We extract the following submatrices of interests.

$$\mathbf{H}_{1|L} = \mathbf{H}_n = \begin{bmatrix} H_1 & H_2 & H_3 & \cdots & H_J \\ H_2 & H_3 & H_4 & & H_{J+1} \\ H_3 & H_4 & H_5 & \cdots & H_{J+2} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ H_L & H_{L+1} & \cdots & \cdots & H_{L+J-1} \end{bmatrix} \quad \Re^{ny\cdot L\times nu\cdot J} \tag{2.38}$$

$$\mathbf{H}_{2|L} = \mathbf{H}_A = \begin{bmatrix} H_2 & H_3 & H_3 & \cdots & H_{J+1} \\ H_3 & H_4 & H_4 & & H_{J+2} \\ H_4 & H_5 & H_5 & \cdots & H_{J+3} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ H_{L+1} & H_{L+2} & \cdots & \cdots & H_{L+J} \end{bmatrix} \quad \Re^{ny\cdot L\times nu\cdot J} \tag{2.39}$$

$$\mathbf{H}_B = \begin{bmatrix} H_1 \\ H_2 \\ H_3 \\ \vdots \\ H_L \end{bmatrix} \quad \Re^{ny\cdot L\times nu} \tag{2.40}$$

$$\mathbf{H}_D = \begin{bmatrix} H_1 & H_2 & H_3 & \cdots & H_J \end{bmatrix} \quad \Re^{ny\times nu\cdot J} \tag{2.41}$$

The realization theory is based on the observation that these matrices are closely related to the observability matrix O_L and the controllability matrix C_J given by.

$$O_L = \begin{bmatrix} D \\ DA \\ DA^2 \\ \vdots \\ DA^{L-1} \end{bmatrix} \quad \Re^{ny \cdot L \times nx} \tag{2.42}$$

$$C_J = \begin{bmatrix} B & AB & A^2B & \cdots & A^{J-1}B \end{bmatrix} \quad \Re^{nx \times nu \cdot J} \tag{2.43}$$

The following factorization is the basic for the realization of the number of states nx and the system matrices A, B and D.

$$\mathbf{H}_n = \mathbf{H}_{1|L} = O_L C_J \tag{2.44}$$

At this stage it must be pointed out that the factorization of the Hankel matrix by numerical methods, usually results in observability and controllability matrices for a model representation in a different co-ordinate system than the underlying system.

The number of states nx is estimated as the rank of $\mathbf{H}_n = \mathbf{H}_{1|L}$. To obtain a proper estimate we must ensure that

$$L \geq nx - \text{rank}(D) + 1, J \geq nx - \text{rank}(B) + 1. \tag{2.45}$$

The reason for this is that, in this case, $\text{rank}(O_L) = nx$ when the pair (D, A) is observable, and similarly, $\text{rank}(C_J) = nx$ when the pair (A, B) is controllable. O_L and C_J may be determined by a suitable factorization of the known block Hankel matrix, $\mathbf{H}_n = \mathbf{H}_{1|L}$. The system matrices is then chosen to satisfy the following three matrix relations.

$$\mathbf{H}_{1|L} = \mathbf{H}_A = O_L A C_J, \quad \mathbf{H}_B = O_L B, \quad \mathbf{H}_D = D C_J \tag{2.46}$$

The system matrices are then estimated by

$$\begin{aligned} A &= (O_L^T O_L)^{-1} O_L^T \mathbf{H}_A C_J^T (C_J C_J^T)^{-1}, & (2.47) \\ B &= (O_L^T O_L)^{-1} O_L^T \mathbf{H}_B, & (2.48) \\ D &= \mathbf{H}_D C_J^T (C_J C_J^T)^{-1}. & (2.49) \end{aligned}$$

Assume for the moment that $\bar{v} = \bar{w} = 0$. The initial values satisfy, in this case

$$z_n = O_L A x_0 \tag{2.50}$$

and can be estimated by

$$x_0 = A^{-1} (O_L^T O)^{-1} O_L^T z_n \tag{2.51}$$

if A is non-singular. If A is singular then the information from time instant $i = 0$, $y_0 = D x_0 + \bar{w}_0$, or $z_0 = y_0 = D x_0$ by assumption, can be added to the algorithm to avoid the inversion. We have

$$x_0 = (O_L^T O_L)^{-1} O_L^T \tilde{z}_n \tag{2.52}$$

where $\tilde{z}_n = [z_0^T, z_{n-1}^T]^T$ is the vector where z_0 is augmented to the first $n-1$ elements in z_n.

Note that the estimated model matrices usually are related to a different co-ordinate system than the underlying model. Let $x = T\tilde{x}$ be the transformation from the estimated state vector $\tilde{x}$ to the underlying state vector x. The observability and controllability matrices are given by $\tilde{O}_L = O_L T$ and $\tilde{C}_J = T^{-1}C_J$. Which means that the transformation T is given by

$$T = (O_L^T O_L)^{-1} O_L^T \tilde{O}_L, \qquad T^{-1} = \tilde{C}_J C_J^T (C_J C_J^T)^{-1}. \tag{2.53}$$

2.2.6 Balanced and normalized realizations

Perform the singular value decomposition, SVD, of the finite block Hankel matrix $\mathbf{H}_n$.

$$\mathbf{H}_{1|L} = \mathbf{H}_n = O_L C_J = USV^T = US_1S_2V^T. \tag{2.54}$$

The order of the state space model is equal to the number of the non zero singular values which is the same as the rank of $\mathbf{H}_n$, ie. $nx = rank(\mathbf{H}_n)$. A reduced model is directly determined by choosing a subset of the non zero singular values.

The following choices for the factorization of the Hankel matrix into the product of the observability and controllability matrices results directly from the SVD.

$$\begin{array}{lll} O_L = US_1, & C_J = S_2V^T & \text{(internally balanced)}, \\ O_L = U, & C_J = SV^T & \text{(output normal)}, \\ O_L = US, & C_J = V^T & \text{(input normal)}. \end{array} \tag{2.55}$$

These factorizations are called internally balanced, output normal and input normal, respectively, according to the definitions by Moore (1981). See also Silverman and Bettayeb (1980). The meaning of these definitions will be made clear later in this section.

If the internally balanced realization, as in Aoki (1990), is used, then the system matrices are estimated by

$$\left.\begin{array}{l} A = S_1^{-T} U^T \mathbf{H}_A V S_2^{-T} \\ B = S_1^{-T} U^T \mathbf{H}_B \\ D = \mathbf{H}_D V S_2^{-T} \end{array}\right\} \quad \text{(internally balanced)} \tag{2.56}$$

The output normal and input normal realizations are related to the co-ordinate system, in which the internally balanced system is presented, by a transformation. These realizations are shown below for completeness.

$$\left.\begin{array}{l} A = U^T \mathbf{H}_A V S^{-1} \\ B = U^T \mathbf{H}_B \\ D = \mathbf{H}_D V S^{-1} \end{array}\right\} \quad \text{(output normal)} \tag{2.57}$$

$$\left.\begin{array}{l} A = S^{-1}U^T\mathbf{H}_A V \\ B = S^{-1}U^T\mathbf{H}_B \\ D = \mathbf{H}_D V \end{array}\right\} \quad \text{(input normal)} \tag{2.58}$$

The initial values may in all cases be estimated from Equation (2.51) if $\bar{v} = \bar{w} = 0$.

Note that the matrix multiplications to perform B and D are not necessary, because B is equal to the first $nx\ x\ nu$ submatrix of $C_J = US_1$ and D is equal to the first $ny\ x\ nx$ submatrix of $O_L = S_2V^T$, in the internally balanced realization.

Note that the SVD is not unique, in the sense that it may exist other orthogonal matrices U and V satisfying $\mathbf{H}_n = USV^T$. One can always change sign for one column in U and in the corresponding column in V. However, S is of course unique, if the singular values are ordered in descending order of magnitude along the diagonal. This means that the estimated system matrices are not unique, because the estimated matrices are dependent on the choice of U and V, this is contradictory to the statement of uniqueness in Aoki (1990) pp. 109.

However, the estimates have the properties that the L - observability and the J - controllability grammians defined by

$$W_{o,L} = O_L^T O_L = \sum_{i=1}^{L} A^{(i-1)T} D^T D A^{(i-1)} \tag{2.59}$$

$$W_{c,J} = C_J C_J^T = \sum_{i=1}^{J} A^{(i-1)} B B^T A^{(i-1)T} \tag{2.60}$$

satisfy

$$\begin{array}{lll} W_{o,L} = S_{nx} & W_{c,J} = S_{nx} & \text{(internally balanced)} \\ W_{o,L} = I & W_{c,J} = S_{nx}^2 & \text{(output normal)} \\ W_{o,L} = S_{nx}^2 & W_{c,J} = I & \text{(input normal)} \end{array} \tag{2.61}$$

This can be shown by substituting the L - observability and the J - controllability matrices given in (2.55) into the grammians, Equations (2.59) and (2.60).

A model which satisfy one of the properties in (2.61) is called internally balanced, output normal or input normal, respectively. This definitions are due to Moore (1981).

The model produced by the method of Aoki (1990) is by this definition internally balanced. A model is called internally balanced if the corresponding observability and controllability grammians are equal to the same diagonal matrix. Particularly, a diagonal matrix with the non zero singular values of the Hankel matrix on the diagonal, in this case. This imply that the new system is "as controllable as it is observable".

Note that the estimates (2.56) are not internally balanced due to the infinite observability and the infinite controllability grammians W_o and W_c, which are the grammians for a stable system when $n \to \infty$ in Equations (2.59) and (2.60). These grammians satisfy the discrete Lyapunov matrix equations

$$A^T W_o A - W_o = -D^T D, \tag{2.62}$$
$$A W_c A^T - W_c = -BB^T, \tag{2.63}$$

if A is stable.

The estimate of the process model is usually obtained from finite Hankel matrices, i.e., for finite L and J. Note also that usually $L = J$. This model estimate may be transformed to a model representation with balanced or normalized grammians which satisfy the Lyapunov equations (2.62) and (2.63). See Moore (1981), Silverman and Bettayeb (1980) and Laub (1980) for the determination of such a transformation.

The statement that the internally balanced estimates are not unique will now be clarified. Suppose that the model (A, B, D) is transformed to a new co-ordinate system with $x = T\tilde{x}$. The grammians for the new system is then given by

$$\tilde{W}_o = T^T W_o T, \tag{2.64}$$
$$\tilde{W}_c = T^{-1} W_c T^{-T}. \tag{2.65}$$

Assume that both realizations is internally balanced so that

$$T^T W_o T = T^{-1} W_c T^{-T} = S_{nx}, \tag{2.66}$$
$$W_o = W_c = S_{nx}. \tag{2.67}$$

This implies that

$$T^{-1} S_{nx}^2 T = S_{nx}^2, \tag{2.68}$$
$$\tag{2.69}$$

which constrain T to be a diagonal matrix with ±1 in each diagonal entry. The argumentation shows that a model which is internally balanced is not unique. However, the set of such models are constrained within sign changes.

2.2.7 Error analysis

When the order of the state space model is chosen less than the rank of the Hankel matrix, errors is introduced.

The output sequence may be written as

$$y_i = \sum_{k=1}^{i} H_{i-k} u_{k-1} + z_i \tag{2.70}$$

where the impulse responses H_i and the noise terms z_i are known. See Section (2.2.3).

The estimated state space model can be written in input output form as

$$\hat{y}_i = \sum_{k=1}^{i} \hat{H}_{i-k} u_{k-1} + \hat{z}_i \tag{2.71}$$

where $\hat{H}_i$ are the impulse responses computed from the estimated state space model, and $\hat{z}_i$ is the output from an estimated model of the noise and initial values based on the computed sequence z_i.

The estimation error is given by

$$\varepsilon_i = \sum_{k=1}^{i} \Delta H_{i-k} u_{k-1} + \Delta z_i \tag{2.72}$$

where $\Delta z_i = z_i + \hat{z}_i$, which states that the output sequence can be written

$$y_i = \hat{y}_i + \varepsilon_i \tag{2.73}$$

2.3 Numerical examples

2.3.1 Example 1

Table 2.1: Input-output time series

t	1	2	3	4	5	6	7	8	9	10	11	12	13	14
u	0.1	0.1	0.1	0.1	5	6	7	8	9	10	0.01	12	13	0.1
y	0.10	0.01	0.05	0.06	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.01	0.01	0.01

2.4 Concluding remarks

Some (preliminary) comments are stated in the following items.

- It is not necessary that the time series are zero mean. If a linearized model representation of a stable system around steady state values is the goal, then, the mean steady state values of the inputs and outputs at time instants $i = 0$ shuld be subtracted from the input and output time series.

- The method works for unstable systems, provided the time series are finite. The method is not based on time series covariance estimation in the usual way, ie. estimating $\Lambda_k = E(y_{i+k} y_i^T) = DA^{k-1}M$ by applying series methods or Fourier transform techniques (M contain information of noise statistics), or cross covariance estimation, ie. estimating $H_k = E(y_{i+k} u_i^T) = DA^{k-1}BE(u_i u_i^T)$.

- The method works for non-minimum phase systems. If the time series are generated from a state-output system (A, B, D), then the algorithm realize an equivalent state-output system $(\tilde{A}, \tilde{B}, \tilde{D})$ with respect to impulse responses, poles and zeros, even if the system is non-minimum phase. The co-ordinate system in which the realized model is presented, is of course generally different from the underlying co-ordinate system. The Markov parameters are estimated from information of both u and y, not only y as in standard realizing algorithms such as Aokis method. Recall that transmission zeros cannot be observed from the outputs alone, which means that realizing algorithms which are based on the estimation of Markov parameters from y alone do not work for non-minimum phase systems.

- If the time series are generated from a transition-output system (A, B, D, E), an essentially equivalent state-output system $(\tilde{A}, \tilde{B}, \tilde{D})$, with respect to poles, zeros and impulse responses, is realized by this method. The instantaneous dynamics from u to y in the transition-output system, are realized with eigenvalues equal to zero in $\tilde{A}$, which means that states are added which are infinitely fast. The spectrum of $\tilde{A}$ is essentially equivalent to the spectrum of A, exept for som zero poles added. The zeros of the realization $(\tilde{A}, \tilde{B}, \tilde{D})$ is generally different from the zeros of an underlying system with realization (A, B, D, E). But the realization $(\tilde{A}, \tilde{B}, \tilde{D})$ can be changed to a realization $(\bar{A}, \bar{B}, \bar{D}, \bar{E})$, with the same zeros as the underlying system (A, B, D, E). This means that if the underlying system (A, B, D, E) is non-minimum phase (zeros outside the unit-circle in the z-plane), the realization $(\tilde{A}, \tilde{B}, \tilde{D})$ can lock like a minimum phase system (zeros inside the unit-circle in the z-plane), hence; If there are zero eigenvalues (or zero in a relative sense) in the realization $(\tilde{A}, \tilde{B}, \tilde{D})$, change to (A, B, D, E) system before computing zeros!

References

Aoki M. (1987). *State Space Modeling of Time Series.* Springer-Verlag Berlin, Heidelberg.

Aoki M. (1990). *State Space Modeling of Time Series.* Springer-Verlag Berlin, Heidelberg.

Laub, A. J. (1980). Computation of "balancing" transformations. *Proc. JACC, San Francisco*, FA8-E.

Martens, H. and T. Næss (1989). "Multivariate Calibration". John Wiley and Sons Ltd. Chichester.

Moore, B. C. (1981). Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction. *IEEE Trans. on Automatic Control*, Vol. AC-26, pp. 17-31.

Silverman L. M. and M. Bettayeb (1980). Optimal approximation of linear systems. *Proc. JACC, San Francisco*, FA8-A.

Chapter 3

Combined *D*eterministic and *S*tochastic System Identification and *R*ealization - *DSR*: A subspace approach based on observations [1]

Abstract

The paper presentes a numerically stable and general algorithm for identification and realization of a complete dynamic linear state space model, including the system order, for combined deterministic and stochastic systems from time series. A special property of this algorithm is that the innovations covariance matrix and the Markov parameters for the stochastic sub-system are determined directly from a projection of known data matrices, without e.g. recursions of non-linear matrix Riccati equations. A realization of the Kalman filter gain matrix is determined from the estimated extended observability matrix and the Markov parameters. Monte Carlo simulations are used to analyze the statistical properties of the algorithm as well as to compare with existing algorithms.

[1] ECC95 paper extended with proofs, new results and theoretical comparison with existing subspace identification methods. Also in *Computer Aided Time Series Modeling*, Edited by Masanao Aoki, Springer Verlag, 1997.

3.1 Introduction

System identification can be defined as building mathematical models of systems based on observed data. Traditionally a set of model structures with some free parameters are specified and a *prediction error* (PE) criterion measuring the difference between the observed outputs and the model outputs is optimized with respect to the free parameters. In general, this will result in a non linear optimization problem in the free parameters even when a linear time invariant model is specified. A tremendous amount of research has been reported, resulting in the so called *prediction error methods* (PEM).

In our view the field of *subspace identification*, Larimore (1983) and (1990), Verhagen (1994), Van Overschee and De Moor (1994), Di Ruscio (1994), not only resolves the problem of system identification but also deals with the additional problem of *structure identification*. In subspace identification methods a data matrix is constructed from certain projections of the given system data. The observability matrix for the system is extracted as the column space of this matrix and the system order is equal to the dimension of the column space.

Descriptions of the advantages of subspace identification methods over traditional PEM can be found in Viberg (1995) and in Van Overschee (1995).

Aoki (1990) has presented a method for the realization of state space linear discrete time stochastic models on innovations form. See also Aoki (1994) for some further improvements of the method. This method has many interesting numerical properties and many of the numerical tools involved in the method are common with the tools used by the *subspace identification* methods. The method is based on the factorization of the Hankel matrix, constructed from covariance matrices of the output time series, by singular value decomposition. The states are presented relative to an internally balanced coordinate system, Moore (1981), which has some interesting properties when dealing with model reduction.

The *subspace identification* methods which are presented in the above references are using instrumental variables constructed from past input and output data in order to remove the effect of noise from future data. It should be pointed out that the method by Aoki also uses instrumental variables constructed from past data.

The method for system identification and state space model realization which is presented in this work is believed to be a valuable tool for the analysis and modeling of observed input and output data from a wide range of systems, in particular combined deterministic and stochastic dynamical systems. Only linear algebra is applied in order to estimate a complete linear time invariant state space model.

Many successful applications of data based time series analysis and modeling methods are reported. One industrial application is presented in Di Ruscio and Holmberg (1996). One particularly important application is the estimation of econometric models, see Aoki (1990).

The remainder of the paper is organized as follows. Section 3.2 gives a definition of the system and the problem considered in this work. In Section 3.3 the data is organized into data matrices which satisfy an extended state space model or matrix equation. Section 3.4 shows how the system order and the model matrices can be extracted from the known data matrices. A numerically stable and efficient implementation is presented in Section 3.5. Section 3.6 gives a comparison of the method presented in this work with other published methods. Real world numerical examples and theoretical Monte Carlo simulations are presented in Section 3.7 and some concluding remarks follow in Section 3.8.

3.2 Preliminary Definitions

3.2.1 System Definition

Assume that the underlying system can be described by a discrete-time, time invariant, linear state space model (SSM) of the form

$$x_{k+1} = Ax_k + Bu_k + Ce_k \tag{3.1}$$

$$y_k = Dx_k + Eu_k + e_k \tag{3.2}$$

where the integer $k \geq 0$ is discrete-time, $x \in \mathbb{R}^n$ is the state vector with initial value x_0, $y \in \mathbb{R}^m$ is the system output, $u \in \mathbb{R}^r$ is the system input, $e \in \mathbb{R}^m$ is an unknown innovations process of white noise, assumed to be covariance stationary, with zero mean and covariance matrix $\mathrm{E}(e_k e_k^T) = \Delta$. The constant matrices in the SSM are of appropriate dimensions. A is the *state transition* matrix, B is the *external input* matrix, C is the *Kalman gain* matrix, D is the *output* matrix and E is the direct *control input to output* (feed-through) matrix. We will assume that (D, A) is an observable pair.

The innovations model, Equations (3.1) and (3.2), is discussed in e.g. Faurre (1976) and Aoki (1990).

3.2.2 Problem Definition

The problem investigated in this paper is to identify a state space model, including the system order (n), for both the deterministic and the stochastic part of the system, i.e. the quadruple matrices (A, B, D, E) and the double matrices (C, Δ) respectively, directly from known system input and output data vectors (or time series) defined as

$$\left.\begin{array}{lll} u_k & \forall & k = 0, \ldots, N-1 \\ y_k & \forall & k = 0, \ldots, N-1 \end{array}\right\} \textbf{Known data vectors}$$

In continuous time systems the matrix E in Equation (3.2) is usually zero. This is not the case in discrete time systems due to sampling. However, E can be forced to be zero by including a structure constraint. This will be commented on later.

3.2.3 Matrix Definitions

Associated with the SSM, Equations (3.1) and (3.2), we make the following definitions:

- The *extended observability* matrix (O_i) for the pair (D, A) is defined as

$$O_i \overset{\text{def}}{=} \begin{bmatrix} D \\ DA \\ \vdots \\ DA^{i-1} \end{bmatrix} \in \mathbb{R}^{im \times n} \tag{3.3}$$

  where subscript i denotes the number of block rows. The matrix O_i is denoted the *extended observability* matrix when the number of block rows i is grater than the minimal number of block rows needed in the observability matrix, in order to check if the system is observable.

- The *reversed extended controllability* matrix (C_i^d) for the pair (A, B) is defined as

$$C_i^d \overset{\text{def}}{=} \begin{bmatrix} A^{i-1}B & A^{i-2} & \cdots & B \end{bmatrix} \in \mathbb{R}^{n \times ir} \tag{3.4}$$

  where subscript i denotes the number of block columns.

  A matrix C_i^s for the pair (A, C) is defined similarly to Equation (3.4), i.e. with C substituted for B in the above definition.

- The *lower block triangular Toeplitz* matrix (H_i^d) for the quadruple matrices (D, A, B, E)

$$H_i^d \overset{\text{def}}{=} \begin{bmatrix} E & 0 & 0 & \cdots & 0 \\ DB & E & 0 & \cdots & 0 \\ DAB & DB & E & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}B & DA^{i-3}B & DA^{i-4}B & \cdots & E \end{bmatrix} \in \mathbb{R}^{im \times ir} \tag{3.5}$$

  where subscript i denotes the number of block rows.

  A *lower block triangular Toeplitz* matrix H_i^s for the quadruple matrices (D, A, C, F) is defined as

$$H_i^s \overset{\text{def}}{=} \begin{bmatrix} F & 0 & 0 & \cdots & 0 \\ DC & F & 0 & \cdots & 0 \\ DAC & DC & F & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}C & DA^{i-3}C & DA^{i-4}C & \cdots & F \end{bmatrix} \in \mathbb{R}^{im \times im} \tag{3.6}$$

  where $F = I$ for the output model formulation, Equation (3.2).

3.2.4 Notation

The projection A/B of two matrices A and B is defined as $AB^T(BB^T)^\dagger B$ where $\dagger$ denotes the Moore-Penrose pseudo-inverse of a matrix.

3.3 Extended State Space Model

The state space model, Equations (3.1) and (3.2), can generally be written as the following extended state space model (ESSM) (Di Ruscio (1994))

$$Y_{k+1|L} = \tilde{A}Y_{k|L} + \tilde{B}U_{k|L+1} + \tilde{C}E_{k|L+1} \tag{3.7}$$

where the known output and input data matrices $Y_{k|L}$ and $U_{k|L+1}$ are defined as follows

$$Y_{k|L} \overset{\text{def}}{=} \begin{bmatrix} y_k & y_{k+1} & y_{k+2} & \cdots & y_{k+K-1} \\ y_{k+1} & y_{k+2} & y_{k+3} & \cdots & y_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{k+L-1} & y_{k+L} & y_{k+L+1} & \cdots & y_{k+L+K-2} \end{bmatrix} \in \mathbb{R}^{Lm \times K} \tag{3.8}$$

$$U_{k|L+1} \overset{\text{def}}{=} \begin{bmatrix} u_k & u_{k+1} & u_{k+2} & \cdots & u_{k+K-1} \\ u_{k+1} & u_{k+2} & u_{k+3} & \cdots & u_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{k+L-1} & u_{k+L} & u_{k+L+1} & \cdots & u_{k+L+K-2} \\ u_{k+L} & u_{k+L+1} & u_{k+L+2} & \cdots & u_{k+L+K-1} \end{bmatrix} \in \mathbb{R}^{(L+1)r \times K} \tag{3.9}$$

The unknown data matrix $E_{k|L+1}$ of innovations noise vectors is defined as

$$E_{k|L+1} \overset{\text{def}}{=} \begin{bmatrix} e_k & e_{k+1} & e_{k+2} & \cdots & e_{k+K-1} \\ e_{k+1} & e_{k+2} & e_{k+3} & \cdots & e_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_{k+L-1} & e_{k+L} & e_{k+L+1} & \cdots & e_{k+L+K-2} \\ e_{k+L} & e_{k+L+1} & e_{k+L+2} & \cdots & e_{k+L+K-1} \end{bmatrix} \in \mathbb{R}^{(L+1)m \times K} \tag{3.10}$$

The scalar integer parameter L defines the number of block rows in the data matrices and the ESSM model matrices. The number of columns in $Y_{k|L}$, $U_{k|L+1}$ and $E_{k|L+1}$ are $K = N - L - k + 1$. Each column in these matrices can be interpreted as extended output, input and noise vectors, respectively. K can be viewed as the number of samples in these extended time series. We also have that $L < K < N$. L is the only necessary parameter which has to be specified by the user. L is equal to the number of block rows in the extended observability matrix ($O_L \in \mathbb{R}^{Lm \times n}$), which will be determined by the algorithm. For a specified L, the maximum possible order of the system to be identified is $n \leq Lm$ (if rank$(D) = m$, i.e. m independent outputs), or $n \leq Ld$ where $1 \leq d = \text{rank}(D) \leq m$, i.e. d independent output variables.

The parameter L can be interpreted as the *identification horizon*. This means that L is the horizon used to recover the present state space vector x_k.

The matrices in the extended state space model, Equation (3.7), are related to the underlying state space model matrices as follows

$$\tilde{A} = O_L A (O_L^T O_L)^{-1} O_L^T \tag{3.11}$$

$$\begin{aligned} \tilde{B} &= \left[\begin{array}{cccccc} OB - \tilde{A}E_1 & E_1 - \tilde{A}E_2 & E_2 - \tilde{A}E_3 & \cdots & E_{L-1} - \tilde{A}E_L & E_L \end{array}\right] \\ &= \left[\begin{array}{cc} O_L B & H_L^d \end{array}\right] - \tilde{A} \left[\begin{array}{cc} H_L^d & 0_{Lm\times r} \end{array}\right] \end{aligned} \tag{3.12}$$

$$\begin{aligned} \tilde{C} &= \left[\begin{array}{ccccc} O_L C - \tilde{A}F_1 & F_1 - \tilde{A}F_2 & F_2 - \tilde{A}F_3 \cdots & F_{L-1} - \tilde{A}F_L & F_L \end{array}\right] \\ &= \left[\begin{array}{cc} O_L C & H_L^s \end{array}\right] - \tilde{A} \left[\begin{array}{cc} H_L^s & 0_{Lm\times m} \end{array}\right] \end{aligned} \tag{3.13}$$

The matrices E_i and F_i, $i = 1, ..., L$, are block columns in the Toeplitz matrices H_L^d and H_L^s defined in Equations (3.5) and (3.6), i.e.

$$H_L^d = \left[\begin{array}{cccc} E_1 & E_2 & \cdots & E_L \end{array}\right] \tag{3.14}$$

$$H_L^s = \left[\begin{array}{cccc} F_1 & F_2 & \cdots & F_L \end{array}\right] \tag{3.15}$$

The importance of the ESSM, Equation (3.7), is that the state vector preliminary is eliminated from the problem. Hence, the number of unknowns is reduced. The ESSM also gives us the relationship between the data matrices and the model matrices which at this stage are unknown.

This paper is concerned with the problem of reconstructing the system order and system matrices in the state space model, (3.1) and (3.2), from the known data matrices $Y_{k|L}$ and $U_{k|L+1}$ which satisfy Equation (3.7). We refer to Di Ruscio (1994) and (1995) for a proof of the above results, which are the basis for the method presented in this work.

Note that the matrices H_L^d and H_L^s satisfy the matrix equation

$$Y_{k|L} = O_L X_k + \left[\begin{array}{cc} H_L^d & 0_{Lm\times r} \end{array}\right] U_{k|L+1} + \left[\begin{array}{cc} H_L^s & 0_{Lm\times m} \end{array}\right] E_{k|L+1} \tag{3.16}$$

where

$$X_k = \left[\begin{array}{ccccc} x_k & x_{k+1} & x_{k+2} & \cdots & x_{k+K-1} \end{array}\right] \in \mathbb{R}^{n\times K} \tag{3.17}$$

is a matrix of state vectors. Equation (3.16) is frequently used in other subspace identification methods, e.g. in Van Overschee and De Moor (1994) and Verhagen (1994).

3.4 System Identification and Realization

3.4.1 Identification and Realization of System Dynamics

The basic step in the algorithm is to identify the system order and the extended observability matrix from known data. In order to do so we will in this section derive an autonomous matrix equation from which the system dynamics can be identified. We will show how the system order n and the extended observability matrix O_L are identified. A realization for the system matrices A and D and the ESSM transition matrix $\tilde{A}$ are then computed.

The term $\tilde{B}U_{k|L+1}$ can be removed from Equation (3.7) by post-multiplying with a projection matrix $U^{\perp}_{k|L+1}$ such that $U_{k|L+1}U^{\perp}_{L|L+1} = 0$. The projection matrix can e.g. be defined as follows

$$U^{\perp}_{k|L+1} = I_{K\times K} - U^T_{k|L+1}(U_{k|L+1}U^T_{k|L+1})^{-1}U_{k|L+1} \tag{3.18}$$

Hence, $U^{\perp}_{k|L+1}$ is the orthogonal projection onto the null-space of $U_{k|L+1}$. A numerically well posed way of computing the projection matrix is by use of the singular value decomposition (SVD). The projection matrix is given by the left singular vectors of $U_{k|L+1}$ which are orthogonal to the null-space. However, in order to solve the complete system identification and realization problem, it is more convenient to use the QR decomposition for computing the projection, as will be shown in Section 3.5. Note that a projection matrix onto the null-space of $U_{k|L+1}$ exists if the number of columns K in the data-matrices satisfies $K > L + 1$.

Post-multiplying Equation (3.7) with the projection matrix $U^{\perp}_{k|L+1}$ gives

$$\begin{aligned} &Y_{k+1|L} - Y_{k+1|L}U^T_{k|L+1}(U_{k|L+1}U^T_{k|L+1})^{-1}U_{k|L+1} = \\ &\tilde{A}(Y_{k|L} - Y_{k|L}U^T_{k|L+1}(U_{k|L+1}U^T_{k|L+1})^{-1}U_{k|L+1}) + \\ &\tilde{C}(E_{k|L+1} - E_{k|L+1}U^T_{k|L+1}(U_{k|L+1}U^T_{k|L+1})^{-1}U_{k|L+1}) \end{aligned} \tag{3.19}$$

Note that the last noise term in equation (3.19) is per definition zero as the number of samples approaches infinity, i.e.

$$\lim_{K\to\infty} \tilde{C}\frac{1}{K}E_{k|L+1}U^T_{k|L+1} = 0 \tag{3.20}$$

Hence, we have the following result

$$Y_{k+1|L}U^{\perp}_{k|L+1} = \tilde{A}Y_{k|L}U^{\perp}_{k|L+1} + \tilde{C}E_{k|L+1} \tag{3.21}$$

The noise term $\tilde{C}E_{k|L+1}$ can be removed from Equation (3.21) by post-multiplying with $\frac{1}{K}W_i^T$ where W_i is defined as a matrix of "instrumental" variables which are uncorrelated with $E_{k|L+1}$, i.e., we are seeking for a matrix with the following property

$$\lim_{K\to\infty} \frac{1}{K}E_{k|L+1}W_i^T = 0 \tag{3.22}$$

An additional property is that W_i should be sufficiently correlated with the informative part in the ESSM in order not to destroy information about e.g. the system order.

An intuitive good choice is to use *past* data as instruments to remove *future* noise. This choice ensures that the instruments are sufficiently correlated with the informative part of the signals and sufficiently uncorrelated with future noise.

Define J as the number of time instants in the *past horizon* which is used for defining the instrumental variable matrix. Define L as the number of time instants in the horizon necessary for identifying the state at time instant k, that is x_k, as well as the extended observability matrix of the system. Define M as a prediction horizon. However, we will restrict ourself to the case $M = 1$ in this work. These horizons are illustrated schematically in Figure 3.1.

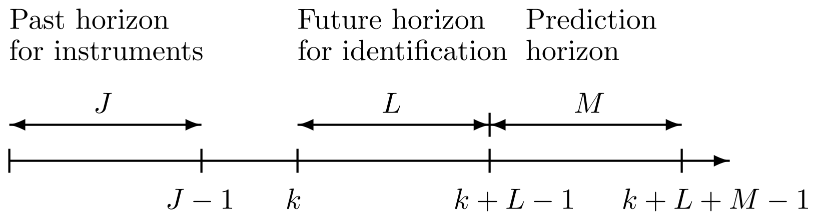

Past horizon for instruments
Future horizon for identification
Prediction horizon
J
L
M
$J-1$
k
$k+L-1$
$k+L+M-1$

Figure 3.1: Illustration of horizons involved in the **DSR** algorithm. Usually $k = J$. That is, the end of the past is the beginning of the future.

Some alternative instruments, W_i, that will remove the noise and which satisfy Equation (3.22) are as follows. Define

$$W_i \quad \in \quad \mathbb{R}^{ni \times K} \quad \forall \quad i = 1, 2, 3 \tag{3.23}$$

where the row dimension ni is the number of instrumental variables and

$$W_1 = \begin{bmatrix} Y_{0|J} \\ U_{l|J} \end{bmatrix} \quad W_2 = Y_{0|J} \quad W_3 = U_{l|J} \qquad \forall \; l \geq 0 \tag{3.24}$$

The choice of instruments is unimportant in the deterministic case, i.e. when the process as well as the observation noise are identically zero which also means that $e_k = 0$ for all discrete time instants. However, the “optimal” choice of instruments is important in the combined deterministic and stochastic case.

Assume that $U_{l|J}$ with $l > 0$ is chosen as the instrumental variable matrix. This means that not only past inputs but also future inputs are used to remove future noise (first time instant in the future horizon satisfy $J \leq k$). Our experience from Monte Carlo simulations indicates that this is not an “optimal” choice. Note also that the future inputs are already used in the projection matrix $U_{k|L+1}^{\perp}$. Hence, it make sense to use only past inputs as instruments, i.e. the choice $U_{0|J}$. It can also be shown (Verhagen, 1994) that by using only past inputs as instruments, only the deterministic part of the model can be recovered.

Past outputs are assumed to be uncorrelated with future noise. This gives a first constraint on the discrete time instant k, i.e. $k \geq J$. We have

$$\lim_{K \to \infty} \frac{1}{K} E_{k|L+1} Y_{0|J}^T = 0 \quad \forall \; k \geq J \tag{3.25}$$

This statement can be proved from Equations (3.16) and (3.20). By incorporating past outputs as instruments we are also able to recover the stochastic part of the model. Note that the states which are exited from the known inputs are not necessarily the same as those which are exited from the unknown process noise variables. It is necessary that all states are exited from both known and unknown inputs and that they are observable from the output, in order to identify them.

Hence, the following past inputs and past outputs instrumental variable matrix is recommended to remove future noise from the model

$$W_1 = \begin{bmatrix} Y_{0|J} \\ U_{0|J} \end{bmatrix} \in \mathbb{R}^{J(m+r)\times K} \quad \forall \; J \geq 1 \tag{3.26}$$

A consistent equation for $\tilde{A}$ is then given by the following autonomous matrix equation

$$Z_{k+1|L} = \tilde{A} Z_{k|L} \qquad \forall \quad k \geq J \tag{3.27}$$

where

$$Z_{k+1|L} \overset{\text{def}}{=} \tfrac{1}{K} Y_{k+1|L} U^{\perp}_{k|L+1} W_i^T \in \mathbb{R}^{mL\times ni} \tag{3.28}$$

$$Z_{k|L} \overset{\text{def}}{=} \tfrac{1}{K} Y_{k|L} U^{\perp}_{k|L+1} W_i^T \in \mathbb{R}^{mL\times ni} \tag{3.29}$$

Equation (3.27) is consistent because W_i, given by Equations (3.23) and (3.24), satisfies Equation (3.22). See also Equation (3.25).

We can now prove that the column space of the matrix $Z_{k|L}$ coincides with the column space of the extended observability matrix O_L, when the identification (*future*) horizon parameter L is chosen great enough to observe all states and the *past* horizon parameter J is chosen adequately. Using Equations (3.16) and (3.29) with the past inputs and past outputs instrumental variable matrix gives

$$Z_{k|L} = \frac{1}{K} Y_{k|L} U^{\perp}_{k|L+1} W_1^T = O_L X_k U^{\perp}_{k|L+1} \frac{1}{K} W_1^T \quad \in \mathbb{R}^{mL\times J(m+r)} \tag{3.30}$$

Assume that both the row and column dimensions of $Z_{k|L}$ are greater or equal to the number of states, i.e., $Lm \geq n$ and $J(m+r) \geq n$, and that L is chosen such that the system is observable. The dimension of the column space of the left hand side matrix must be equal to the system order, i.e. $\text{rank}(X_k) = n$. Hence,

$$\text{rank}(Z_{k|L}) = \text{rank}(\frac{1}{K} Y_{k|L} U^{\perp}_{k|L+1} W_1^T) = \text{rank}(O_L X_k U^{\perp}_{k|L+1} \frac{1}{K} W_1^T) = n \tag{3.31}$$

The row constraints have a theoretical lower limit. From system theory we know that a number of $L \geq n - \text{rank}(D) + 1$ observations of the output is sufficient in order to observe the states of a linear system, Kalman, Falb and

Arbib (1969), p. 37. However, the theoretical lower limit is the ceiling function $L = \lceil n/m \rceil$, defined as the integer ratio n/m rounded towards plus infinity.

From the column dimension we must ensure that the past horizon J for defining the instrumental variable matrix must satisfy $J(m+r) \geq n$. Hence, the theoretical lower limit is $J = \lceil n/(m+r) \rceil$.

The maximum system order which can be specified by the user for a specified choice of the parameter L is $n = Lm$. In this case the observability matrix can be estimated as the column space of $Z_{k|L}$ only if the past horizon parameter J is chosen such that $J(m+r) \geq Lm$. A reasonable choice is therefore $J = L$.

Monte Carlo simulation experiments shows a relatively constant statistical behavior of the estimates as a function of the past horizon parameter J. Hence, we simply recommend putting $J = L$.

We have the following algorithm for analysis and modeling of system dynamics.

Algorithm 3.4.1 (System order, n, and the pair (D, A))
Given the positive integer parameters L and J and the matrices $Z_{k+1|L}$ and $Z_{k|L}$ with $k \geq J$ which satisfy the autonomous matrix equation

$$Z_{k+1|L} = \tilde{A} Z_{k|L} \tag{3.32}$$

where

$$\tilde{A} = O_L A (O_L^T O_L)^{-1} O_L^T \tag{3.33}$$

and O_L is the extended observability matrix for the pair (A, D).
1. The system order n
Determine the Singular Value Decomposition (SVD)

$$Z_{k|L} = USV^T \tag{3.34}$$

where $U \in \mathbb{R}^{mL \times mL}$, $S \in \mathbb{R}^{mL \times ni}$ and $V \in \mathbb{R}^{ni \times ni}$ are given by

$$U = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \quad S = \begin{bmatrix} S_n & 0 \\ 0 & S_2 \end{bmatrix} \quad V = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \tag{3.35}$$

where $S_n \in \mathbb{R}^{n \times n}$ and n is the number of "non-zero" singular values of $Z_{k|L}$, which is equal to the system order. n is determined by inspection of the "non-zero" diagonal elements of S or SS^T. The term $U_2 S_2 V_2^T$ represents the error by estimating the system order as the n first principal singular values.
2. The extended observability matrix O_L for the pair (D, A)
The (extended) observability matrix can be taken directly as the first left part in U, i.e. U_1. We have

$$O_L = U(1:Lm, 1:n) = U_1 \tag{3.36}$$

3. The system matrix A
The system matrix A can be determined as

$$A = O_L^T Z_{k+1|L} V \begin{bmatrix} S_n^{-1} \\ 0 \end{bmatrix} = U_1^T Z_{k+1|L} V_1 S_n^{-1} \tag{3.37}$$

4. The system output matrix D
The matrix D can be taken as the $m \times n$ upper sub-matrix in the observability matrix O_L, i.e.

$$D = U(1:m, 1:n) \tag{3.38}$$

5. The extended system matrix $\tilde{A}$
We have

$$\tilde{A} = O_L A (O_L^T O_L)^{-1} O_L^T = Z_{k+1|L} V_1 S_n^{-1} U_1^T \tag{3.39}$$

△

Note that it can be difficult to estimate the system order as the non-zero or large singular values of the matrix $Z_{k|L}$ when the signal to noise ratio is "small". In practice, the model order is chosen by a trial and error procedure combined with both model validation and physical knowledge of the data generating process. A procedure for chosing the model order from some statistical test is possibile, see e.g. Ch. 9.6 in Aoki (1990).

We have chosen $O_L = U_1$ in Step 2 for simplicity, because we have $O_L^T O_L = I_{n\times n}$ in this case. This gives an output normal realization when $L \to \infty$. The algorithm can also be formulated with the choice $O_L = U_1 S_n^{\frac{1}{2}}$ which gives a balanced realization when $L \to \infty$. $O_L^T O_L$ is equal to the observability grammian as L tends to infinity because, in this case, DA^{L-1} tends to zero. A third choice is $O_L = U_1 S_n$ which gives an input normal realization. These definitions are attributable to Moore (1981). These choices only represent different scalings of the column space and give similar state space model matrices. The scaling does not affect the statistical properties of the algorithm.

3.4.2 Realization of the Deterministic Sub-system

At this stage the system matrices A and D as well as the extended observability matrix O_L are known, see Section 3.4.1. In order to obtain a complete realization for the deterministic part of the system we need to compute the system matrices B and E.

There are many alternatives for extracting the B and E matrices. See e.g. Section 3.5.2 for an alternative to the method presented below.

A consistent estimate of the $\tilde{B}$ matrix can be computed from

$$\tilde{B} \frac{1}{K} U_{k|L+1} U_{k|L+1}^T = \frac{1}{K} (Y_{k+1|L} - \tilde{A} Y_{k|L}) U_{k|L+1}^T \tag{3.40}$$

where $\tilde{A}$ is determined from Algorithm 3.4.1, because

$$\lim_{K\to\infty} \tilde{C} \frac{1}{K} E_{k|L+1} U_{k|L+1}^T = 0 \quad \forall\ k \geq 0 \tag{3.41}$$

For known data and system matrices A and D, Equation (3.40) can be written as an over determined set of linear equations in the unknown system matrices B and E .

We will in the rest of this section discuss the simpler solution when $U_{k|L+1}U_{k|L+1}^T$ is non-singular. The matrix $\tilde{B}$ can be computed directly from Equation (3.40) in this case. We refer to Section 3.5.2 for the case when $U_{k|L+1}U_{k|L+1}^T$ is singular. The system matrices B and E can be extracted from $\tilde{B}$. E is given directly as the lower right $m \times r$ sub-matrix of $\tilde{B}$. We have

$$E = \tilde{B}(m(L-1)+1 : mL, rL+1 : r(l+1)) \tag{3.42}$$

B is given as a function of the block columns in $\tilde{B}$ and the matrices $\tilde{A}$ and O_L. $\tilde{A}$ and O_L are known from Algorithm 3.4.1. Define

$$\tilde{B}_i = \tilde{B}(1 : mL, r(i-1)+1 : ri) \quad 1 \le i \le L+1 \tag{3.43}$$

as block column number i of $\tilde{B}$. We have

$$O_L B = \sum_{i=1}^{L+1} \tilde{A}^{i-1}\tilde{B}_i \tag{3.44}$$

and

$$B = \sum_{i=1}^{L+1} A^{i-1}(O_L^T O_L)^{-1} O_L^T \tilde{B}_i \tag{3.45}$$

One strategy for recursively extracting the B, E, H_L^d and $O_L B$ matrices is as follows.

Algorithm 3.4.2 *Determination of H_L^d, $O_L B$, E and B from known $\tilde{B}$, O_L and A.*

```
$\tilde{A} = O_L A (O_L^T O_L)^{-1} O_L^T$
$E_L = \tilde{B}_{L+1}$
$E = E_L(m(L-1)+1 : mL, 1 : m)$
for $i = 1, \dots, L$
    $E_{L-i} = \tilde{B}_{L-i+1} + \tilde{A} E_{L-i+1}$
    if $i < L$
        $E_{L-i}(1 : r(L-i-1), 1 : r) = zeros(r(L-i-1), r)$
    end
end
$O_L B = E_0$
$B = (O_L^T O_L)^{-1} O_L^T O_L B$
```

△

The lower block triangular Toeplitz matrix H_L^d is given by the block columns $E_1, \cdots, E_L$ according to Equations (3.14) and (3.5).

3.4.3 Realization of the Stochastic Sub-system

This section is mainly concerned with the problem of identifying the stochastic part of the system. However, for natural reasons most of the results in Sections 3.4.1 and 3.4.2 are extended and resolved in parallel.

The ESSM, Equation (3.7), gives a relation between the future data matrices and the SSM matrices. The following extension of the ESSM can be proved

$$\begin{aligned} Y_{k|L+1} &= O_{L+1}A^kO_J^\dagger Y_{0|J} \\ &+ \begin{bmatrix} O_{L+1}C_k^d & H_{L+1}^d \end{bmatrix} \begin{bmatrix} U_{0|k} \\ U_{k|L+1} \end{bmatrix} \\ &- \begin{bmatrix} O_{L+1}A^kO_J^\dagger H_J^d & 0_{(L+1)m\times(k+L+1-J)r} \end{bmatrix} \begin{bmatrix} U_{0|J} \\ U_{J|k+L+1-J} \end{bmatrix} \\ &+ \begin{bmatrix} O_{L+1}C_k^s & H_{L+1}^s \end{bmatrix} \begin{bmatrix} E_{0|k} \\ E_{k|L+1} \end{bmatrix} \\ &- \begin{bmatrix} O_{L+1}A^kO_J^\dagger H_J^s & 0_{(L+1)m\times(k+L+1-J)r} \end{bmatrix} \begin{bmatrix} E_{0|J} \\ E_{J|k+L+1-J} \end{bmatrix} \end{aligned} \tag{3.46}$$

Assume $k = J$ for the sake of simplicity. Then we have

$$\begin{aligned} Y_{J|L+1} &= O_{L+1}A^JO_J^\dagger Y_{0|J} \\ &+ \begin{bmatrix} O_{L+1}C_J^d - O_{L+1}A^JO_J^\dagger H_J^d & H_{L+1}^d \end{bmatrix} \begin{bmatrix} U_{0|J} \\ U_{J|L+1} \end{bmatrix} \\ &+ \begin{bmatrix} O_{L+1}C_J^s - O_{L+1}A^JO_J^\dagger H_J^s & H_{L+1}^s \end{bmatrix} \begin{bmatrix} E_{0|J} \\ E_{J|L+1} \end{bmatrix} \end{aligned} \tag{3.47}$$

This last equation nicely shows the connection between the past and future data matrices and the unknown model matrices. As we will see, it also yields some important results. We have the following projection

$$\begin{aligned} & Y_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^\perp \\ &= \begin{bmatrix} O_{L+1}C_J^s - O_{L+1}A^JO_J^\dagger H_J^s & H_{L+1}^s \end{bmatrix} \begin{bmatrix} E_{0|J} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^\perp \\ E_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^\perp \end{bmatrix} \end{aligned} \tag{3.48}$$

The projection of future inputs, past inputs and outputs onto the null-space of future noise is equal to $E_{J|L+1}$. We have separated the deterministic part of the system from the data by the projection in Equation (3.48). Hence we have the following theorem concerning the stochastic part of the system

Theorem 3.4.1 (Realization of the Toeplitz matrix H^s_{L+1})
The lower triangular Toeplitz matrix H^s_{L+1} with Markov parameters for the stochastic sub-system is given by the projection of past inputs, past outputs, and future inputs onto the null-space of the future outputs, i.e.

$$Z^s_{J|L+1} \overset{\text{def}}{=} Y_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp}$$

$$= P^s_{L+1} E_{0|J} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} + H^s_{L+1} E_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} \tag{3.49}$$

where

$$P^s_{L+1} = O_{L+1} C^s_J - O_{L+1} A^J O^\dagger_J H^s_J \tag{3.50}$$

and when $N \to \infty$, then

$$Z^s_{J|L+1} \overset{\text{def}}{=} Y_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} = P^s_{L+1} E_{0|J} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} + H^s_{L+1} E_{J|L+1} \tag{3.51}$$

which is a linear problem in H^s_{L+1}.

In order to see that this is a linear problem in the unknown matrix H^s_{L+1}, one only needs to write up the equations obtained by using the structure of H^s_{L+1}, Equation (3.15), and a partition of the last term on the right hand side into sub-matrices.

This result with proof is to our knowledge new. However, the same solution resulting from a QR decomposition was presented in Di Ruscio (1995b).

From Theorem 3.4.1 and Equation (3.47) we can immediately state the following important result. The extended observability matrix can be recovered from the column space of the matrix on the left hand side of the following matrix Equation (3.52).

Theorem 3.4.2 (Realization of the extended observability matrix O_{L+1})

Given the following matrix equation

$$Z_{J|L+1} \overset{\text{def}}{=} (Y_{J|L+1} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) U^\perp_{J|L+1} =$$

$$O_{L+1} \begin{bmatrix} A^J O^\dagger_J & C^d_J - A^J O^\dagger_J H^d_J & C^s_J - A^J O^\dagger_J H^s_J \end{bmatrix} \begin{bmatrix} Y_{0|J} \\ U_{0|J} \\ E_{0|J} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} U^\perp_{J|L+1} \tag{3.52}$$

then the column space of the matrix $Z_{J|L+1}$ coincides with the column space of the extended observability matrix O_{L+1} and the system order n of the SSM is given as the dimension of the column space.

The proof of Theorem 3.4.2 and Equation (3.52) is simple. From Theorem 3.4.1, Equation (3.49), we have

$$Y_{J|L+1} - Y_{J|L+1}/\underline{\tilde{Y}} = P^s_{L+1}E_{0|J} - P^s_{L+1}E_{0|J}/\underline{\tilde{Y}} + H^s_{L+1}E_{J|L+1} \tag{3.53}$$

where

$$\underline{\tilde{Y}} \overset{\text{def}}{=} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{3.54}$$

and for the sake of simplicity the other matrices are defined according to Equations (3.49) and (3.50). Substituting Equation (3.53) into equation (3.47) in order to remove the stochastic term $H^s_{L+1}E_{J|L+1}$ gives

$$Y_{J|L+1}/\underline{\tilde{Y}} = O_{L+1}A^JO^\dagger_JY_{0|J} + P^d_{L+1}U_{0|J} + P^s_{L+1}E_{0|J}/\underline{\tilde{Y}} + H^d_{L+1}U_{J|L+1} \tag{3.55}$$

Hence, an extra projection $U^\perp_{J|L+1}$ on the right hand side is necessary in order to remove the deterministic term $H^d_{L+1}U_{J|L+1}$ and in order to recover the extended observability matrix. The matrix on the right hand side of Equation (3.52) is proportional with the extended observability matrix O_{L+1}. Hence, the column space of the matrix $Z_{J|L+1}$ coincides with O_{L+1}. The dimension of the column space of the matrix $Z_{J|L+1}$ is the order n of the SSM. **QED**.

From Theorem 3.4.2 we immediately have the following corollary concerning the system dynamics.

Corollary 3.4.1 (Identification of system dynamics)
From $Z_{J|L+1}$ defined in Theorem 3.4.2, Equation (3.52) we have the following relationship

$$Z_{J+1|L} = \tilde{A}Z_{J|L} \tag{3.56}$$

where

$$\tilde{A} \overset{\text{def}}{=} O_LA(O^T_LO_L)^{-1}O^T_L$$

and where $Z_{J+1|L}$ is the L last $(m \times K)$ block rows in $Z_{J|L+1}$ and $Z_{J|L}$ is the L first $(m \times K)$ block rows in $Z_{J|L+1}$, i.e.

$$Z_{J+1|L} \overset{\text{def}}{=} (Y_{J+1|L}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix})U^\perp_{J|L+1}, \quad Z_{J|L} \overset{\text{def}}{=} (Y_{J|L}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix})U^\perp_{J|L+1}$$

Moreover, the column space of $Z_{J|L}$ coincides with the column space of the extended observability matrix O_L and the dimension of the column space is equal to the order n of the SSM. A realization of n, O_L, A, D and $\tilde{A}$ is determined by using Equation (3.56) in combination with Algorithm 3.4.1.

The matrix on the left hand side of Equation (3.52) can be shown to be equivalent to

$$\begin{aligned} Z_{J|L+1} &= Y_{J|L+1} W_c \\ W_c &\overset{\text{def}}{=} U^{\perp}_{J|L+1} W_1^T (W_1 U^{\perp}_{J|L+1} W_1^T)^{-1} W_1 U^{\perp}_{J|L+1} \end{aligned} \tag{3.57}$$

Comparing Equation (3.57) with Equation (3.30) shows that these matrices are related. We have shown that the column space of both equations coincides with the column space of the extended observability matrix. The difference can be viewed as a different column weighting matrix W_c, on the right hand side of Equation (3.57). Equation (3.30) can be viewed as a special case factorization of equation (3.57) with $k = J$ and $W_c = U^{\perp}_{J|L+1} W_1^T / K$. Later in Section 3.6, we will show that this last Equation (3.57) is extremely important and useful in order to partly compare other subspace identification methods.

From Theorem 3.4.1 we have the following result concerning the stochastic part of the system.

Theorem 3.4.3 (Realization of Δ and $\tilde{C}$)
Assume that the number of system input and output observations, $N \to \infty$. Define

$$Z^s_{J|L+1} \overset{\text{def}}{=} Y_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} \tag{3.58}$$

then we have

$$Z^s_{J+1|L} = \tilde{A} Z^s_{J|L} + \tilde{C} E_{J|L+1} \tag{3.59}$$

and

$$Z^s_{J+1|L} (Z^s_{J|L})^{\perp} = \tilde{C} E_{J|L+1} (Z^s_{J|L})^{\perp}$$

where

$$\tilde{C} \overset{\text{def}}{=} \begin{bmatrix} O_L C & H^s_L \end{bmatrix} - \tilde{A} \begin{bmatrix} H^s_L & 0_{Lm \times m} \end{bmatrix}$$

The covariance matrix of the innovations Δ is estimated directly from the column space of the left hand side matrix Equation (3.58), e.g.

$$Z^s_{J+L|1} (Z^s_{J|L})^{\perp} = F E_{J+L|1} (Z^s_{J|L})^{\perp} = F E_{J+L|1} \tag{3.60}$$

and

$$\Delta = F F^T \tag{3.61}$$

when

$$E_{J+L|1} E^T_{J+L|1} = I_{m \times m}$$

△

The importance of Theorem 3.4.3 is that it shows that the innovations covariance matrix can be estimated directly from the column space of the data matrix, Equation (3.58). The factorization of the left hand side matrix of Equation (3.60) into the product $FE_{J+L|1}$ and $E_{J+L|1}E^T_{J+L|1} = I_{m\times m}$ can be performed by the QR decomposition. See also Section 3.5. The Kalman filter gain matrix C can be extracted when A and D is known.

Finally, we have the following Theorem 3.4.4 for the realization of the deterministic part of the system.

Theorem 3.4.4 (Realization of $\tilde{B}$)
Given

$$Z^d_{J|L+1} \overset{\text{def}}{=} Y_{J|L+1}/ \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{3.62}$$

then we have

$$Z^d_{J+1|L} = \tilde{A}Z^d_{J|L} + \tilde{B}U_{J|L+1} \tag{3.63}$$

and

$$Z^d_{J+1|L}(Z^d_{J|L})^\perp = \tilde{B}U_{J|L+1}(Z^d_{J|L})^\perp \tag{3.64}$$

where

$$\tilde{B} \overset{\text{def}}{=} \begin{bmatrix} O_L B & H^d_L \end{bmatrix} - \tilde{A}\begin{bmatrix} H^d_L & 0_{Lm\times r} \end{bmatrix}$$

Theorem 3.4.4 is proved from Equation (3.55). Remark that the SSM matrix E can be estimated directly from Equation (3.64). This is so because of the structure of the ESSM matrix $\tilde{B}$.

All the projections in this section, Equations (3.49), (3.52) and (3.62) can be effectively computed from a QR decomposition, either directly from the projections defined in this section, or as will be shown in Section (3.5).

We will conclude this section by pointing out the relationship between the data matrices and the projection matrices. The data matrix with future system outputs and the projection matrices, (3.49) and (3.62) are related as

$$Y_{J|L+1} = Z^d_{J|L+1} + Z^s_{J|L+1}$$

Z^d represents the outputs from the deterministic part of the system. Z^s represents the outputs from the stochastic part of the system. $Y_{J|L+1}$ is the data matrix with future outputs from the combined deterministic and stochastic system.

3.5 Implementation with QR Decomposition

We will here use the QR decomposition in order to compute the column space of the projection matrices derived in Section 3.4. The QR decomposition is also used in the subspace identification methods by Verhagen (1994) and Van Overschee and De Moor (1994).

Define the following QR decomposition

$$\frac{1}{\sqrt{K}}\tilde{Y} = RQ =$$

$$\frac{1}{\sqrt{K}}\begin{bmatrix} U_{k|L+1} \\ W_i \\ Y_{k|L} \\ Y_{k+1|L} \end{bmatrix} = \begin{bmatrix} R_{11} & 0 & 0 & 0 \\ R_{21} & R_{22} & 0 & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix} \tag{3.65}$$

where

$$R \quad \in \mathbb{R}^{(r(L+1)+ni+2mL)\times(r(L+1)+ni+2mL)} \tag{3.66}$$

$$Q \quad \in \mathbb{R}^{(r(L+1)+ni+2mL)\times K} \tag{3.67}$$

Note that this decomposition could perhaps more precisely have been defined as a lower Left, Q-orthogonal (LQ) decomposition. See Golub and Van Loan (1983) for the computation.

The QR decomposition can be viewed as a data compression step. The data matrix $\tilde{Y}$ which usually has a large number of columns is compressed to a usually much smaller lower triangular matrix R which contains all relevant information of the system for which the data was generated. As we will show, the orthogonal Q matrix is not needed in the algorithm.

Note that the first $(L-1)m$ rows in $Y_{k+1|L}$ are equal with the last $(L-1)m$ rows in $Y_{k|L}$. This means that $Y_{k+1|L}$ can be substituted for $Y_{k+L|1}$ in the QR decomposition, Equation (3.65). This is utilized in the efficient implementation of the **DSR** algorithm. However, for the sake of simplicity we will present the results according to Equation (3.65).

By definition, the instrumental variable matrix W_i is uncorrelated with $E_{k|L+1}$. We can therefore remove the noise matrix $E_{k|L+1}$ from Equation (3.7) by post-multiplying with $\frac{1}{K}W_i^T$. We have from (3.65) that

$$\lim_{K\to\infty}\frac{1}{K}E_{k|L+1}W_i^T = (\lim_{K\to\infty}\frac{1}{K}E_{k|L+1}\begin{bmatrix} Q_1^T & Q_2^T \end{bmatrix})\begin{bmatrix} R_{21}^T \\ R_{22}^T \end{bmatrix} = 0 \tag{3.68}$$

Post-multiplying Equation (3.7) with $[Q_1^T Q_2^T]$, using (3.68) and substituting for the corresponding R_{ij} sub-matrices from (3.65) gives

$$\begin{bmatrix} R_{41} & R_{42} \end{bmatrix} = \tilde{A}\begin{bmatrix} R_{31} & R_{32} \end{bmatrix} + \tilde{B}\begin{bmatrix} R_{11} & 0 \end{bmatrix} \tag{3.69}$$

which gives one matrix equation for $\tilde{A}$ and one for both $\tilde{A}$ and $\tilde{B}$. We will in the next Sections 3.5.1 and 3.5.2 show how the order n and the system quadruple (A, B, D, E) are computed from (3.69).

The stochastic part of the system, defined by the matrices C and Δ, is computed from

$$R_{43} - \tilde{A}R_{33} = \tilde{C}E_{k|L+1}Q_3^T \tag{3.70}$$

$$R_{44} = \tilde{C}E_{k|L+1}Q_4^T \tag{3.71}$$

This will be shown in Section 3.5.3.

Note that the QR decomposition compresses the possible large data matrices into a number of (smaller) matrices which contain the information of the system. It is also interesting that the matrices (information) which define the deterministic part (A, B, D, E) and the stochastic part (Δ, C) are separated by the QR decomposition. The user must specify the parameter $k \geq J$ in Equation 3.65. See Figure 3.1 for a definition of the horizons involved. We recommend putting $k = J$. The matrix W_1 with $l = 0$, Equation (3.24), is recommended for W_i in 3.65.

3.5.1 Realization of A and D

We have from Equation (3.69) that

$$R_{42} = \tilde{A}R_{32} \tag{3.72}$$

and we choose

$$Z_{k+1|L} = R_{42} \tag{3.73}$$

$$Z_{k|L} = R_{32} = USV^T \tag{3.74}$$

in Algorithm (3.4.1) in order to determine A, D and the extended observability matrix O_L. The system order is determined by inspection of the dominant singular values of S or SS^T.

Note that the first $(L-1)m$ rows in R_{42} are equal to the last $(L-1)m$ rows in R_{32}. This is utilized in the efficient implementation of the **DSR** algorithm, in order to reduce the computational work.

Note also that if $\tilde{A}$ is computed as the projection of R_{32} onto R_{42} then $\tilde{A}$ takes a special canonical form. This is due to the common rows.

The A matrix may also be determined as follows. From the extended observability matrix we have

$$O_1 \overset{\text{def}}{=} O_L(1 : (L-1)m, 1 : n) \tag{3.75}$$

$$O_2 \overset{\text{def}}{=} O_L(m+1 : Lm, 1 : n) \tag{3.76}$$

then

$$A = (O_1^T O_1)^{-1} O_1^T O_2 \tag{3.77}$$

However, we must put $L =: L+1$ in this case if the extended observability matrix is estimated as the left singular vectors in (3.74). This will increase the computational work. This last method is the so called *shift invariance* method for computing the transition matrix A from the extended observability matrix, Kung (1978). The *shift invariance* method is used in the subspace algorithms (**N4SID**) by Van Overschee and De Moor (1994) and (**MOESP**) by Verhagen (1994). The parameter which defines the number of block rows in the **N4SID** and **MOESP** algorithms is denoted I. This parameter is related to the **DSR** parameter L as $I = L+1$. This is one of the differences between the **DSR** algorithm and the **N4SID** and **MOESP** algorithms.

This means that **N4SID** and **MOESP** computes a number $Im = (L+1)m$ of singular values. However, the system order can only be chosen according to Lm of these singular values, i.e. the maximum system order which can be chosen for a specified parameter $I = L+1$ is $n = Lm$. For comparison, the **DSR** algorithm computes only a number Lm of singular values and the maximum system order which can be chosen for a specified parameter L is $n = Lm$. Hence, the **DSR** algorithm seems to be more consistent with respect to choosing the system order as the number of non-zero singular values.

The *shift invariance* method can be included in the **DSR** algorithm, but at a higher computational expense. This strategy can be described as follows. The extended observability matrix can be estimated from the column space of the matrix formed from R_{32} and R_{42}. Compute the SVD

$$\begin{bmatrix} R_{32} \\ R_{42} \end{bmatrix} = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} S_n & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = \begin{bmatrix} U_{11} S_n V_1^T \\ U_{21} S_n V_1^T \end{bmatrix}$$

We then have that A and D are determined from the *shift invariance* method, e.g. from the left singular vectors as follows

$$O_L = U_{11} \tag{3.78}$$

$$O_L A = U_{21} \tag{3.79}$$

$$A = (U_{11}^T U_{11})^{-1} U_{11}^T U_{21} \tag{3.80}$$

$$D = U_{11}(1:m, 1:n) \tag{3.81}$$

3.5.2 Realization of B and E

We have from Equation (3.69) that

$$\tilde{B} R_{11} = R_{41} - \tilde{A} R_{31} \tag{3.82}$$

$\tilde{B}$ can be determined directly from (3.82) if the input u is persistently exiting of order $L+1$. R_{11} is non singular in this case. We have

$$\tilde{B} = (R_{41} - \tilde{A} R_{31}) R_{11}^T (R_{11} R_{11}^T)^{-1} \tag{3.83}$$

The B and E matrices are then extracted from $\tilde{B}$ as pointed out in Section 3.4.2.

At this stage, the system order is identified (Algorithm 3.4.1). It is possible to determine B and E if the input is only persistently exiting of order $p+1$ where $L_{\text{min}} \leq p \leq L$, directly without recomputing the algorithm with $L = L_{\text{min}}$ or $L = p$. The minimal observability index, for a given system order, is $L_{\text{min}} = n - \text{rank}(D) + 1$ when $n \geq \text{rank}(D)$ and $L_{\text{min}} = 1$ when $n \leq \text{rank}(D)$.

Define

$$\begin{aligned} O_p &= O_L(1:mp, 1:n) && (3.84)\\ \tilde{A}_p &= O_p A (O_p^T O_p)^{-1} O_p^T && (3.85)\\ R_{11}^p &= R_{11}(1:r(p+1), 1:r(p+1)) && (3.86)\\ R_{31}^p &= R_{31}(1:mp, 1:r(p+1)) && (3.87)\\ R_{41}^p &= R_{41}(1:mp, 1:r(p+1)) && (3.88) \end{aligned}$$

We then have

$$\tilde{B}_p R_{11}^p = R_{41}^p - \tilde{A}_p R_{31}^p \tag{3.89}$$

This result is a consequence of Equation (3.7) with L substituted with p. Note that the minimal observability index, for a given system order, is $p = n - \text{rank}(D) + 1$ when $n \geq \text{rank}(D)$. The B and E matrices are then extracted from $\tilde{B}_p$ as shown in Section 3.4.2 and Algorithm 3.4.2 with L substituted with p.

3.5.3 Realization of C and Δ

Corollary 3.5.1 (Realization of C and Δ)
Consider the lower left triangular matrix R, determined by the QR decomposition in Equation (3.65). An estimate of the square root of the innovations covariance matrix is given by the $m \times m$ lower right sub-matrix of R, i.e.

$$F = R_{44}(m(L-1)+1:mL, m(L-1)+1:mL) \tag{3.90}$$

and the estimate of the innovations covariance matrix is

$$\Delta = FF^T \tag{3.91}$$

Furthermore, when $J > 1$, then an estimate of the Kalman filter gain matrix C can be computed from

$$O_L CF = R_{43}(1:mL, 1:m) \tag{3.92}$$

If F is non-singular, then we have

$$C = (O_L^T O_L)^{-1} O_L^T R_{43}(1:mL, 1:m) F^{-1} \tag{3.93}$$

An estimate of the lower left block triangular Toeplitz matrix H_L^s for the stochastic subsystem (D, A, CF, F) is given by

$$H_L^s = R_{43}(1:mL, m+1:m(L+1)) \tag{3.94}$$

The estimate of the lower left block triangular Toeplitz matrix for the stochastic subsystem (D, A, C, I), according to Equation (3.15), can be formed from the block columns $F_1, \cdots, F_L$ which can be computed from

$$\begin{bmatrix} F_1F & F_2F & \cdots & F_LF \end{bmatrix} = R_{43}(1 : mL, m+1 : m(L+1)) \tag{3.95}$$

△

The stochastic subsystem is identified separately from the deterministic subsystem. The necessary separation into deterministic and stochastic subsystems are implicitly done by the QR decomposition.

The first $(L-1)m$ rows in $Y_{k+1|L}$ are equal to the $(L-1)m$ last rows in $Y_{k|L}$. Q_4 is uncorrelated with $U_{k|L+1}$, W_i, $Y_{k|L}$ and with the $(L-1)m$ first rows in $Y_{k+1|L}$. The first $(L-1)m$ rows in $Y_{k+1|L}Q_4^T$ and $E_{k|L+1}Q_4^T$ are therefore zero. We then have from (3.65) and the structure of $\tilde{C}$, given by (3.13) and (3.15), that

$$Y_{k+1|L}Q_4^T = \tilde{C}E_{k|L+1}Q_4^T = R_{44} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & F \end{bmatrix} \tag{3.96}$$

where $R_{44} \in \mathbb{R}^{Lm \times Lm}$. Hence, the square root of the innovations noise process covariance matrix is estimated directly as the $m \times m$ lower left matrix, denoted F, in the lower triangular matrix R from the QR decomposition, Equation (3.65). Note that F also is lower left triangular and can be compared to a Cholesky factorization of Δ. This result is believed to be of some importance. The result (3.91) is then clarified.

The matrices Q_i, $i = 1, 2, 3, 4$, are orthogonal matrices and we have

$$\frac{1}{K}E_{k|L+1}Q_3^T = \begin{bmatrix} F & 0 & \cdots & 0 \\ 0 & F & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & F \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{(L+1)m \times Lm} \tag{3.97}$$

Another strategy is then to compute OCF from Equations (3.70) and (3.97), and an algorithm similar to Algorithm 3.4.2. This is formulated in the following Corollary 3.5.2.

Corollary 3.5.2 (Realization of C)
Given the sub-matrices R_{43} and R_{33} from the QR decomposition in Equation (3.67) and the ESSM transition matrix $\tilde{A}$. Define according to Equation (3.70)

$$\tilde{CF} \overset{\text{def}}{=} R_{43} - \tilde{A}R_{33} \tag{3.98}$$

The matrix O_LCF can then be computed from $\tilde{CF}$, e.g. by a procedure similar to Algorithm 3.4.2.

△

This strategy is found from Monte Carlo simulations to be the best one when the past horizon parameter is $J = 1$, but no significant difference is found when $J > 1$.

3.5.4 Special Remarks

One advantage of the QR implementation of the algorithm is that potential ill-conditioning of the covariance matrices are concentrated in a certain triangular matrix. This ill-conditioning usually results from ill-conditioned noise processes (process noise and measurements noise) and from rounding-off errors. Note that the triangular matrix R is the square root of the covariance matrix ($H = \frac{1}{K}\tilde{Y}\tilde{Y}^T$ where $\tilde{Y}$ is defined in (3.65)) and that the triangular matrix is computed without ever computing the covariance matrix. The method can therefore be defined as numerically stable.

The QR decomposition is not unique. The R matrix is post-multiplied by a diagonal permutation matrix E such that $R := RE$ has positive diagonal elements. The diagonal elements of E are equal to the sign of the corresponding diagonal elements of R which was the result from the QR decomposition. Note also that $Q := EQ$ and $EE = I$. This gives a more unique coordinate system for the estimated (A, B, D, E) quadruple. This scaling is also one of the reasons for the simple solutions for the C and Δ matrices in Section 3.5.3. The scaling ensures that the diagonal blocks of (3.97) gets the same sign.

Note that common rows in the data matrices $Y_{k|L}$ and $Y_{k+1|L}$ can be removed prior to obtaining the QR decomposition in equation (3.67). It is also clear from the above that the orthogonal Q matrix ($QQ^T = I$) is not needed in the algorithm. This will reduce the computational effort considerably. In fact, the QR factorization works on a data matrix of size only $r(2L+1)+m(2L+1)\times K$ and not of size $r(2L+1)+3mL\times K$ as indicated in (3.65).

Another strategy for determining R is to first compute $H = \frac{1}{K}\tilde{Y}\tilde{Y}^T$ where $\tilde{Y}$ is defined in (3.65) and then the SVD, $H = USV^T$, followed by a QR decomposition of $US^{\frac{1}{2}}$ in order to obtain the lower triangular matrix, R. This strategy reduced the number of flops and increased the accuracy of the R matrix when MATLAB was used for the computations. However, no significant difference in the estimated models was observed. This strategy can be numerically ill-conditioned due to possible rounding errors when forming the product of rows in $\tilde{Y}$ with columns in $\tilde{Y}^T$ in order to compute the correlation matrix $\tilde{Y}\tilde{Y}^T$. This strategy is therefore not recommended.

Equation (3.82) is defined by the triangular factors for Equation (3.40) with $k \geq J$.

$$\tilde{B}U_{k|L+1}U_{k|L+1}^T = Y_{k+1|L}U_{k|L+1}^T - \tilde{A}Y_{k|L}U_{k|L+1} \tag{3.99}$$

It is also possible to extract the triangular factors for

$$\tilde{B}U_{0|L+1}U_{0|L+1}^T = Y_{1|L}U_{0|L+1}^T - \tilde{A}Y_{0|L}U_{0|L+1} \tag{3.100}$$

directly from the QR decomposition (3.65) when $k = J = L$. The first block row in $U_{L|L+1}$ is equal to the last block row in $U_{0|L+1}$. Hence, the matrices in Equation (3.100) can be defined from the lower triangular R matrix, equation (3.65). It is therefore natural to choose

$$\tilde{B} \left[\begin{array}{cc} R_{11} & R_{11}^0 \end{array} \right] = \left[\begin{array}{cc} R_{41} & R_{41}^0 \end{array} \right] - \tilde{A} \left[\begin{array}{cc} R_{31} & R_{31}^0 \end{array} \right] \tag{3.101}$$

for defining $\tilde{B}$ and an equation for computing the B and E system matrices. Equation (3.101) consists of the triangular factors for both $\frac{1}{K} U_{k|L+1} U_{k|L+1}^T$ and $\frac{1}{K} U_{0|L+1} U_{0|L+1}^T$. Equation (3.101) has effect for systems where the input signal is poor with frequencies, but gives no additional effect compared to (3.82) for e.g. white noise inputs.

Note that the stochastic part of the model is determined from QR and SV Decompositions only. The Markov parameters and the square root of the innovations covariance matrix are determined from a QR decomposition only. The Kalman filter gain matrix is determined from the Markov parameters and the extended observability matrix. No matrix Lyapunov or non-linear matrix Riccati equations have to be solved.

The method has in this work been illustrated for systems which are not strictly proper (only proper), i.e. the case when $E \neq 0$ in the underlying model (3.2). The method can also be implemented to handle proper systems, i.e. systems where E is known to be zero. This can be done by deleting the last block row in $U_{k|L+1}$ and the last block column in $\tilde{B}$, see Equation (3.7).

3.6 Comparison with Existing Algorithms

A comparison with the **DSR** algorithm and three different subspace algorithms will be given, namely **N4SID**, Van Overschee and De Moor (1994), **CVA**, Larimore (1983), (1990) and **PO-MOESP**, Verhagen (1994).

The first and common step in subspace identification algorithms is to estimate the extended observability matrix from the column space of a known data matrix. We will therefore concentrate our discussion on the similarities and differences in which these methods estimate the extended observability matrix. We will only briefly discuss how the system matrices are estimated by the different methods.

It is shown in Van Overschee and De Moor (1995) that the different methods are related through certain row and column weightings with the **N4SID** data matrix as the key matrix.

We will below present a different approach, with the matrix $Z_{J|L+1}$ defined in Section 3.4.3, Theorem 3.4.2 and Equation (3.57), as the key matrix.

Multiplying Equation (3.57) from left with an extra row weighting matrix W_r and using the SVD as discussed in Algorithm 3.4.1 gives

$$W_r Z_{J|L+g} = W_r Y_{J|L+g} W_c = U_1 S_n V_1^T + U_2 S_2 V_2^T \tag{3.102}$$

where g is an integer parameter and

$$W_c \overset{\text{def}}{=} U^{\perp}_{J|L+1} W_1^T (W_1 U^{\perp}_{J|L+1} W_1^T)^{-1} W_1 U^{\perp}_{J|L+1}$$

The extended observability matrix can be estimated as, e.g.

$$O_{L+g} = W_r^{-1} U_1 \tag{3.103}$$

where W_r is a non-singular matrix. The matrix W_c is symmetric and can be written as

$$W_c \overset{\text{def}}{=} \underbrace{U^{\perp}_{J|L+1} W_1^T (W_1 U^{\perp}_{J|L+1} W_1^T)^{-\frac{1}{2}}}_{W_c^2} \underbrace{(W_1 U^{\perp}_{J|L+1} W_1^T)^{-\frac{1}{2}} W_1 U^{\perp}_{J|L+1}}_{(W_c^2)^T} \tag{3.104}$$

From the above factorization of the matrix W_c we have at least four matrices (W_c^i, $i = 1, \cdots, 4$) which all have essentially the same column space as W_c, i.e.

$$\begin{aligned}
W_c^1 &= U^{\perp}_{J|L+1} W_1^T (W_1 U^{\perp}_{J|L+1} W_1^T)^{-1} W_1 U^{\perp}_{J|L+1} \\
W_c^2 &= U^{\perp}_{J|L+1} W_1^T (W_1 U^{\perp}_{J|L+1} W_1^T)^{-\frac{1}{2}} \\
W_c^3 &= U^{\perp}_{J|L+1} W_1^T (W_1 U^{\perp}_{J|L+1} W_1^T)^{-1} K \\
W_c^4 &= U^{\perp}_{J|L+1} W_1^T \tfrac{1}{K} \\
W_c^5 &= U^{\perp}_{J|L+1}
\end{aligned}$$

The matrix W_c^5 is sufficient only for purely deterministic systems and is shown for the sake of completeness.

These column weighting matrices are used in the **DSR** algorithm which is presented in this work. The past horizon parameter J is usually chosen as $J \geq L$. The parameter $g = 0$ in Equation (3.102) and the row weighting matrix is the $Lm \times Lm$ identity matrix, denoted $W_r = I_{Lm}$. Algorithm 3.4.1 is used in order to identify the extended observability matrix O_L from the column space of the matrix $Z_{J|L}$.

We will now illustrate the **DSR** algorithm's similarity and difference to two published algorithms, **CVA** by Larimore (1990) and **PO-MOESP** by Verhagen (1994).

3.6.1 PO-MOESP

The **PO-MOESP** algorithm in Verhagen (1994) estimates the extended observability matrix O_{L+1} from Equations (3.102) and (3.103) with the following matrices

$$\left.\begin{aligned}
W_c &= U^{\perp}_{L+1|L+1} W_1^T (W_1 U^{\perp}_{L+1|L+1} W_1^T)^{-1} W_1 U^{\perp}_{L+1|L+1} \\
W_r &= I_{(L+1)m} \\
g &= 1
\end{aligned}\right\} \textbf{PO-MOESP} \tag{3.105}$$

From Theorem 3.4.2 and the factorization in Equation (3.104) we conclude that the two algorithms **PO-MOESP** and **DSR** estimate the extended observability matrix from a known data matrix which essentially has the same column space. The only difference is that **PO-MOESP** estimates the extended observability matrix O_{L+1} (of larger size than **DSR** does) in order to use the shift invariance method for extracting the system matrix A as explained in Section 3.5.1.

Using the triangular factors from the QR decomposition in Section 3.5, we have

$$R_{32}R_{22}^T(R_{22}R_{22}^T)^{-1}R_{22}Q_2 = (U_1S_nV_1^T + U_2S_2V_2^T)Q_2 \tag{3.106}$$

where the orthogonal matrix Q_2 is not needed because the column space can be estimated as the matrix U_1.

A major difference is that the **PO-MOESP** algorithm does not estimate the stochastic part of the model. **DSR** estimates the Kalman gain and innovations covariance matrix directly from the data as shown in Sections 3.4.3 and 3.5.3.

3.6.2 Canonical Variate Analysis (CVA)

The **CVA** algorithm in Larimore (1990) estimates the extended observability matrix O_{L+1} from Equations (3.102) and (3.103) with the following matrices

$$\left.\begin{array}{l} W_c = U_{L+1|L+1}^{\perp}W_1^T(W_1U_{L+1|L+1}^{\perp}W_1^T)^{-\frac{1}{2}} \\ W_r = (Y_{L+1|L+1}U_{L+1|L+1}^{\perp}Y_{L+1|L+1}^T)^{-\frac{1}{2}} \\ g = 1 \end{array}\right\}\textbf{CVA} \tag{3.107}$$

As we can see, the column weighting matrix W_c used by the **CVA** algorithm fits into the factorization in Equation (3.104). A difference is that the **CVA** algorithm uses a row weighting matrix W_r.

The only difference is that the **DSR** algorithm takes the SVD of a matrix of size only $Lm \times J(r+m)$ where usually $J = L$, in order to identify O_L. The other methods take the SVD of a matrix of size $(L+1)m \times J(r+m)$, in order to identify O_{L+1}. This is to separate out the sub-matrices O_L and O_LA from O_{L+1}. See also Section 3.5.1 for a discussion.

From Theorem 3.4.2 and the factorization in Equation (3.104) we conclude that the two algorithms **CVA** and **DSR** essentially have the same column space.

An interpretation of the **CVA** algorithm is that the system order is estimated as the number of principal angles between the matrix $Y_{L+1|L+1}U_{L+1|L+1}^{\perp}$ and $W_1U_{L+1|L+1}^{\perp}$ different from $\pi/2$. The principal angles can be effectively computed using the SVD, see e.g. Van Overschee (1995), p. 29 and Golub and Van Loan (1989), p. 428.

By using the triangular factors as shown in Section 3.5 we get the following method for computing the principal angles

$$(R_{32}R_{32}^T + R_{33}R_{33}^T)^{-\frac{1}{2}}R_{32}R_{22}^T(R_{22}R_{22}^T)^{-\frac{1}{2}} = U_1S_nV_1^T + U_2S_2V_2^T \tag{3.108}$$

The system order is here identified as the number of singular values equal to one.

The next step in the **CVA** algorithm is to define a memory which defines a valid sequence of system states. The system matrices can then be estimated from a least squares problem.

3.6.3 N4SID

The **N4SID** algorithm is different. The following weighting matrices are used

$$\left.\begin{array}{l} W_c = U^{\perp}_{L+1|L+1} W_1^T (W_1 U^{\perp}_{L+1|L+1} W_1^T)^{-1} W_1 \\ W_r = I_{(L+1)m} \\ g = 1 \end{array}\right\} \textbf{N4SID} \tag{3.109}$$

The column weighting matrix used in the **N4SID** algorithm does not generally have the same column space as W_c or any of the column weighting matrices which result from Equation (3.104) and Theorem 3.4.2. This is possibly the reason for why **N4SID** gives bad results for deterministic input signals.

In Viberg (1995) it is pointed out that only the difference between the **PO-MOESP** and **N4SID** algorithms is the extra projection $U^{\perp}_{L+1|L+1}$. However, it is also claimed that *the resulting subspace estimates should therefore have very similar properties*. From Theorem 3.4.2 and the above discussion we conclude that this conclusion in Viberg (1995) is wrong. This is illustrated in example 2, Section 3.7.2.

From the above discussion we have the following relationship between the column weighting matrix W_c in Equation (3.104) and the matrix W_c in (3.109) used by **N4SID**.

$$W_c = W_c^{\text{N4sid}} U^{\perp}_{L+1|L+1} \tag{3.110}$$

In Theorem 3.4.2, Equations (3.52) and (3.55), it is proved that the extra projection $U^{\perp}_{L+1|L+1}$ is necessary in order to remove the deterministic term $H^d_{L+1} U_{L+1|L+1}$ and establish the data matrix $Z_{L+1|L+1}$ which has the same column space as the extended observability matrix. See also Example 2, Section 3.7.2, for an illustration.

The **N4SID** method computes the SVD of the data matrix defined in Equation (3.102) with the above matrices W_r and W_c, Equation (3.109). If the triangular factors as shown in Section 3.5 are used then we have

$$R_{32} R_{22}^T (R_{22} R_{22}^T)^{-1} [R_{21} R_{22}] \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = (U_1 S_n V_1^T + U_2 S_2 V_2^T) \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} \tag{3.111}$$

The orthogonal matrices Q_1 and Q_2 are not used. The system order is identified as the number of non-zero singular values and the extended observability matrix O_{L+1} is estimated from the column space. The rest of the **N4SID** algorithm can briefly be described as follows. From the extended observability matrix

O_{L+1} and the system data, a valid sequence of system states are estimated and a least squares problem is solved in order to construct the system matrices A, B, D, E. The covariance matrices for the process and measurements noise are then identified from a residual, and a Riccati equation is solved in order to construct the Kalman filter gain matrix C and the innovations covariance matrix Δ.

The **DSR** method does not use state sequences and the Kalman gain matrix C and the innovations covariance matrix Δ are constructed directly from the data, without recursions of non-linear matrix Riccati equations.

3.6.4 Main Differences and Similarities

Both of the algorithms **N4SID** and **CVA** estimate in the first instance a sequence of states. When the states are known, the state space model matrices can be determined by simple linear regression. Both methods must solve a matrix Riccati equation in order to identify the Kalman filter gain and the innovations covariance matrices.

The **DSR** algorithm is based on first writing up an extended state space model (ESSM) where the unknown states are eliminated from the problem. The ESSM shows us the relationship between the known data matrices and the SSM matrices. Hence, the **DSR** algorithm does not have any problems with unknown states; unknown initial values etc. The state space model matrices are then extracted from the ESSM. **DSR** estimates the Kalman gain and innovations covariance matrices (stochastic part of the model) directly from the data, without recursions of non-linear matrix equations, e.g. the Riccati equation.

The **PO-MOESP** algorithm does not estimate the stochastic part of the model.

We have shown that the **CVA**, **PO-MOESP** and **DSR** algorithms gives consistent estimates of the extended observability matrix. The algorithms fit into the same Theorem 3.4.2. We have shown that the **N4SID** algorithm in general does not give consistent estimates of the extended observability matrix. However, it will give consistent results if an extra projection of future inputs is included. From the above discussion we have the following relationship between the column weighting matrix W_c in Equation (3.104) and the matrices used by **PO-MOESP**, **CVA** and **N4SID**. See (3.105), (3.107) and (3.109).

$$W_c = W_c^{\text{PO-MOESP}} = W_c^{\text{CVA}} (W_c^{\text{CVA}})^T = W_c^{\text{N4sid}} U_{L+1|L+1}^{\perp}$$

These are the most important similarities and differences between the method presented in this work and previously published methods.

3.7 Numerical Examples

3.7.1 Example 1: Monte Carlo Simulation

A single input single output (SISO) system with one state is chosen to compare the algorithm presented in this paper, entitled **DSR** (**D**eterministic and **S**tochastic System Identification and **R**ealization), with two other algorithms, **CVA** (which stands for Canonical Variate Analysis, Larimore (1983), (1990)) and the prediction error method implemented in the MATLAB function **ARMAX** (i.e., in the system identification toolbox, Ljung (1991)).

$$\begin{aligned} x_{k+1} &= 0.9x_k + 0.5u_k + 0.6e_k && (3.112) \\ y_k &= 1.0x_k - 1.0u_k + e_k && (3.113) \end{aligned}$$

Three types of input signals were used. One input equal to a sum of four sinusoid signals, u^1, one input equal to a white noise signal with unit covariance, u^2, and one equal to a sine, u^3.

$$\begin{array}{ll} u^1 & u_k = 0.2(\sin(\frac{k}{25}) + \sin(\frac{k}{10}) + \sin(\frac{k}{5}) + \sin(k)) \\ u^2 & \text{White noise, unit covariance} \\ u^3 & u_k = \sin(k) \end{array}$$

For each input the time series (y_k, u_k) was generated by simulating the model with 100 different white noise sequences e_k, also with unit variance.

The **DSR** algorithm parameter L was changed from 1 to 5 and the **CVA** parameter I from 2 to 6. For each L and I, the mean and standard deviation of the parameters of the 100 different estimated models are presented in Tables 3.1 to 3.6. The results obtained by the **ARMAX** algorithm are also shown in the tables. See Ljung (1991) for the description of the parameters $nn = [1, 2, 1, 0]$ which is used as arguments for **ARMAX**.

The true deterministic system quadruple is denoted $(a, b, d, e) := (0.9, 0.5, 1, -1)$ and the deterministic steady state gain and deterministic zero are denoted $H^d(1) = 4.0$ and $p_d(1) = 1.4$, respectively. The parameters in the stochastic part of the model are $(c, \Delta) := (0.6, 1)$. The stochastic steady state gain and stochastic zero are denoted $H^s(1) = 7.0$ and $p_s(1) = 0.3$, respectively. The signal to noise ratio is approximately 0.4, hence, the identification problem is not simple.

The **CVA** algorithm sometimes estimated systems with negative (b,d) parameters, i.e. sometimes an estimated quadruple (a,b,d,e) and sometimes (a,-b,-d,e). This happened with the algorithm parameter $I = 3$ and with a random input signal. It is believed that this can be avoided by using a scaling similar to that presented in Section 3.5.4.

The results are very good for both the **DSR** and the **CVA** algorithms, see Tables 3.1 and 3.2. There are small differences in the estimated models for both methods when N is large, see Tables 3.1 and 3.2. This indicates that the asymptotic statistical distribution of the parameter estimates is the same. The example indicates that both the **DSR** and **CVA** algorithms are insensitive to variation in the number of block rows.

The **DSR** algorithm is found to be marginally better than the **CVA** when the number of samples is small (for this example and with $N = 200$ and $N = 500$ samples) see Tables 3.3 to 3.6.

It is also interesting to observe that the results from **DSR** are as good as the results from the **ARMAX** function, even for a simple SISO system. This indicates that **DSR** gives asymptotically statistical optimal results for this example, both for purely deterministic inputs (u^1 and u^3) and stochastic input sequences (u^2). Note that the prediction error method (**ARMAX** function) is based on iterative optimization but that the **DSR** algorithm is based on SVD and QR decompositions only. Prediction error methods are rather complicated for MIMO systems while the **DSR** algorithm is very simple.

Figures 3.2 to 3.8 are included in order to illustrate the asymptotic mean and variance properties of the **DSR** algorithm for varying horizon parameters L and J. The figures illustrate that the algorithm is numerically robust and that the estimates are consistent and fairly insensitive for the parameters L and J.

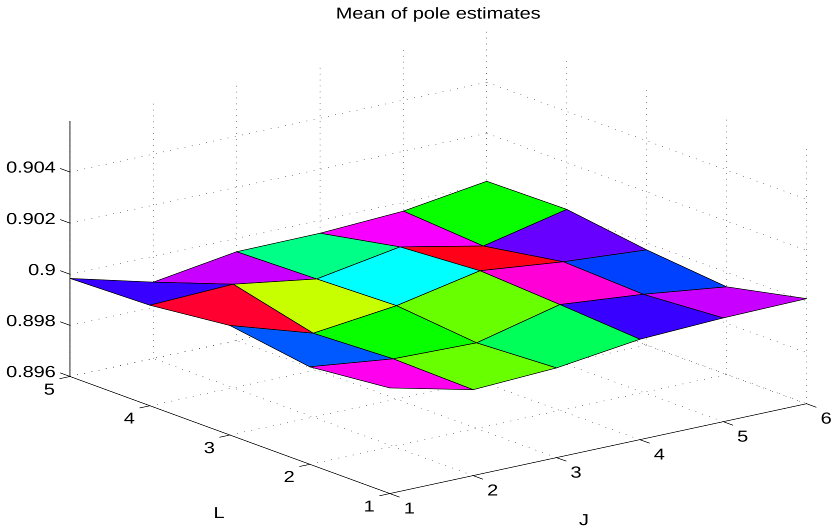


Figure 3.2: The mean value of the pole estimates for a Monte Carlo simulation with varying past horizon parameter J and identification parameter L. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sum of 5 sinusoid signals (u^1). The maximum estimate was 0.9003 for $L = 3$ and $J = 4$. The minimum estimate was 0.8990 for $L = 5$ and $J = 2$. The closest estimate to the actual pole ($a = 0.9$) was 0.89997 for $L = 3$ and $J = 5$.

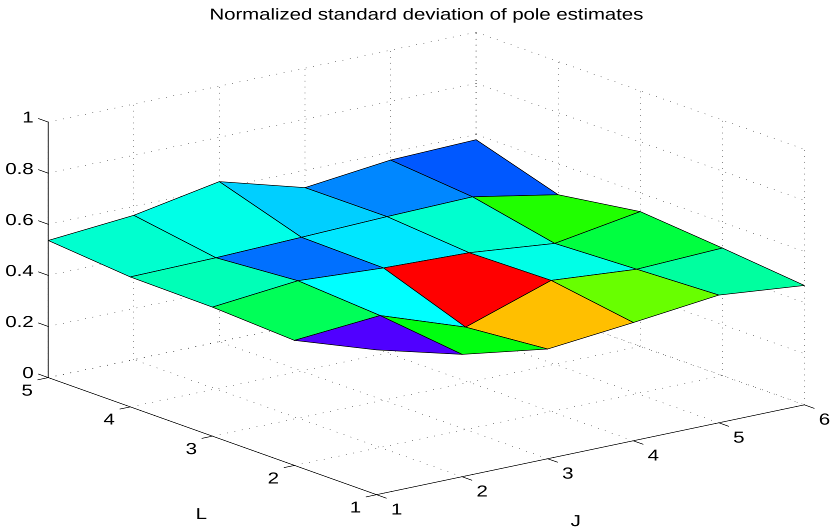


Figure 3.3: The standard deviation multiplied (normalized) with $N^{1/2}$ of the pole estimates for a Monte Carlo simulation with varying past horizon parameter J and varying identification parameter L. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sum of 5 sinusoid signals (u^1). The maximum standard deviation of the pole estimates was $0.6265/N^{1/2}$ at $L = 5$ and $J = 3$. The minimum standard deviation was $0.4005/N^{1/2}$ at $L = 2$ and $J = 3$.

Table 3.1: Mean of the parameters of 100 different estimated models: input type u^1, $N = 10000$. The models was scaled so that $d = 1$.

Alg., Par.	a	b	e	$p_d(1)$	$H^d(1)$	c	Δ	$p_s(1)$	$H^s(1)$
DSR, L=1	0.8998	0.497	-1.002	1.396	3.9598	0.6088	1.0045	0.2910	7.0910
DSR, L=2	0.8995	0.497	-1.002	1.398	3.9516	0.6003	0.9989	0.2992	6.9867
DSR, L=3	0.8998	0.497	-1.002	1.398	3.9604	0.5992	0.9985	0.3006	6.9940
DSR, L=4	0.8996	0.497	-1.002	1.398	3.9572	0.5992	0.9981	0.3004	6.9875
DSR, L=5	0.8998	0.496	-1.002	1.398	3.9644	0.5989	0.9977	0.3009	6.9976
CVA, I=2	0.8996	0.497	-1.002	1.398	3.9545	0.6008	1.0046	0.2988	7.0015
CVA, I=3	0.8996	0.497	-1.003	1.398	3.9572	0.5995	1.0000	0.3001	6.9849
CVA, I=4	0.8998	0.495	-1.003	1.396	3.9471	0.5990	0.9994	0.3008	6.9930
CVA, I=5	0.8996	0.498	-1.001	1.400	3.9714	0.5992	0.9993	0.3003	6.9830
CVA, I=6	0.8996	0.498	-1.002	1.399	3.9659	0.5993	0.9993	0.3003	6.9872
ARMAX	0.8995	0.498	-1.002	1.399	3.9604	0.5995		0.3000	6.9769

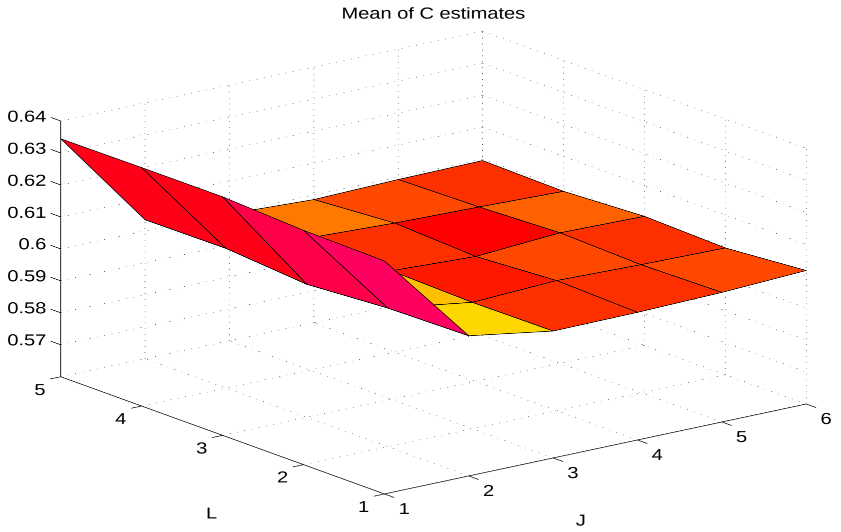


Figure 3.4: The mean value of the Kalman filter gain estimates for a Monte Carlo simulation with varying past horizon parameter J and identification parameter L. The estimates are computed by the algorithm in Corollary 3.5.1. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sinusoid signal (u^3). The actual value is $C = 0.6$. The mean of the estimates at the flat region described by $3 \leq J \leq 6$ and $1 \leq L \leq 5$ was 0.6000. This indicates that the estimates are consistent for past horizon parameters $J > 2$ independently of the choice of L, but clearly biased for $J = 1$. The estimates for $J = 2$ should be further investigated.

Table 3.2: Standard deviation of the parameters of 100 different estimated models: input type u^1, $N = 10000$.

Alg., Par.	a	b	e	$p_d(1)$	$H^d(1)$	c	Δ	$p_s(1)$	$H^s(1)$
DSR, L=1	0.0054	0.027	0.068	0.045	0.3020	0.0101	0.0153	0.0130	0.3170
DSR, L=2	0.0052	0.026	0.068	0.043	0.3063	0.0099	0.0152	0.0121	0.3021
DSR, L=3	0.0054	0.026	0.068	0.043	0.3190	0.0102	0.0152	0.0128	0.3211
DSR, L=4	0.0056	0.026	0.068	0.044	0.3236	0.0105	0.0151	0.0135	0.3309
DSR, L=5	0.0059	0.026	0.068	0.043	0.3328	0.0111	0.0151	0.0145	0.3447
CVA, I=2	0.0052	0.027	0.068	0.044	0.2959	0.0101	0.0153	0.0124	0.2994
CVA, I=3	0.0051	0.029	0.069	0.047	0.3423	0.0097	0.0152	0.0118	0.3009
CVA, I=4	0.0053	0.033	0.070	0.052	0.3920	0.0096	0.0152	0.0118	0.3155
CVA, I=5	0.0055	0.038	0.070	0.057	0.4456	0.0096	0.0151	0.0119	0.3236
CVA, I=6	0.0058	0.037	0.07	0.056	0.4554	0.0096	0.0152	0.0122	0.3391
ARMAX	0.0051	0.027	0.066	0.042	0.2925	0.0095		0.0115	0.2983

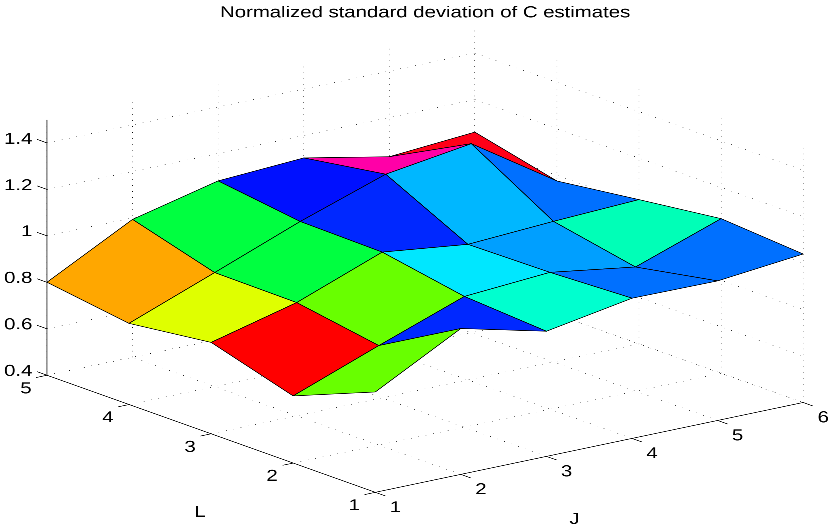


Figure 3.5: The standard deviation multiplied (normalized) with $N^{1/2}$ of the kalman filter gain estimates for a Monte Carlo simulation with varying past horizon parameter J and varying identification parameter L. The estimates are computed by the algorithm in Corollary 3.5.1. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sinusoid signal (u^3). The following parameters is found from the region with consistent estimates. The maximum standard deviation of the pole estimates was $1.2135/N^{1/2}$ at $L = 4$ and $J = 5$. The minimum standard deviation was $0.9344/N^{1/2}$ at $L = 2$ and $J = 5$. The mean of all standard deviations was $1.0315/N^{1/2}$.

Table 3.3: Mean of the parameters of 100 different estimated models: input type u^2, $N = 200$. The models was scaled sa that $d = 1$.

Alg., Par.	a	b	e	$p_d(1)$	$H^d(1)$	c	Δ	$p_s(1)$	$H^s(1)$
DSR, L=1	0.8861	0.507	-0.992	1.405	4.0874	0.6093	0.9961	0.2768	7.0385
DSR, L=2	0.8884	0.502	-0.995	1.399	3.9833	0.6045	0.9699	0.2839	6.9509
DSR, L=3	0.8886	0.502	-0.997	1.398	3.9781	0.6094	0.9476	0.2792	7.0266
DSR, L=4	0.8891	0.5	-0.993	1.399	4.0110	0.6128	0.9218	0.2763	7.1207
DSR, L=5	0.8902	0.5	-0.994	1.399	4.0741	0.6139	0.8991	0.2763	7.2495
CVA, I=2	0.8865	0.508	-0.992	1.406	3.9629	0.6027	1.0102	0.2838	6.8462
CVA, I=3	0.8865	0.503	-0.995	1.399	3.8955	0.6072	0.9953	0.2793	6.8840
CVA, I=4	0.8859	0.500	-0.997	1.396	3.8534	0.6111	0.9916	0.2749	6.9022
CVA, I=5	0.8852	0.499	-0.997	1.394	3.8399	0.6122	0.9844	0.2730	6.9141
CVA, I=6	0.8840	0.501	-0.931	1.396	3.7969	0.6145	0.9801	0.2695	6.8643
ARMAX	0.8864	0.504	-0.994	1.399	3.8848	0.5987	0.9971	0.2877	6.7799

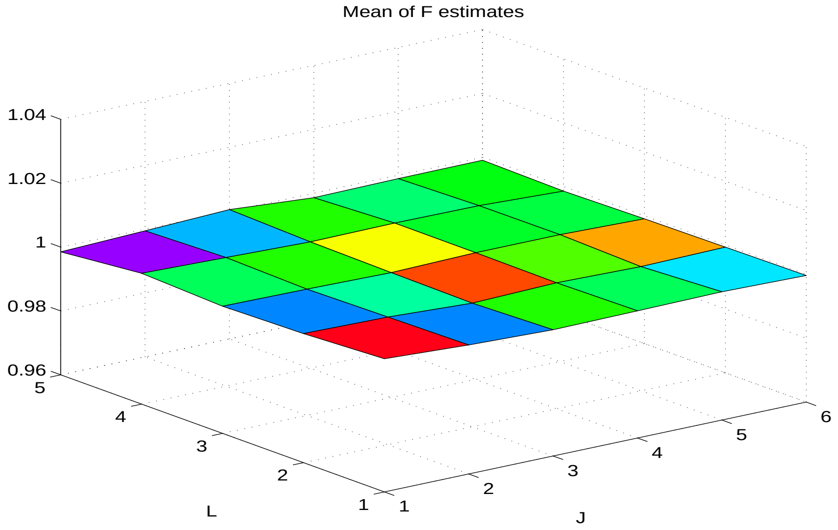


Figure 3.6: The mean value of the square root of the innovations variance estimates F for a Monte Carlo simulation with varying past horizon parameter J and identification parameter L. The estimates are computed by the algorithm in Corollary 3.5.1. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sinusoid signal (u^3). The actual parameter value is $F = 1$. The mean of all the estimates in the figure is 0.9996 with a standard deviation of $7.4 \cdot 10^{-4}$. This indicates that the estimates are consistent.

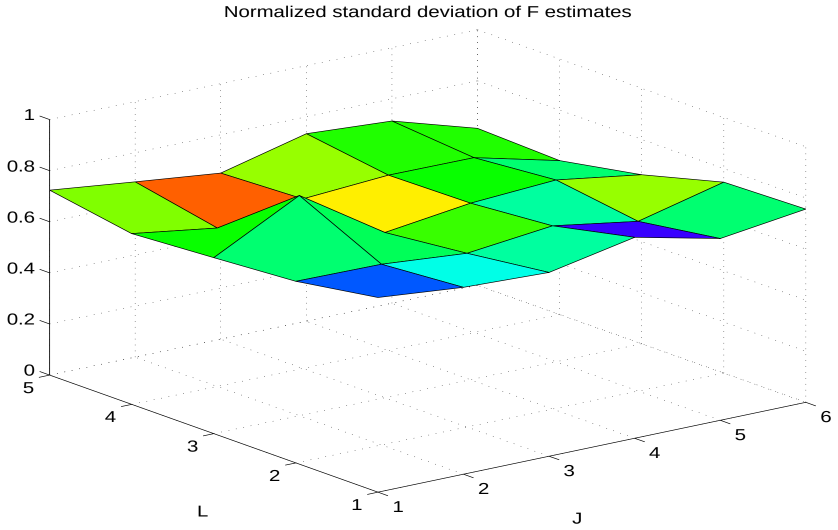


Figure 3.7: The standard deviation multiplied (normalized) with $N^{1/2}$ of the innovations variance estimates F for a Monte Carlo simulation with varying past horizon parameter J and varying identification parameter L. The estimates are computed by the algorithm in Corollary 3.5.1. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sinusoid signal (u^1). The minimum standard deviation was $0.6016/N^{1/2}$ at $L = 4$ and $J = 6$.

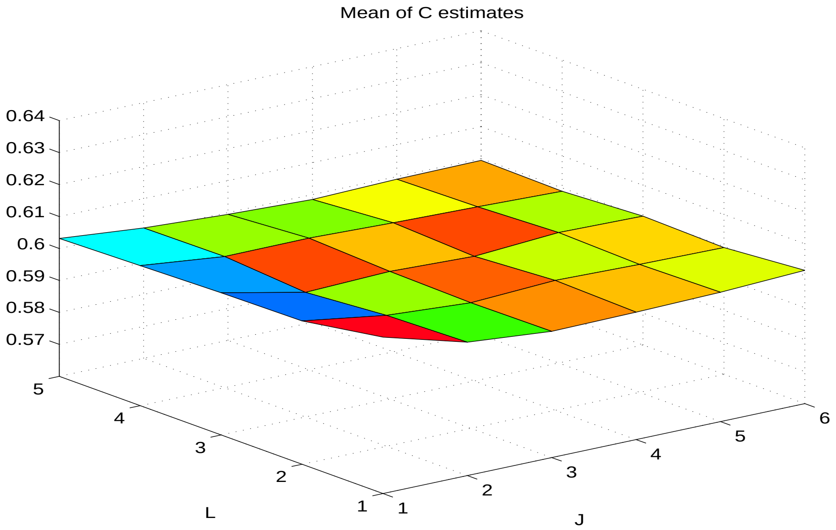


Figure 3.8: The mean value of the Kalman filter gain estimates for a Monte Carlo simulation with varying past horizon parameter J and identification parameter L. The estimates are computed by the algorithm in Corollary 3.5.2. The number of samples in each simulation was $N = 15000$ and the number of simulations for each pair L, J was 100. The input was a sinusoid signal (u^3). The actual value is $C = 0.6$. The bias for $J = 1$ as shown in Figure 3.4 when the estimates are computed as in Corollary 3.5.2 is approximately eliminated.

Table 3.4: Standard deviation of the parameters of 100 different estimated models: input type u^2, $N = 200$.

Alg., Par.	a	b	e	$p_d(1)$	$H^d(1)$	c	Δ	$p_s(1)$	$H^s(1)$
DSR, L=1	0.0379	0.096	0.073	0.132	2.3261	0.0101	0.0907	0.0884	2.4984
DSR, L=2	0.0351	0.091	0.072	0.122	1.8766	0.0131	0.0889	0.0915	1.9967
DSR, L=3	0.0357	0.093	0.079	0.123	1.8585	0.0158	0.0887	0.0964	2.0583
DSR, L=4	0.0363	0.095	0.084	0.123	1.9096	0.0183	0.0872	0.1046	2.1884
DSR, L=5	0.0361	0.098	0.089	0.121	1.9666	0.0210	0.0865	0.1095	2.4102
CVA, I=2	0.0354	0.096	0.073	0.133	1.9081	0.0104	0.0936	0.0844	1.9693
CVA, I=3	0.0351	0.099	0.074	0.137	1.8680	0.0106	0.0959	0.0839	2.0161
CVA, I=4	0.0357	0.099	0.077	0.138	1.8757	0.0115	0.0931	0.0871	2.0303
CVA, I=5	0.0367	0.099	0.078	0.135	1.9511	0.0119	0.0951	0.0904	2.1405
CVA, I=6	0.0376	0.098	0.079	0.133	1.8513	0.0123	0.0955	0.0974	2.0864
ARMAX	0.0348	0.092	0.068	0.123	1.7760	0.0723	0.0933	0.0867	1.8969

Table 3.5: Mean of the parameters of 100 different estimated models: input type u^3, $N = 500$. The models was scaled so that $d = 1$.

Alg., Par.	a	b	e	$p_d(1)$	$H^d(1)$	c	Δ	$p_s(1)$	$H^s(1)$
DSR, L=1	0.8952	0.493	-1.001	1.392	4.0591	0.6013	1.0006	0.2939	7.0975
DSR, L=2	0.8938	0.494	-1.002	1.390	3.9881	0.5961	0.9879	0.2977	6.9587
DSR, L=3	0.8934	0.493	-1.003	1.389	3.9382	0.6000	0.9784	0.2934	6.9510
DSR, L=4	0.8931	0.493	-1.002	1.389	3.9376	0.6018	0.9695	0.2914	6.9572
DSR, L=5	0.8936	0.493	-1.002	1.390	3.9738	0.6030	0.9612	0.2906	7.0060
CVA, I=2	0.8944	0.493	-1.002	1.391	3.9908	0.5944	1.0047	0.2999	6.9579
CVA, I=3	0.8933	0.492	-1.001	1.389	3.9558	0.5959	0.9998	0.2974	6.9273
CVA, I=4	0.8931	0.492	-1.000	1.389	3.9105	0.5966	0.9992	0.2965	6.8970
CVA, I=5	0.8930	0.492	-1.000	1.389	3.9064	0.5967	0.9986	0.2963	6.8926
CVA, I=6	0.8931	0.491	-1.999	1.389	3.8941	0.5972	0.9977	0.2960	6.8930
ARMAX	0.8936	0.493	-1.002	1.39	3.9354	0.5952	0.9980	0.2983	6.8954

Table 3.6: Standard deviation of the parameters of 100 different estimated models: input type u^3, $N = 500$.

Alg., Par.	a	b	e	$p_d(1)$	$H^d(1)$	c	Δ	$p_s(1)$	$H^s(1)$
DSR, L=1	0.0277	0.066	0.077	0.102	1.6168	0.0409	0.0644	0.0542	1.4790
DSR, L=2	0.0274	0.066	0.075	0.099	1.6257	0.0442	0.0637	0.0580	1.4551
DSR, L=3	0.0268	0.066	0.077	0.099	1.5491	0.0456	0.0638	0.0583	1.4129
DSR, L=4	0.0272	0.066	0.076	0.100	1.5699	0.0470	0.0626	0.0600	1.4067
DSR, L=5	0.0275	0.067	0.076	0.100	1.6054	0.0479	0.0640	0.0609	1.4341
CVA, I=2	0.0269	0.066	0.076	0.100	1.5547	0.0417	0.0653	0.0542	1.4149
CVA, I=3	0.0275	0.066	0.076	0.099	1.5903	0.0431	0.0642	0.0560	1.4462
CVA, I=4	0.0268	0.067	0.075	0.099	1.5164	0.0415	0.0645	0.0532	1.3767
CVA, I=5	0.0267	0.066	0.075	0.099	1.5091	0.0420	0.0645	0.0539	1.3780
CVA, I=6	0.0262	0.067	0.076	0.101	1.5003	0.0433	0.0660	0.0539	1.3668
ARMAX	0.0260	0.066	0.075	0.099	1.5207	0.0421	0.0643	0.0535	1.3581

3.7.2 Example 2

We will in this example investigate the problem with colored input signals and the **N4SID** algorithm.

Consider the same SISO one state example as in Example 1, Equations (3.112) and (3.113). Two different input signals were chosen, one equal to a pure sinusoid signal, input type u^3, and one equal to a white noise sequence with unit variance, input type u^2. The inputs are the same as defined in Example 1. The number of samples was fixed to $N = 500$. The standard deviation of the innovation was varied from $\Delta^{0.5} = 0$ to $\Delta^{0.5} = 0.01$ in order to investigate the sensitivity for noise. The number of block rows in the data matrices was chosen as $L = 2$.

The extended observability matrix O_3 was estimated from the column space of the matrix $Z_{L+1|L+1}$, Equation (3.102). The dimension of the column space is estimated as the number of "non zero" singular values, see Figure 3.9. We have also introduced the normalized singular value $(s_1 - s_2)/s_1$ as shown in Figure 3.10. This means that when $(s_1 - s_2)/s_1 = 1$ then the number of states is $n = 1$.

The conclusions to be drawn from Figures 3.9 and 3.10 are that the **DSR** algorithm gives reasonable estimates for both the system order and the actual pole (the **CVA** and **PO-MOESP** give essentially the same results for this example) and that the **N4SID** algorithm does not work at all for this system with a pure deterministic sinusoid input signal ($u_k = sin(k)$). However, note that when the input was changed to a white noise sequence (input type u^3) then the two algorithms gave essentially the same singular values as well as pole estimates.

3.7.3 Example 3

A two input two output system with the following model matrices is considered.

$$A = \begin{bmatrix} 1.5 & 1 & 0.1 \\ -0.7 & 0 & 0.1 \\ 0 & 0 & 0.85 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{3.114}$$

$$D = \begin{bmatrix} 3 & 0 & -0.6 \\ 0 & 1 & 1 \end{bmatrix} \quad E = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.115}$$

$$C = \begin{bmatrix} 0 & 0.1 \\ 0.1 & 0 \\ 0 & 0.2 \end{bmatrix} \quad \Delta = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.116}$$

The system have a deterministic zero at -3, i.e. the system is non-minimum phase. The eigenvalues of A are $0.85, 0.75 \pm 0.3708i$.

The deterministic and stochastic gain matrices are given by

$$H^d(1) = \begin{bmatrix} 16 & 15 \\ 2.6667 & -2.5 \end{bmatrix} \quad H^s(1) = \begin{bmatrix} 2.5 & 4.7 \\ -0.25 & 1.1833 \end{bmatrix}$$

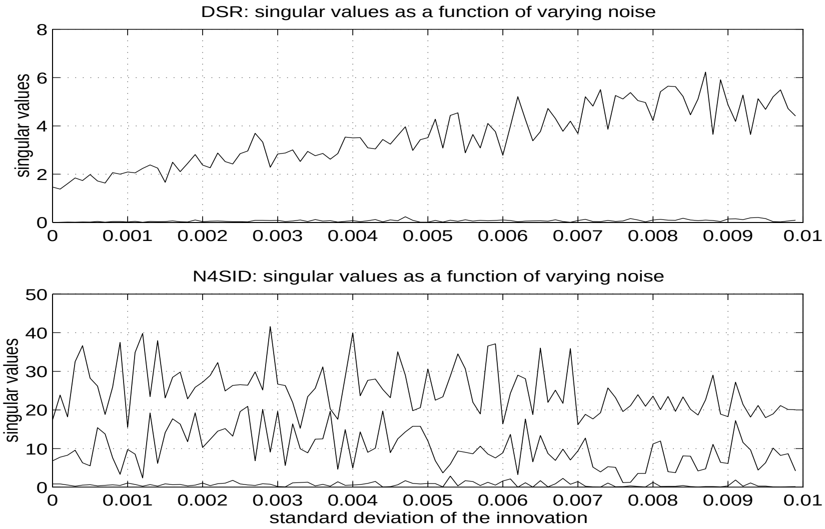


Figure 3.9: The singular values for model order selection as a function of varying innovations noise level. The input to the system was a pure sinusoid signal (input type u^3). The other parameters are as described in Example 2. The singular values from the **DSR** algorithm are shown in the upper figure and for the **N4SID** algorithm in the lower. The actual system order is $n = 1$. As we can see, the system order is fairly well detected by the **DSR** algorithm, while the **N4SID** algorithm does not work at all for this system with a sinusoid input signal.

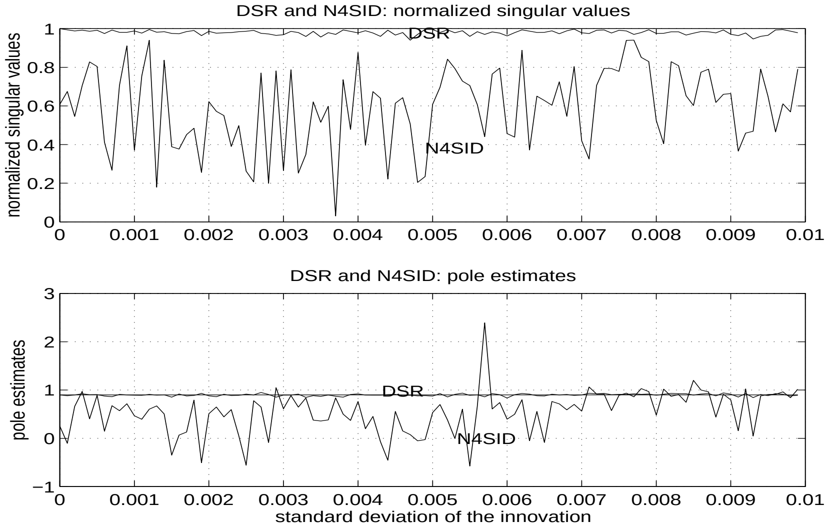


Figure 3.10: This figure shows the normalized singular values $(s_1 - s_2)/s_1$ and the pole estimates as a function of varying innovations noise level for the system in Example 2. The input to the system was a pure sinusoid signal (input type u^3). The other parameters are as described in Example 2. The normalized singular values from both the **DSR** and **N4SID** algorithms are shown in the upper figure. The pole estimates are shown in the lower figure. The actual system order is $n = 1$ and the actual pole is 0.9. As we can see, the system order and the pole is fairly well estimated by the **DSR** algorithm, while the **N4SID** algorithm does not work at all for this system with a sinusoid input signal.

The algorithm gives exact results when $\Delta = 0$. Hence, this result is not presented. The time series y_k, u_k was generated by simulating the model with one particular random noise process e_k with covariance Δ. The input was $u = [u^2 u^1]^T$. The **DSR** parameter was fixed to $L = 6$. The following estimates are obtained by **DSR**.

$$\hat{H}^d(1) = \begin{bmatrix} 15.6575 & 14.6168 \\ 2.5576 & -2.4188 \end{bmatrix} \quad \hat{H}^s(1) = \begin{bmatrix} 2.3625 & 4.5583 \\ -0.2436 & 1.1275 \end{bmatrix}$$

$$\hat{\Delta} = \begin{bmatrix} 1.0531 & -0.0244 \\ -0.0244 & 0.9859 \end{bmatrix}$$

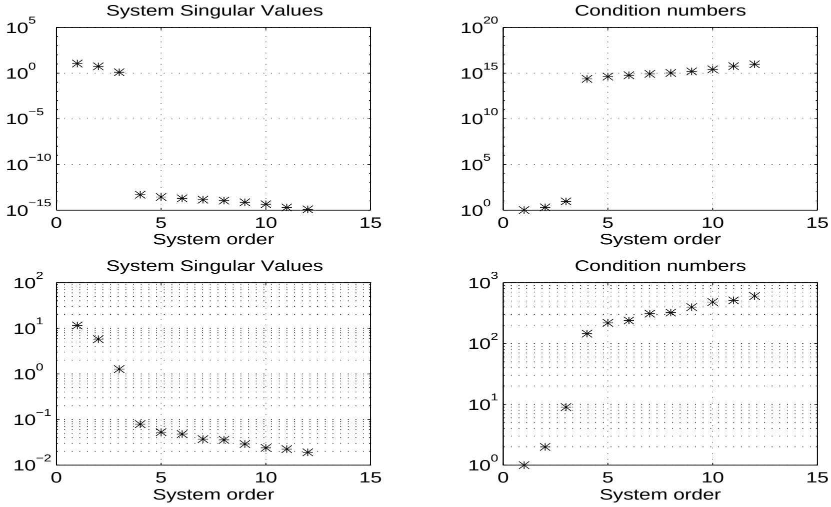


Figure 3.11: Singular values ($s_i, i = 1, ..., 12$) and condition numbers ($\frac{s_1}{s_i}, i = 1, ..., 12$) for the system order selection in Example 3. The noise free deterministic case with $\Delta = 0$, upper figure. The case with $\Delta = I$, lower figure.

3.7.4 Example 4: Modeling the crude oil prize

The idea of this example is to investigate if the crude oil prize can be modeled by a combined deterministic and stochastic model with the USD vs. NKR exchange rate treated as an exogenous input variable. It is assumed that the exchange rate is managed and exactly known. There are of course other variables than the exchange rate which influences upon the crude oil prize. These variables are in this example assumed to be unknown stochastic variables.

A number of 504 observations of the crude oil prize ($[\frac{NKR}{barrel}]$) in the period from 1993 to 1995 is shown in Figure 3.13. The data is from Norges Bank (the Bank of Norway). The crude oil prize at observation k is defined as y_k and also defined as an output variable. At the same time instants the USD vs. NKR exchange rate ($[\frac{USD}{NKR}]$) in the period from 1993 to 1995 is shown in Figure 3.14. The exchange rate at day k is defined as u_k and also defined as an input variable.

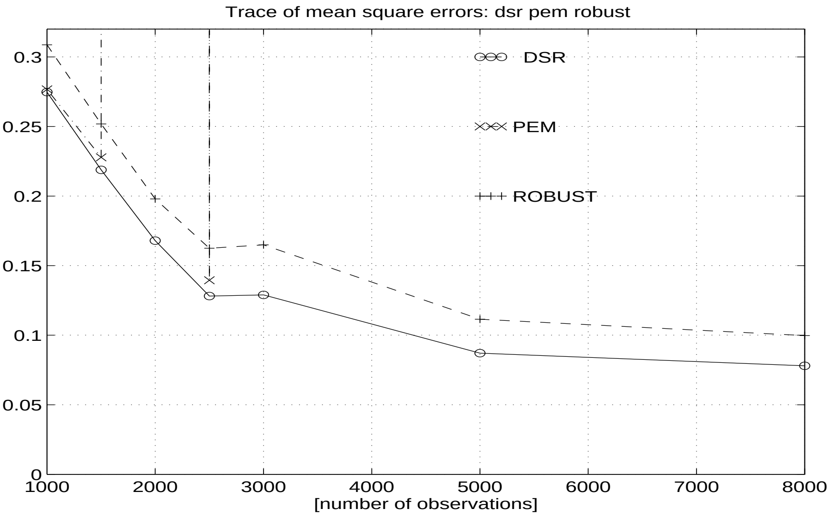


Figure 3.12: Results from a Monte Carlo simulation in order to investigate the accuracy of the innovation estimates. The model in Example 3 is simulated 100 times with varying number of samples. The **DSR** estimates are computed from Theorem 3.4.3. The **PEM** algorithm converged only for number of samples $N < 3000$.

The time series are separated into a trend and a series of variations. Define

$$dy_k = y_k - y_0(k) \tag{3.117}$$

$$du_k = u_k - u_0(k) \tag{3.118}$$

where dy_k is the variation in the crude oil prize around the trend $y_0(k)$ and du_k is the variation in the exchange rate around the trend $u_0(k)$. The exchange rate trend is simply chosen as the first observation, i.e. $u_0(1)$. Inspection of the series shows that there is an immediately fall in the crude oil prize at observation number $k = 368$ of about 18 $[\frac{NKR}{barrel}]$. This drop in the crude oil prize can hardly be explained by changes in the exchange rate (u_k). The explainnation is more likely an economical decision (from a OPEC meeting etc.). The trend is therefore chosen as shown in Figure 3.13. However, note that it is not necessary to remove trends from the time series when the algorithm for Subspace Identification (4SID), (Di Ruscio (1995) algorithm name DSR) is used.

The 200 first observations in the time series dy_k and du_k was used as inputs to the DSR algorithm. The algorithm was also executed with the parameter $L = 5$ (number of block rows in the extended observability matrix). The singular values and condition numbers used for estimating the order of the state space model is shown in Figure 3.15. It is clear from the figure that a first order model ($n = 1$) is a good choice. The estimated model parameters are $A = 0.9669$, $B = 160.6463$, $C = 1.8302$, $D = 0.4759$, $E = -719.6302$ and $\Delta = 2.9844$ and the model

$$x_{k+1} = 0.9669 x_k + 160.6463 du_k + 1.8302 e_k \tag{3.119}$$

$$\hat{dy}_k = 0.4759 x_k - 719.6302 du_k + e_k \tag{3.120}$$

where e_k is a zero mean, Gaussian distributed stochastic sequence with covariance

$$\Delta = \mathrm{E}(e_k e_k^T) = 2.9844 \tag{3.121}$$

and unknown realization.

First, the deterministic part of the estimated model was simulated with all 504 exchange rate observations as inputs. The simulated deterministic model output (dy_k^d) and the actual observations (dy_k) are illustrated in Figure 3.16. The error between the actual and simulated time series can also be measured as (the covariance estimate)

$$\begin{aligned} J_1 &= \frac{1}{200}\sum_{i=1}^{200}(dy_k - dy_k^d)^2 = 23.996 && (3.122) \\ J_2 &= \frac{1}{504}\sum_{i=1}^{504}(dy_k - dy_k^d)^2 = 38.88 && (3.123) \end{aligned}$$

The total combined deterministic and stochastic model was simulated. The stochastic input (innovations process) was generated with the MATLAB function $e = randn(501, 1) * sqrt(\Delta)$ also with seed zero. Note that a different noise realization (different seed) will give a slightly different result. However, the results are illustrated in Figure 3.17.

The innovations model can be written as an optimal prediction model for the output (the crude oil prize). Define the innovations process as $e_k = y_k - \bar{y}_k$ where $\bar{y}_k$ is the optimal prediction of the output y_k. Then we have

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + C(y_k - Dx_k - Eu_k) && (3.124) \\ \bar{y}_k &= Dx_k + Eu_k && (3.125) \end{aligned}$$

The optimal prediction of the crude oil prize variations is illustrated in Figure 3.18.

Even if the model is generated only by the 200 first observations it is capable of roughly predicting the slowly variations in all the 504 crude oil prize observations.

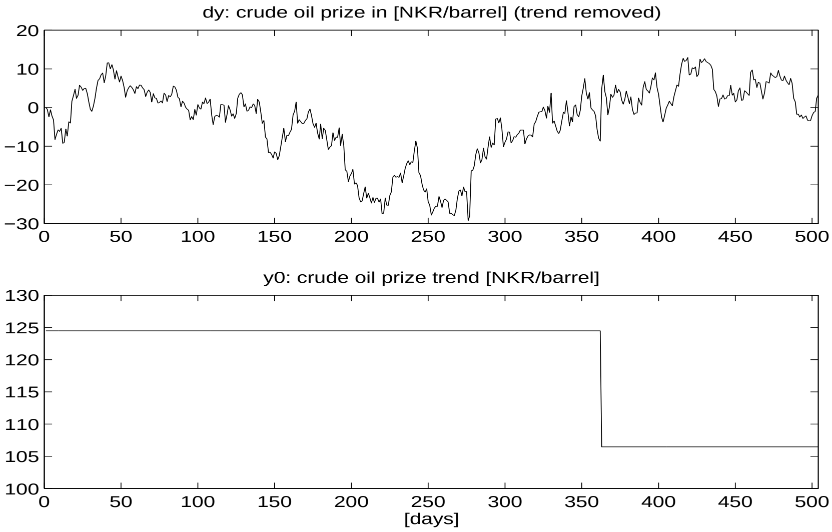


Figure 3.13: The crude oil prize $y_k = dy_k + y_0(k)$ $[\frac{NKR}{barrel}]$ in the period 1993 to 1995 is separated into crude oil prize variations dy_k (upper figure) and trend $y_0(k)$ (lower figure).

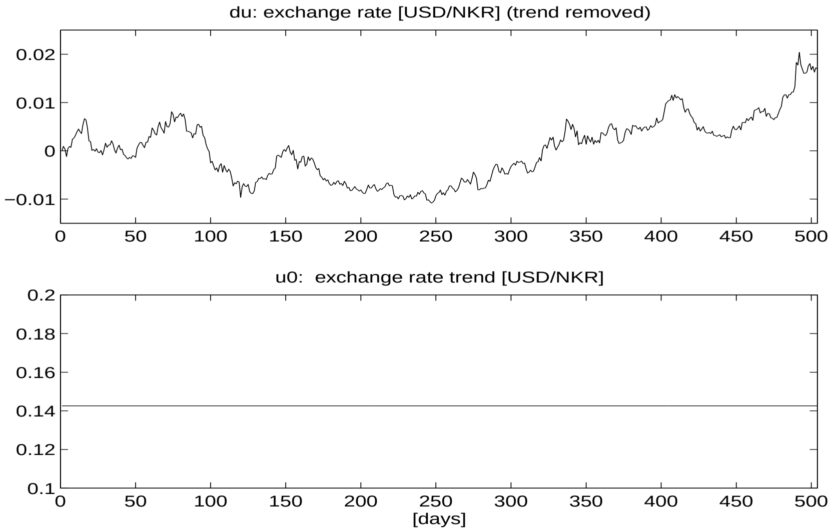


Figure 3.14: The exchange rate $u_k = du_k + u_0(k)$ $[\frac{USD}{NKR}]$ in the period 1993 to 1995 is separated into rate variations du_k (upper figure) and trend $u_0(k)$ (lower figure).

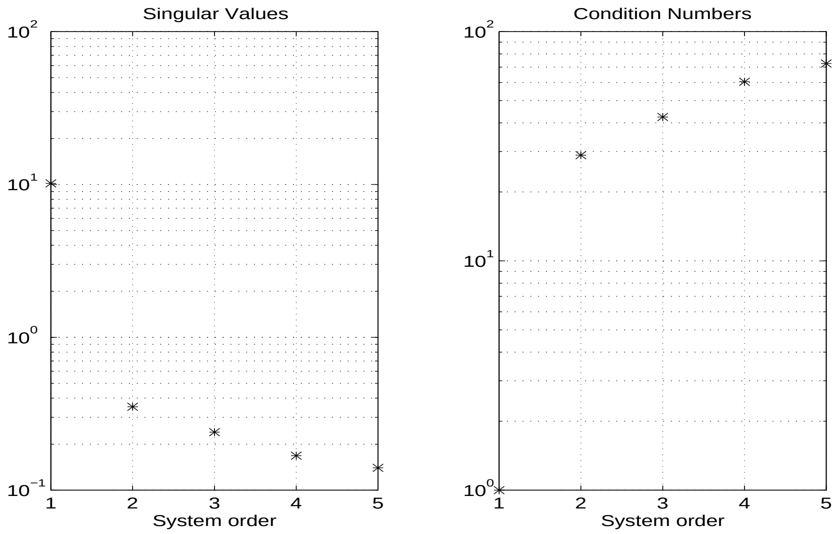


Figure 3.15: The singular values ($s_i,\ i = 1, ..., 5$) and condition numbers ($\frac{s_1}{s_i},\ i = 1, ..., 5$) used to investigate the order of the state space model in Example 4. The model order is estimated as the number of large singular values or the number of small condition numbers.

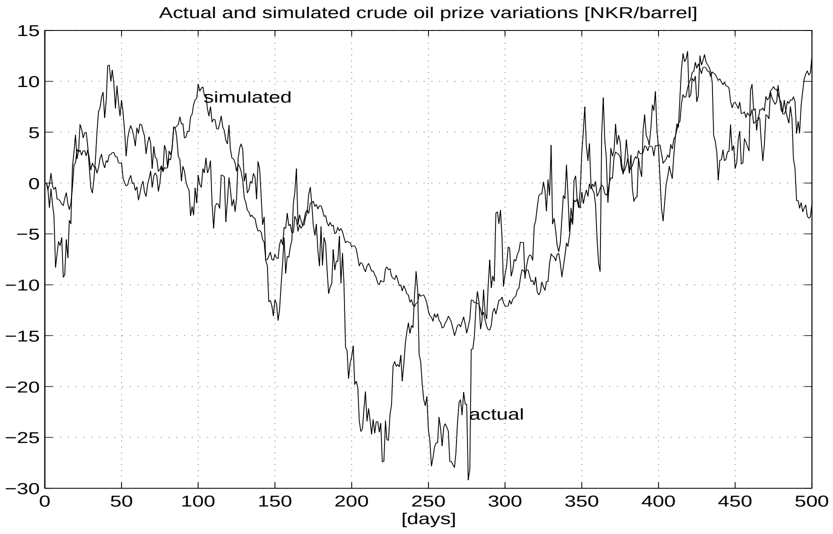


Figure 3.16: Actual and simulated (deterministic part of model) crude oil prize.

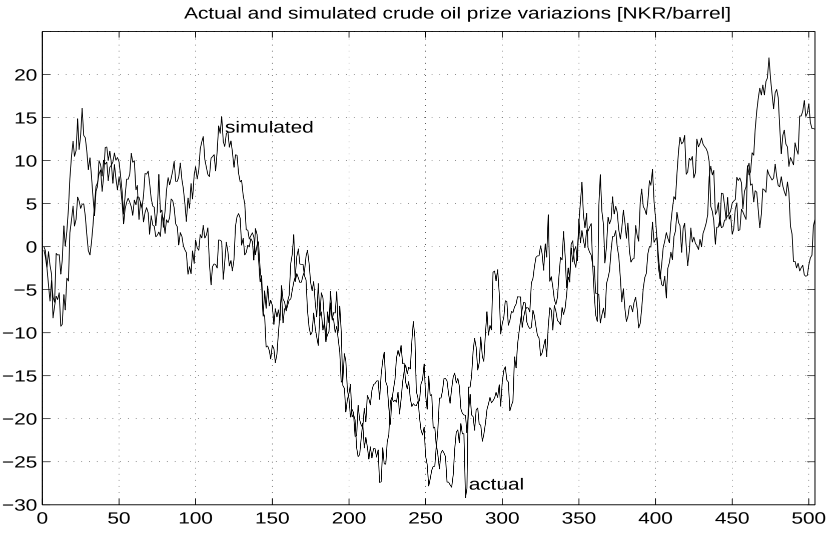


Figure 3.17: Actual and simulated (combined deterministic and stochastic model) crude oil prize. This model was generated from the 200 first samples of the input output samples, only. The algorithm parameter was $L = 5$.

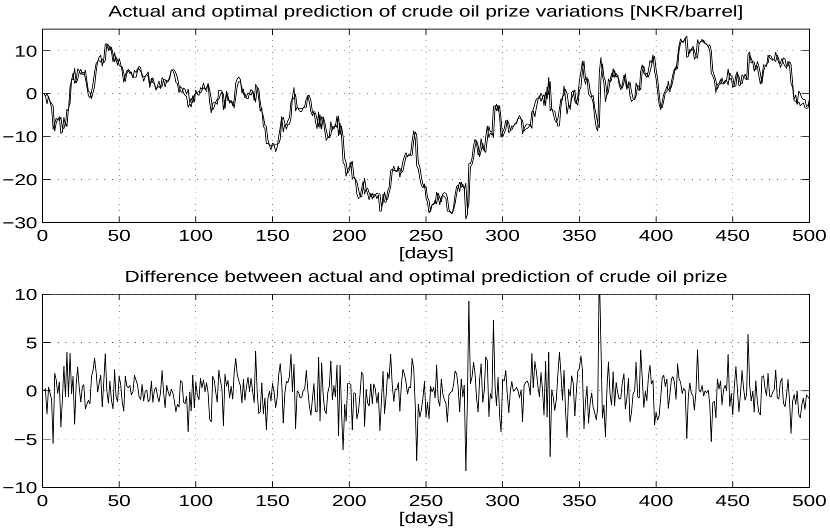


Figure 3.18: Actual and simulated optimal prediction of the crude oil prize is shown in the upper part of the figure. The difference between the actual and the optimal prediction of the crude oil prize is shown in the lower part of the figure. This model was generated from the 200 first samples of the input output samples, only. The algorithm parameter was $L = 5$.

3.7.5 Example 5: Modeling PH

The time series in this example is from a tray (plate) column scrubber at Union Co., Skien, Norway (printed with permission). Liquid travels down (vertical) the column over a series of trays. Gas travels up through contacting devices in each tray. There is a PH control loop at each tray. A suspension of slurry is used as input variable (here defined as u_k) to control the PH (output variable here defined as y_k). The time series shown in Figure 3.19 are from an open loop experiment at one tray in the column.

The sampling time was $\Delta t = 4$ $[sec]$. All the $N = 901$ samples was used for identification. The algorithm parameter was chosen as $L = 2$. Note that $L = 1$, $L = 2$ and $L = 3$ gives about the same model and that a first order model was detected from inspection of the L singular values (when $L \geq 2$).

The estimated model is

$$\begin{aligned} x_{k+1} &= 0.9881 x_k + 5.09 \cdot 10^{-4} u_k + 0.4358 e_k && (3.126) \\ \hat{y}_k &= 0.7113 x_k + 4.76 \cdot 10^{-5} u_k + e_k && (3.127) \end{aligned}$$

where e_k is a zero mean, Gaussian distributed stochastic sequence with covariance

$$\Delta = \mathrm{E}(e_k e_k^T) = 5.26 \cdot 10^{-4} \tag{3.128}$$

The estimated deterministic gain (gain from u_k to $\hat{y}_k$) is 0.031 and the estimated time constant is $T = -\frac{\Delta t}{ln(a)} = 5.6$ $[min]$. Inspection of the actual time series shows that these parameters are reasonable.

Singular values and principal angles for model order selection are shown in Figure 3.20. A simulation of the deterministic part of the model and the measured PH is shown in Figure 3.21.

3.8 Conclusions

A method for subspace identification and realization of state space models on innovations form directly from given input and output data is presented.

The method determines both the deterministic part and the stochastic part of the model.

The algorithm gives exact results in the deterministic case and consistent results when the system is influenced by noise.

The stochastic part of the model is computed from standard linear algebra decomposition methods and no matrix equations (e.g. Riccati or Lyapunov equations) need to be solved.

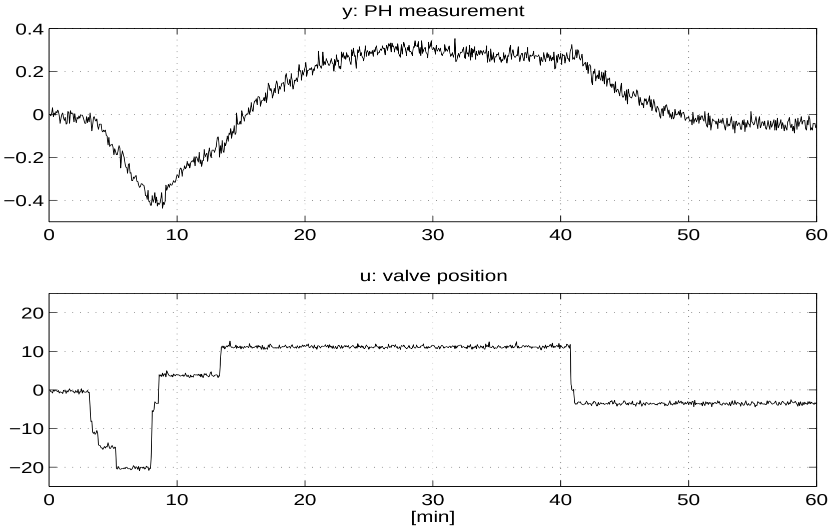


Figure 3.19: The measured PH (output from the process) upper figure. The manipulated input to the process (valve position) is shown in the lower figure.

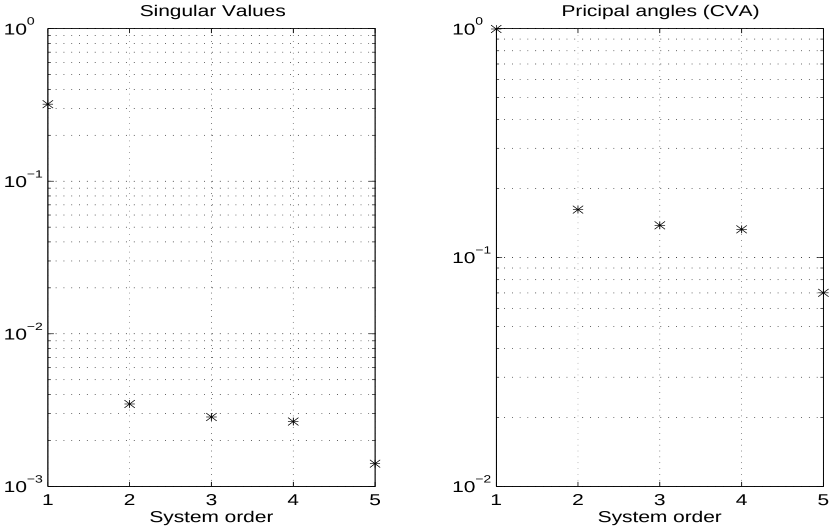


Figure 3.20: Singular values and principal angles for inspection of model order. The principal angles are computed by the **CVA** algorithm. The **DSR** parameter $L = 5$ was chosen. The figure shows that a reasonable model order is $n = 1$. See Example 5 for details.

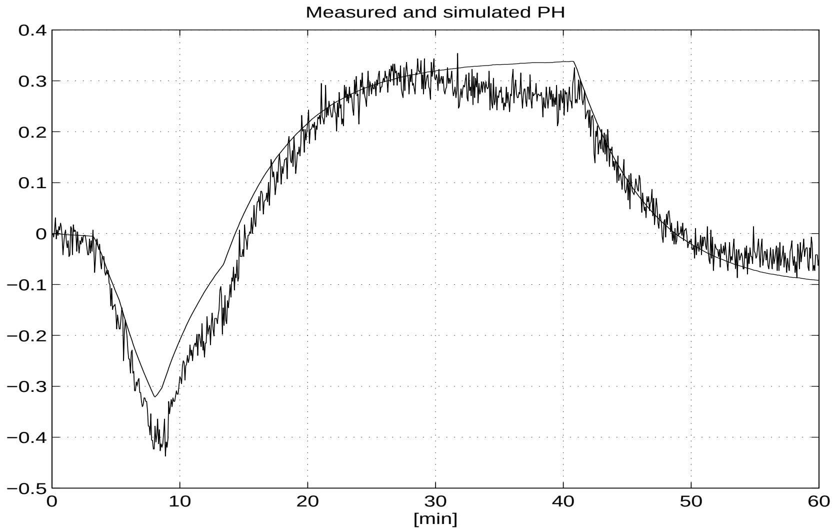


Figure 3.21: Measured and simulated (deterministic model simulation) PH. The model was generated from all the $N = 901$ samples. The algorithm parameter was $L = 2$.

References

Aoki, M. (1990). *State Space Modeling of Time Series.* Second, Revised and Enlarged Edition. Springer-Verlag Berlin, Heidelberg.

Aoki, M. (1994). Two Complementary Representations of Multiple Time Series in State-Space Innovation Forms. *Journal of Forecasting*, Vol. 13, pp. 69-90.

Di Ruscio, D. (1994). Methods for the identification of state space models from input and output measurements. *SYSID 94, The 10th IFAC Symposium on System Identification*, Copenhagen, July 4 - 6.

Di Ruscio, D. (1995). A method for the identification of state space models from input and output measurements. *Modeling, Identification and Control*, Vol. 16, no. 3. Program commercial available by Fantoft Process AS, Box 306, N-1301 Sandvika.

Di Ruscio, D. (1995b). A method for identification of combined deterministic and stochastic systems. *Proceedings of the third European Control Conference, ECC95*, Roma, September 5-8, pp. 429-434.

Di Ruscio, D. and A. Holmberg (1996). Subspace identification for dynamic process analysis and modeling. *Control Systems 96*, Halifax, Nova Scotia, May 1996.

Golub, G. H. and C. F. Van Loan (1983). *Matrix Computations.* North Oxford Academic Publishers Ltd.

Larimore, W. E. (1983). System identification, reduced order filtering and modeling via canonical variate analysis. *Proc. of the American Control Conference*, San Francisco, USA, pp. 445-451.

Larimore, W. E. (1990). Canonical Variate Analysis in Identification, Filtering and Adaptive Control. *Proc. of the 29th Conference on Decision and Control*, Honolulu, Hawaii, December 1990, pp. 596-604.

Ljung, L. (1991). *System Identification Toolbox.* The Mathworks, Inc.

Faurre, P. L. (1976). Stochastic realization algorithms. In: *System Identification: Advances and Case Studies*, (eds. R. K. Mehra and D. G. Lainiotis), Academic Press.

Kalman, R. E., P. L. Falb and M. A. Arbib (1969). *Topics in mathematical system theory.* McGraw-Hill Book Company.

Kung, S. Y. (1978). A new identification and Model Reduction Algorithm via Singular Value Decomposition. *Conf. on Circuits, Systems and Computers*, Pacific Grove, CA, November 1978, pp. 705-714.

Moore, B. C. (1981). Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction. *IEEE Trans. on Automatic Control*, Vol. AC-26, pp. 17-31.

Van Overschee, P. and B. De Moor (1994). N4SID: Subspace Algorithms for the Identification of Combined Deterministic Stochastic Systems. *Automatica*, vol. 30, No. 1, pp.75-94.

Van Overschee, P. (1995). *Subspace Identification: theory-implementation-application.* PhD thesis, Katholieke Universiteit Leuven, Belgium.

Van Overschee, P. and B. De Moor (1995). A Unifying Theorem for Three Subspace System Identification Algorithms. *Automatica*, vol. 31, No. 12, pp. 1853-1864.

Verhagen, M. (1994). Identification of the deterministic part of MIMO state space models given on innovations form from input output data. *Automatica*, vol. 30, No. 1, pp. 61-74.

Viberg, M. (1995). Subspace-Based Methods for the Identification of Linear Time-invariant Systems. *Automatica*, vol. 31, No. 12, pp. 1835-1851.

Corrections

1. Index error in the matrices with zeroes on the right hand side (rhs) of Equation (3.46).
   In the third rhs term, change $0_{Lm\times(k+L+1-J)r}$ to $0_{(L+1)m\times(k+L+1-J)r}$. In the fifth rhs term, change $0_{Lm\times(k+L+1-J)r}$ to $0_{(L+1)m\times(k+L+1-J)m}$. The error is changed.

2. Index error on the right hand side of Equation (3.49). Change $E_{0|L+1}$ to $E_{0|J}$. The error is changed.

3. Index error on the right hand side of Equation (3.51). Change $E_{0|L+1}$ to $E_{0|J}$. The error is changed.

4. Index error on the right hand side of Equation (3.52). Change $E_{0|L+1}$ to $E_{0|J}$. The error is changed.

5. The matrix $U^{\perp}_{L+1|L+1}$ is missing on the right hand side in Equation (3.109). The column weighting matrix W_c in the **N4SID** algorithm should be

   $$W_c = U^{\perp}_{L+1|L+1} W_1^T (W_1 U^{\perp}_{L+1|L+1} W_1^T)^{-1} W_1$$

   The error is changed.

Chapter 4

On the DSR algorithm

4.1 Introduction

The theory of subspace identification (SID) methods will be presented in general. The theory and application of one particular SID method will be presented in some detail.

A SID method can be viewed as a realization based approach to estimating state space models from input and output data. This is a most effective and useful method, in particular for multivariable input and output (combined deterministic and stochastic) systems.

A lower Left Q-orthogonal (LQ) decomposition is often used in subspace identification methods in order to compute certain projection matrices and subspaces of the known data matrices and to estimate the system order and the extended observability matrix of the dynamic system.

The dynamics of the system can be extracted from the column space $R(Z)$ of one particular projected matrix Z which is computed from the input and output data matrices Y, U and a method for computing subspaces, e.g., LQ decomposition, singular value decomposition. An alternative method (to compute the projection matrices, subspaces and the column space $R(Z)$) which is based on the Partial Least Squares (PLS) method (decomposition) is also presented.

Two examples are presented in order to compare different SID methods. First: a Monte Carlo simulation experiment of a MIMO system is presented in order to compare the numerically reliability of one particular subspace method with two other subspace methods presented in the literature.

Second: a real world example from the pulp and paper industry is presented in order to compare the quality of the methods. For this example there are three input variables in the U data matrix and two output variables in the Y data matrix. The data was collected from an experiment design. The quality of the different models and validation aspects are addressed. The estimated state space model is then used in a model predictive control strategy. Simulation results are presented.

4.2 BASIC SYSTEM THEORETIC DESCRIPTION

Underlying system described by a State space model (SSM)

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Gv_k && (4.1) \\ y_k &= Dx_k + Eu_k + w_k && (4.2) \end{aligned}$$

- Process noise v_k, measurement noise w_k are zero-mean and white.
- Colored process noise model included in SSM.
- The system is driven by both known input variables u_k, unknown input variables v_k, w_k and unknown initial values x_0.
- Description of combined deterministic and stochastic linear systems.
- Linearized approximation of non-linear systems.

Innovations form of state space model (ISSM)

is the starting point for the development of the algorithm.

$$\begin{array}{l}\text{Algorithm}\\ \text{developed}\\ \text{from ISSM}\end{array} \left\{ \begin{array}{c} \bar{x}_{k+1} = A\bar{x}_k + Bu_k + Ce_k \\ y_k = D\bar{x}_k + Eu_k + Fe_k \\ \mathbf{E}(e_k e_k^T) = I, \;\; \bar{x}_0 = \mathrm{E}(x_0) \end{array} \right. \qquad (4.3)$$

Traditional innovations form of SSM (Kalman filter)

$$\begin{aligned} \bar{x}_{k+1} &= A\bar{x}_k + Bu_k + K\epsilon_k && (4.4) \\ y_k &= D\bar{x}_k + Eu_k + \epsilon_k && (4.5) \end{aligned}$$

where

$$\begin{array}{ll} \epsilon_k = Fe_k & \text{Innovations process} \\ K = CF^{-1} & \text{Kalman filter gain} \\ \mathbf{E}(\epsilon_k \epsilon_k^T) = FF^T & \text{Innovations covariance matrix} \end{array} \qquad (4.6)$$

Why use ISSM (4.3) ?

- Does not have to invert matrix F.
- Innovations e_k has unit covariance matrix.
- ISSM a more unique representation of the system (4.1) and (4.2).

4.3 PROBLEM DESCRIPTION

From known system input and output data

$$\left.\begin{array}{lll} u_k & \forall & k = 1, 2, ..., N \\ y_k & \forall & k = 1, 2, ..., N \end{array}\right\} \quad \textbf{Known data}$$

find the system order n, the initial state vector x_0 and matrices in the ISSM (up to within a similarity transformation)

$$\begin{array}{l} A \; B \; C \\ D \; E \; F \; x_0 \end{array} \qquad \left\{\begin{array}{l} \textbf{find system order and} \\ \textbf{matrices from known data} \end{array}\right.$$

4.4 BASIC DEFINITIONS

Fundamental subspaces of a matrix $Z \in \mathbb{R}^{m \times K}$

- $R(Z)$:
  - The column space of Z.
  - I.e., the number of independent columns of Z.
  - $R(Z)$ is often called the range of Z.
  - $\dim[R(Z)] = n$
- $N(Z)$:
  - The null space of Z.
  - $\dim[N(Z)] = K - n$

$$\dim[R(Z)] + \dim[N(Z)] = K \quad \text{Number of columns}$$

$$\operatorname{rank}(Z) = \dim[R(Z)] = n$$

In subspace identification:

- the observability matrix is estimated from the column space of a matrix Z.
- the system order is identified as the dimension of the column space of Z.

4.5 ORTHOGONAL PROJECTIONS

- A matrix Y can be decomposed into two matrices with orthogonal row spaces.

$$Y = Y/P + YP^{\perp}$$

- Projection of the row space of Y onto the row space of P.

$$Y/P = YP^{T}(PP^{T})^{\dagger}P \tag{4.7}$$

- Projection of the row space of Y onto the orthogonal complement of the row space of P.

$$YP^{\perp} = Y - YP^{T}(PP^{T})^{\dagger}P \tag{4.8}$$

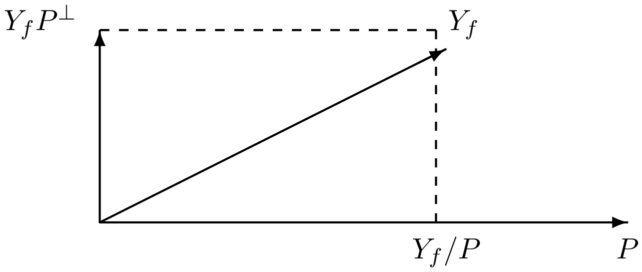


Figure 4.1: Two dimensional illustration of orthogonal projections.

Some useful results

Lemma 4.5.1 *The following equality is true*

$$U/\begin{bmatrix} U \\ W \end{bmatrix} = U \tag{4.9}$$

Lemma 4.5.2 *The following equality is true*

$$U\begin{bmatrix} U \\ W \end{bmatrix}^{\perp} = 0 \tag{4.10}$$

4.6 MATRIX DEFINITIONS

- The *extended observability* matrix for (D, A)

$$O_i \overset{\text{def}}{=} \begin{bmatrix} D \\ DA \\ \vdots \\ DA^{i-1} \end{bmatrix} \qquad \in \; \mathbb{R}^{im \times n} \tag{4.11}$$

Subscript i denotes the number of block rows.

- The *reversed extended controllability* matrix for (A, B)

$$\mathcal{C}_i^d \overset{\text{def}}{=} \begin{bmatrix} A^{i-1}B & A^{i-2}B & \cdots & B \end{bmatrix} \qquad \in \; \mathbb{R}^{n \times ir} \tag{4.12}$$

Subscript i denotes the number of block columns.

- The *reversed extended controllability* matrix for (A, C)

$$\mathcal{C}_i^s \overset{\text{def}}{=} \begin{bmatrix} A^{i-1}C & A^{i-2}C & \cdots & C \end{bmatrix} \qquad \in \; \mathbb{R}^{n \times ir} \tag{4.13}$$

Subscript i denotes the number of block columns.

- The *lower block triangular Toeplitz* matrix for (D, A, B, E)

$$H_i^d \overset{\text{def}}{=} \begin{bmatrix} E & 0 & 0 & \cdots & 0 \\ DB & E & 0 & \cdots & 0 \\ DAB & DB & E & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}B & DA^{i-3}B & DA^{i-4}B & \cdots & E \end{bmatrix} \quad \in \; \mathbb{R}^{im \times ir} \tag{4.14}$$

where the subscript i denotes the number of block rows.

- A *lower block triangular Toeplitz* matrix for (D, A, C, F)

$$H_i^s \overset{\text{def}}{=} \begin{bmatrix} F & 0 & 0 & \cdots & 0 \\ DC & F & 0 & \cdots & 0 \\ DAC & DC & E & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}C & DA^{i-3}C & DA^{i-4}C & \cdots & F \end{bmatrix} \quad \in \; \mathbb{R}^{im \times im} \tag{4.15}$$

4.7 BASIC MATRIX EQUATION IN SUBSPACE IDENTIFICATION

$$Y_{k|L} = O_L X_k + H_L^d U_{k|L} + H_L^s E_{k|L} \tag{4.16}$$

where

$$Y_{k|L} = \overbrace{\begin{bmatrix} y_k & y_{k+1} & y_{k+2} & \cdots & y_{k+K-1} \\ y_{k+1} & y_{k+2} & y_{k+3} & \cdots & y_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{k+L-1} & y_{k+L} & y_{k+L+1} & \cdots & y_{k+L+K-2} \end{bmatrix}}^{\textbf{Known} \text{ data matrix of output variables}} \in \mathbb{R}^{Lm \times K} \tag{4.17}$$

$$U_{k|L} = \overbrace{\begin{bmatrix} u_k & u_{k+1} & u_{k+2} & \cdots & u_{k+K-1} \\ u_{k+1} & u_{k+2} & u_{k+3} & \cdots & u_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{k+L-1} & u_{k+L} & u_{k+L+1} & \cdots & u_{k+L+K-2} \end{bmatrix}}^{\textbf{Known} \text{ data matrix of input variables}} \in \mathbb{R}^{Lr \times K} \tag{4.18}$$

$$E_{k|L} = \overbrace{\begin{bmatrix} e_k & e_{k+1} & e_{k+2} & \cdots & e_{k+K-1} \\ e_{k+1} & e_{k+2} & e_{k+3} & \cdots & e_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_{k+L-1} & e_{k+L} & e_{k+L+1} & \cdots & e_{k+L+K-2} \end{bmatrix}}^{\textbf{Unknown} \text{ matrix of noise variables}} \in \mathbb{R}^{Lm \times K} \tag{4.19}$$

$$X_k = \overbrace{\begin{bmatrix} x_k & x_{k+1} & x_{k+2} & \cdots & x_{k+K-1} \end{bmatrix}}^{\textbf{Unknown} \text{ matrix of state vectors}} \in \mathbb{R}^{n \times K} \tag{4.20}$$

4.8 MATRIX EQUATION WITH STATES ELIMINATED

Define extended state space model (ESSM)

$$Y_{k+1|L} = \tilde{A}_L Y_{k|L} + \tilde{B}_L U_{k|L+1} + \tilde{C}_L E_{k|L+1} \tag{4.21}$$

where

$$\tilde{A}_L = O_L A (O_L^T O_L)^{-1} O_L^T \tag{4.22}$$

$$\tilde{B}_L = \begin{bmatrix} O_L B & H_L^d \end{bmatrix} - \tilde{A}_L \begin{bmatrix} H_L^d & 0_{Lm \times r} \end{bmatrix} \tag{4.23}$$

$$\tilde{C}_L = \begin{bmatrix} O_L C & H_L^s \end{bmatrix} - \tilde{A}_L \begin{bmatrix} H_L^s & 0_{Lm \times m} \end{bmatrix} \tag{4.24}$$

H_L^d and H_L^s are the lower left block triangular Toepliz matrices.

NOTE

Known	$\tilde{B}_L$	→	E given as lower right sub-matrix in $\tilde{B}_L$
Known	$\tilde{B}_L$, O_L and $\tilde{A}_L$	→	B given

4.9 MATRIX OF INSTRUMENTAL VARIABLES W

Need instrument to “remove” noise term in e.g., the ESSM.

$$\begin{aligned} Y_{J+1|L} &= \tilde{A}_L Y_{J|L} + \tilde{B}_L U_{J|L+1} + \tilde{C}_L \overset{\substack{\textbf{unknown} \\ \downarrow}}{E_{J|L+1}} \\ Y_{J|L} &= O_L X_J + H_L^d U_{J|L} + H_L^s \underset{\substack{\uparrow \\ \text{need instrument to} \\ \text{remove this term}}}{E_{J|L}} \end{aligned}$$

1. Instrument should be “uncorrelated” with the noise matrix $E_{J|L+1}$,

$$\lim_{K\to\infty} E_{J|L+1}/W = 0, \qquad \lim_{K\to\infty} E_{J|L+1}W^{\perp} = E_{J|L+1} \tag{4.25}$$

2. Instrument should be sufficiently correlated with informative part of ESSM in order not to destroy information, e.g about system order.

$$\text{rank}(X_J/W) = n$$

Define the matrix of instrumental variables as

$$W = \begin{bmatrix} Y_{0|J} \\ U_{0|J} \end{bmatrix} \qquad \left.\vphantom{\begin{bmatrix} Y_{0|J} \\ U_{0|J} \end{bmatrix}}\right\} \begin{array}{l} \text{instrument with } J \text{ “}\textbf{past}\text{”} \\ \text{inputs and outputs} \end{array} \tag{4.26}$$

$$\left.\begin{array}{l} J \geq L \\ Y_{J+1|L} = \tilde{A}_L Y_{J|L} + \tilde{B}_L U_{J|L+1} + \tilde{C}_L E_{J|L+1} \end{array}\right\} \begin{array}{l} \text{ESSM with “}\textbf{future}\text{”} \\ \text{inputs and outputs} \end{array} \tag{4.27}$$

Interpretation of the horizons

- J: Past horizon for instruments.
- L: Future horizon used to predict the number of states.

How to chose parameters J and L

- Choose $J = L$. Seems no reasons for choosing $J > L$.
- Min. number of block rows for observability are $L_{\text{min}} = n - m + 1$. Chose $L \geq L_{\text{min}}$.
- System order $n \leq \min(Lm, J(m + r))$.

4.10 Subspace identification of O_L: autonomous systems

Consider the matrix equation

$$Z_{k|L} = O_L X_k \quad \in \ \mathbb{R}^{Lm \times K} \tag{4.28}$$

Assumptions

- Data matrix $Z_{k|L}$ known.
- The pair (D, A) observable.
- $\text{rank}(X_k) = n$.

Column space of product

- The column space of $O_L X_k$ is contained in the column space of O_L.

$$R(O_L X_k) \subseteq R(O_L) \tag{4.29}$$

Sylvesters inequality gives

$$\text{rank}(O_L) + \text{rank}(X_k) - n \leq \text{rank}(Z_{k|L}) \leq \min\{\text{rank}(O_L), \text{rank}(X_k)\} \tag{4.30}$$

Conclusions

- The column space of $Z_{k|L}$ coincide with the column space of O_L.

$$R(Z_{k|L}) = R(O_L) \tag{4.31}$$

- The rank of $Z_{k|L}$ (which is equal to the dimension of the column space) is equal to the number of states n.

$$\text{rank}(Z_{k|L}) = \dim[R(Z_{k|L})] = n \tag{4.32}$$

- Can estimate both the system order n and the extended observability matrix O_L from the matrix $Z_{k|L}$.

4.11 Subspace identification of O_L: deterministic systems

Deterministic SSM can be written as the matrix equation

$$Y_{k|L} = O_L X_k + H_L^d U_{k|L} \tag{4.33}$$

- Data-matrices $Y_{k|L}$ and $U_{k|L}$ are known.

Results

$$Z_{k|L} \overset{\text{def}}{=} Y_{k|L} U_{k|L}^{\perp} = O_L X_k U_{k|L}^{\perp} \tag{4.34}$$

and

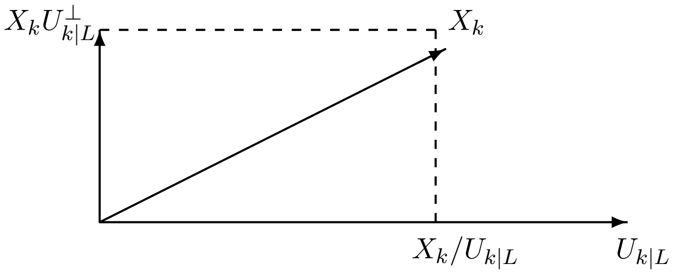
$$\text{rank}(Z_{k|L}) = n \quad \text{if} \quad \text{rank}(X_k U_{k|L}^{\perp}) = n \tag{4.35}$$

- $X_k U_{k|L}^{\perp}$ loose rank below n ?

Analysis

A matrix X_k can be decomposed into two matrices with orthogonal row spaces.

$$X_k = X_k / U_{k|L} + X_k U_{k|L}^{\perp}$$



- $X_k/U_{k|L}$: Projection of the row space of X_k on the row space of $U_{k|L}$.
- $X_k U_{k|L}^{\perp}$: Projection of the row space of X_k onto the orthogonal complement of the row space of $U_{k|L}$.

Conclusion

$$\text{rank}(X_k U_{k|L}^{\perp}) = n \quad \text{if} \quad X_k \neq U_{k|L} \tag{4.36}$$

4.12 Subspace identification of O_{L+1}: “general” case

Theorem 4.12.1 (Realization of the extended observability matrix O_{L+1})

Define from the known data

$$Z_{J|L+1} \overset{\text{def}}{=} (Y_{J|L+1}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix})U^{\perp}_{J|L+1}, \tag{4.37}$$

$Z_{J|L+1}$ *is related to the extended observability matrix as*

$$Z_{J|L+1} = O_{L+1}X^a_J \tag{4.38}$$

where

$$X^a_J = X_J/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} U^{\perp}_{J|L+1} = \begin{bmatrix} A^J & C^d_J & C^s_J \end{bmatrix} \begin{bmatrix} X_0/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \\ U_{0|J} \\ E_{0|J}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} U^{\perp}_{J|L+1}$$

- *The column space of $Z_{J|L+1}$ coincides with column space of O_{L+1}.*
- *The system order n is given as the dimension of the column space.*

4.13 BASIC PROJECTIONS IN THE DSR ALGORITHM

"Autonomous system" for system dynamics, A and D

$$Z_{J|L+1} = O_{L+1} X_J^a \tag{4.39}$$

$$Z_{J+1|L} = \tilde{A}_L Z_{J|L} \tag{4.40}$$

where

$$Z_{J|L+1} \overset{\text{def}}{=} (Y_{J|L+1} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) U_{J|L+1}^{\perp} \tag{4.41}$$

Hence,

$$(A, D) = \arg\min_{A,D} \parallel Z_{J+1|L} - \tilde{A}_L Z_{J|L} \parallel_F^2 \tag{4.42}$$

"Deterministic system" for B and E

$$Z_{J+1|L}^d = \tilde{A}_L Z_{J|L}^d + \tilde{B}_L U_{J|L+1} \tag{4.43}$$

where

$$Z_{J|L+1}^d \overset{\text{def}}{=} Y_{J|L+1} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{4.44}$$

Hence,

$$(B, E) = \arg\min_{B,E} \parallel Z_{J+1|L}^d - \tilde{A}_L Z_{J|L}^d - \tilde{B}_L U_{J|L+1} \parallel_F^2 \tag{4.45}$$

"Stochastic system" for C and F

$$Z_{J+1|L}^s = \tilde{A}_L Z_{J|L}^s + \tilde{C}_L E_{J|L+1} \tag{4.46}$$

where

$$Z_{J|L+1}^s \overset{\text{def}}{=} Y_{J|L+1} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} \tag{4.47}$$

Hence,

$$(C, F) = \arg\min_{C,F} \parallel Z_{J+1|L}^s - \tilde{A}_L Z_{J|L}^s - \tilde{C}_L E_{J|L+1} \parallel_F^2 \tag{4.48}$$

4.14 COMPUTING PROJECTIONS BY QR DECOMPOSITION

DATA COMPRESSION BY QR-DECOMPOSITION

Compute the QR decomposition (or lower **L**eft triangular, Q-orthogonal (LQ))

$$\frac{1}{\sqrt{K}}\tilde{Y} \overset{\text{def}}{=} \frac{1}{\sqrt{K}} \overbrace{\begin{bmatrix} U_{J|L+1} \\ W \\ Y_{J|L} \\ Y_{J+1|L} \end{bmatrix}}^{\overset{\text{“large”}}{K}} = \underbrace{\overbrace{\begin{bmatrix} R_{11} & 0 & 0 & 0 \\ R_{21} & R_{22} & 0 & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix}}^{\overset{\text{“small and square”}}{(2L+1)(r+m)}}}_{\text{needed in algorithm}} \underbrace{\overbrace{\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix}}^{\overset{\text{“large”}}{K}}}_{\text{not needed}} = RQ \tag{4.49}$$

POINTS

- Compress large and known data matrices into (usually) a “smaller” matrix **R**.
- All relevant system information in lower left triangular part of matrix **R**. Q is orthogonal.
- **R** is the “square root” of the covariance matrix $\frac{1}{K}\tilde{Y}\tilde{Y}^T$.
- **R** is computed without never computing the covariance matrix $\frac{1}{K}\tilde{Y}\tilde{Y}^T$.
- QR decomposition make algorithm numerically robust, e.g. due to rounding off errors (usually in situations when N is large), and ill-conditioned noise realizations.

SEPARATION INTO

- AUTONOMOUS SYSTEM
- DETERMINISTIC SYSTEM AND
- STOCHASTIC SYSTEM

I.e., “data” separated into three parts by the QR-decomposition

1. One part for analyzing and determination of system dynamics, i.e. the system order n and state equation system matrix A

$$R_{42} = \tilde{A}_L R_{32} \tag{4.50}$$

2. One part for determination of the **deterministic part of the system**

$$R_{41} - \tilde{A}_L R_{42} = \tilde{B}_L R_{11} \tag{4.51}$$

3. One part for determination of the **stochastic part of the system**

$$R_{44} = \tilde{C}_L E_L Q_4^T \tag{4.52}$$

$$R_{43} - \tilde{A} R_{33} = \tilde{C}_L E_L Q_3^T \tag{4.53}$$

EXTRACTING THE SYSTEM MATRICES (FROM R)

SOLUTION:

$$\begin{array}{lcl} \text{From sub-matrices in } R & & \text{Extract order } n \text{ and matrices in the ISSM} \\ R & \to & \left\{ \begin{array}{l} A \; B \; C \\ D \; E \; F \end{array} \right. \end{array}$$

1. System order: n and matrix D

$$R_{32} = \overbrace{\left[\begin{array}{cc} U_1 & U_2 \end{array} \right] \left[\begin{array}{cc} S_1 & 0 \\ 0 & S_2 \end{array} \right] \left[\begin{array}{c} V_1^T \\ V_2^T \end{array} \right]}^{\text{SVD}} \left. \begin{array}{c} \\ \\ \\ \\ \end{array} \right\} \begin{array}{ll} n & \text{System order by inspection} \\ & \text{of “non zero” singular values.} \\ O_L = U_1 & \text{Extended observability matrix.} \\ D = O_L(1:m,:) & \text{Matrix in ISSM output eq.} \end{array}$$

2. Matrix A

$$R_{42} = \tilde{A}_L R_{32} \qquad \left. \begin{array}{c} \\ \\ \end{array} \right\} \begin{array}{ll} A = U_1^T R_{42} V_1 S_1^{-1} & \text{System matrix in ISSM state eq.} \\ \tilde{A}_L = OA(O^T O)^{-1} O^T & \text{System matrix in ESSM.} \end{array}$$

3. Matrices B and E

$$R_{41} - \tilde{A}_L R_{31} = \tilde{B}_L R_{11} \to \text{cs}(\left[\begin{array}{c} B \\ E \end{array} \right]) = N^\dagger \text{cs}(R_{41} - \tilde{A}_L R_{31})$$

4. Matrices C and F

$$\begin{array}{lcl} R_{44} & \to & F \quad \text{and } \text{E}(\epsilon_k \epsilon_k^T) = FF^T \\ R_{43} - \tilde{A}_L R_{33} = \tilde{C}_L E_L Q_3^T & \to & C \end{array}$$

where $E_L Q_3^T$ block diagonal with F on the diagonal, see paper eq. (66).

4.15 COMPUTING PROJECTIONS WITH PLS

- PLS: Partial Least Squares, Wold (1966).
- Projections in DSR algorithm can be computed by the use of PLS.
- Some effect for purely exiting input signals.
- Regularization. Bias and variance.

4.16 MODEL ORDER SELECTION BY EXAMPLE

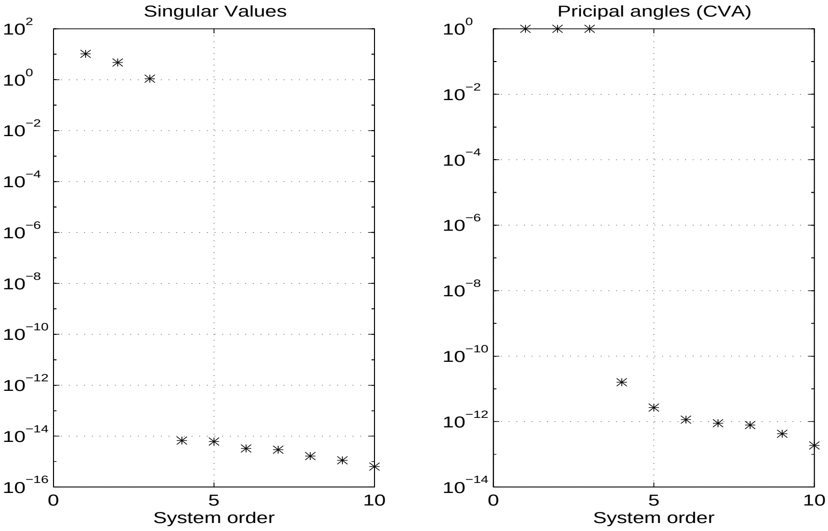


Figure 4.2: Model order selection for process with two inputs and two outputs and three states (no noise case).

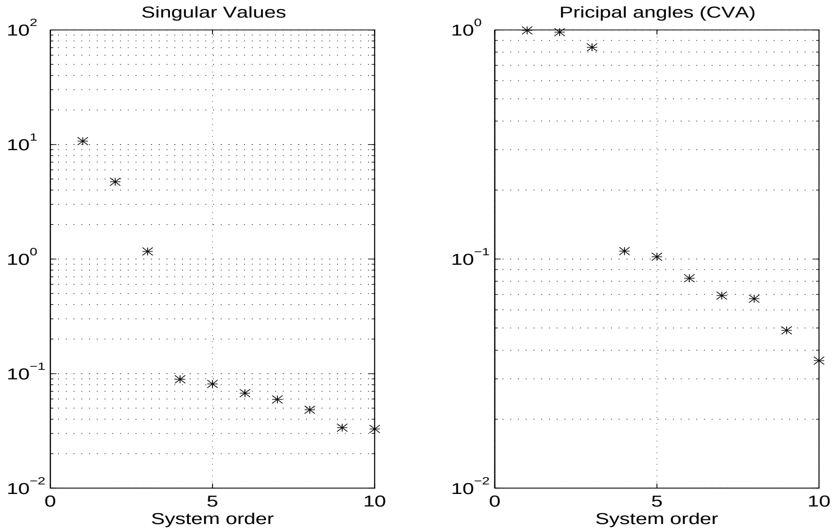


Figure 4.3: Model order selection for process with two inputs and two outputs and three states (noisy case).

4.17 PROPERTIES OF THE ALGORITHM

- Numerically robust implementation. The complete model is determined from QR and SVD only.
- Practical and easy to use implementation.
- Only **one** scalar parameter to be specified by the user.
  $\triangleright$ the number of block rows in the extended observability matrix.
  Choose
  $$L \geq 1 \tag{4.54}$$
- Can choose system order
  $$n \leq \min(Lm, J(m+r)) \tag{4.55}$$
- QR-decomposition of a data matrix with row size only
  $$(2L+1)(r+m) \tag{4.56}$$
- Do not solve any Riccati equation in order to determine
  - C - the Kalman filter gain matrix
  - Δ - the innovations process noise covariance matrix
- “Exact” results for deterministic systems.
- Consistent results for combined deterministic and stochastic systems (or pure stochastic systems).
- **No** problems with non-zero initial values (of the states).

4.18 COMPARISON WITH CLASSICAL APPROACHES

1. **Parameterizations:**

   - Classical **PE** approaches need a certain user-specified model parameterization, so-called canonical forms.
     There has been an extensive amount of research to determine canonical forms.
     Many problems with canonical forms (minimal parameterizations).
     Can lead to numerically ill-conditioned problems, meaning that the parameters in the canonical form model is extremely sensitive to small perturbations.
   - **DSR** method need no parameterization.

2. **Convergence:**

   - **PE** approaches is iterative. Many hard to deal with problems such as
     problems with lack of convergence; *no convergence*; *slow convergence*;
     local minima; *numerical instability*;
   - **DSR** method is non-iterative, there are no problems with convergence and the method is numerically robust.

3. **Speed:**
   DSR method faster than classical **PE** approaches (because the method is non-iterative).

4.19 INDUSTRIAL APPLICATIONS

A short description is given of applications of the DSR identification method for analysis and modeling of five different industrial data sets.

1. **Thermo Mechanical Pulping (TMP) refiner**. Models successfully identified between almost all input and output variables in the process.
2. **Paper machine.** Dynamic model with 2 input and 2 output variables identified.
3. **Plate column scrubber.** 4 input and 5 output variables basic for modeling. (*student project*)
4. **Raw oil distillation column**. 4 input and 3 output variables are used for identifying a dynamic modeling.
5. **Rotary oven**. 4 input and 2 output variables are used for identifying a dynamic model.

4.20 SOFTWARE

- DSR Toolbox for MATLAB. Fantoft Prosess A.S, P.O. Box 306, 1301 Sandvika, Norway.
- DSR Windows 3.x, 95 or NT program. Fantoft Prosess A.S, P.O. Box 306, 1301 Sandvika, Norway.
- Deterministic and Stochastic system identification and Realization (DSR).

How to use matlab script

Known process input and output data

$$\left.\begin{array}{lll} u_k & \forall & k = 1, 2, ..., N \\ y_k & \forall & k = 1, 2, ..., N \end{array}\right\} \quad \textbf{known}$$

Organize into data matrices

$$\overbrace{Y = \begin{bmatrix} y_0^T \\ y_1^T \\ \vdots \\ y_{N-1}^T \end{bmatrix}}^{\textbf{Known} \text{ data matrix of output variables}} \in \mathbb{R}^{N\times m} \tag{4.57}$$

$$\overbrace{U = \begin{bmatrix} u_0^T \\ u_1^T \\ \vdots \\ u_{N-1}^T \end{bmatrix}}^{\textbf{Known} \text{ data matrix of input variables}} \in \mathbb{R}^{N\times r} \tag{4.58}$$

- Computing the dynamic model

$$\left[\ A,B,D,E,C,F,x_0\ \right] = \mathrm{DSR}(Y, U, L) \tag{4.59}$$

- One scalar parameter L.
- Choose system order $n \leq \min(Lm, J(m + r))$.

4.21 Analysis and modeling of a scrubber

- Time series from a tray (plate) column scrubber, Union Co., Skien, Norway.
- Sampling time 4 [sec]. $N = 901$ samples used for identification.

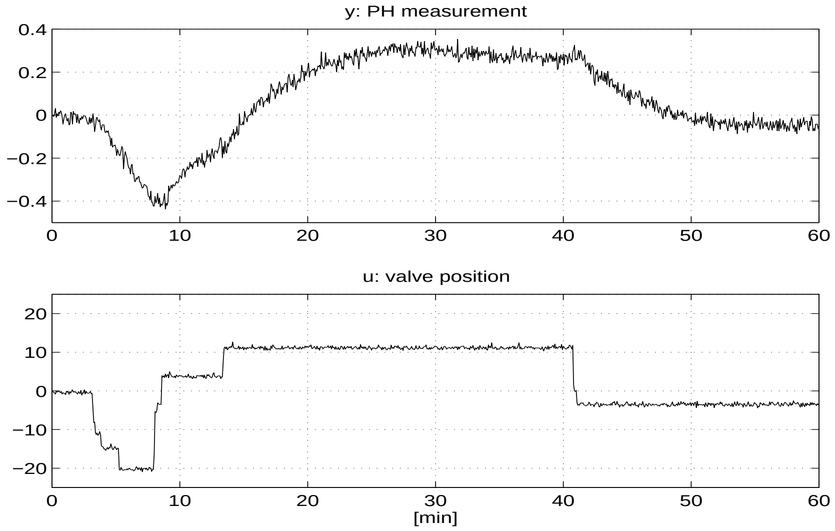


Figure 4.4: The measured PH in upper figure. The input (valve position) in lower figure.

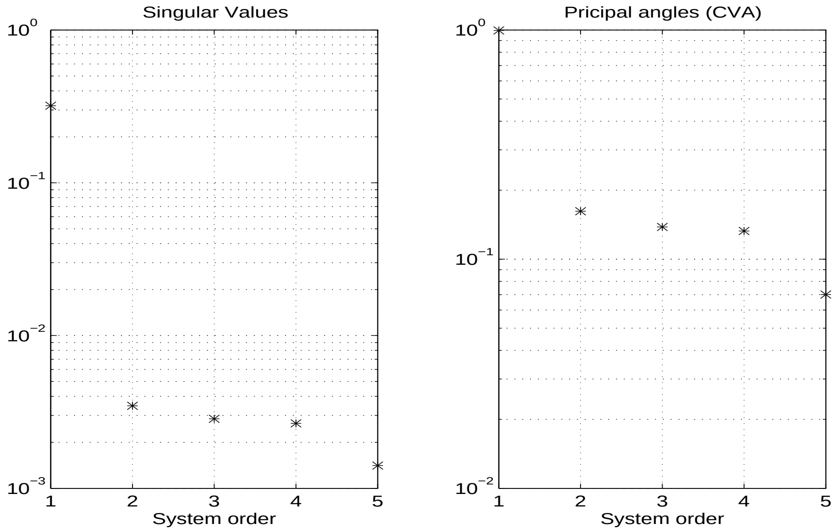


Figure 4.5: Principal directions and principal angles for dynamic order selection.

4.22 MONTE CARLO SIMULATION OF MIMO SYSTEM

System data:

Number of inputs:	$r = 2$
Number of outputs:	$m = 2$
Number of states:	$n = 3$

Input experiment signal:

$$u_t = \left[\begin{array}{l} \sin(\frac{t}{14}) + \sin(\frac{t}{7}) + \sin(\frac{t}{3}) \\ (\sin(\frac{t}{25}) + \sin(\frac{t}{10}) + \sin(\frac{t}{5}) + \sin(t))0.2 \end{array} \right] \Bigg\} \text{Case 1: results in Figure 4.6}$$

$$u_t = \left[\begin{array}{l} (\sin(\frac{t}{14}) + \sin(\frac{t}{7}) + \sin(\frac{t}{3}))0.05 \\ (\sin(\frac{t}{25}) + \sin(\frac{t}{10}) + \sin(\frac{t}{5}) + \sin(t))0.1 \end{array} \right] \Bigg\} \text{Case 2: results in Figure 4.7}$$

Noise covariance matrices:

$$\begin{aligned} \mathrm{E}(w_k) = 0, \quad \mathrm{E}(w_k w_k^T) &= \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \\ \mathrm{E}(v_k) = 0, \quad \mathrm{E}(v_k v_k^T) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned} \tag{4.60}$$

Simulation parameters:

Number of samples:	$N = 5000$
Number of experiments:	$M = 100$

Algorithms and parameters used:

DSR,	$L = 2$
CVA, Larimore (1990),	$I = 3$
ROBUST, Van Overschee (1995),	$I = 3$

- For purely exiting input signals, choose **DSR** parameter L small.
- L is the horizon used to predict the number of states.
- For choice $L = 2$, must choose $n \leq Lm = 4$.
- **CVA** and **ROBUST**: The smallest parameter is $I = 3$ in order to identify a 3rd order model.

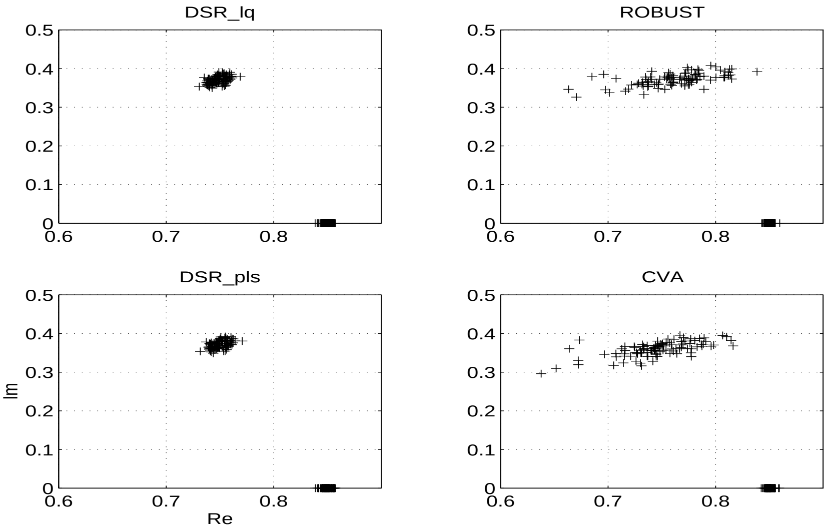


Figure 4.6: Pole estimates of a system with two inputs, two outputs and three states. **DSR** parameters $L = J = 2$ for both the LQ and PLS method for computing the projection matrices. Parameter $I = 3$ for **ROBUST** and **CVA**. The exact poles are $\lambda_{1,2} = 0.75 \pm 0.3708i$ and $\lambda_3 = 0.85$.

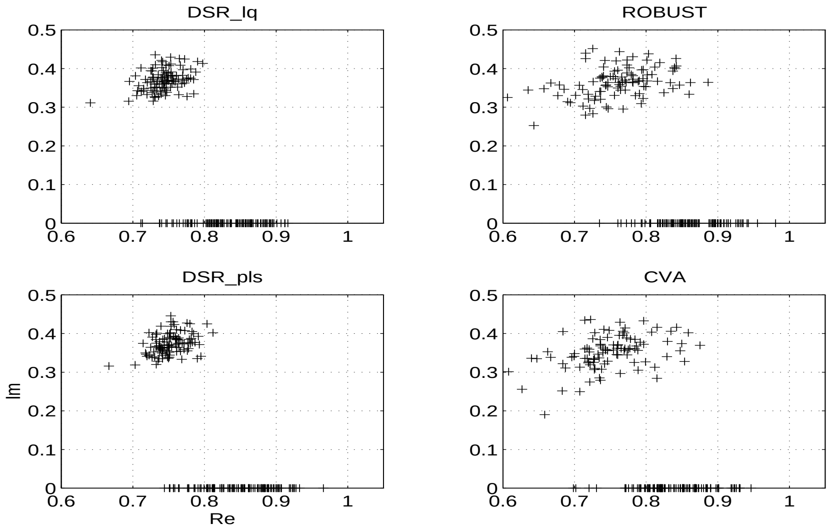


Figure 4.7: Pole estimates of a system with two inputs, two outputs and three states. **DSR** parameters $L = J = 2$ for both the LQ and PLS method for computing the projection matrices. Parameter $I = 3$ for **ROBUST** and **CVA**. The exact poles are $\lambda_{1,2} = 0.75 \pm 0.3708i$ and $\lambda_3 = 0.85$.

4.23 STATISTICAL ANALYSIS

- The method (**DSR**) is consistent.
- Efficiency is analyzed with Monte Carlo simulation

Remarks

- *DSR* is "more" efficient than *n4sid*, for a "small" number of observations.
- PEM did not converge sufficiently when initial-values (to PEM) was estimated with the functions in the matlab *ident* toolbox.
- Refinement of the prediction error approach is possible, but at a higher computational expense.

Conclusions:

- This example shows a larger variability in the complex pole estimates from **CVA** and**ROBUST** than the corresponding pole estimates from **DSR**.
- **CVA** and**ROBUST** estimated the real pole with reasonable accuracy.
- The estimates from **DSR** is close to the estimates from **PEM**.

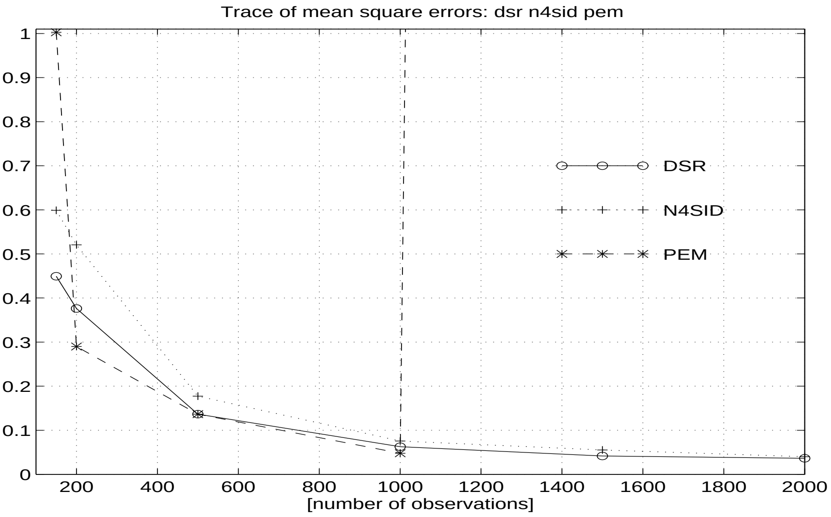


Figure 4.8: The trace of the covariance matrix of the difference between the estimated and the true parameter vector. PEM did not converge sufficiently when the number of samples was small and large.

4.24 IDENTIFICATION AND CONTROL OF TMP PROCESS

- The process is a Thermo Mechanical Pulping (TMP) plant at Union Co, Skien, Norway. A key part in this process is a Sunds Defibrator double (rotating) disk RGP 68 refiner.

- Subspace identification and MPC of a TMP refiner are addressed.

  – Real process data for model identification.

  – MPC algorithm simulation results only.

- The refiner is modeled by a MIMO (3-input, 2-output and 6-state) state space model.

Description of the process variables

Input and output time series from a TMP refiner are presented in Figures 4.9 and 4.10. The time series is the result of a statistical experimental design .

The manipulable input variables

$$\text{Refiner input variables} \begin{cases} u_1 & : \text{ Plug screw speed, } [rpm] \\ u_2 & : \text{ Flow of dilution water, } [\frac{kg}{s}] \\ u_3 & : \text{ Plate gap, } [mm] \end{cases}$$

The following vector of input variables is defined

$$u_k = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}_k \in \mathbb{R}^3. \tag{4.61}$$

The output variables

The process outputs used for identification is defined as follows

$$\text{Refiner output variables} \begin{cases} y_1 & : \text{ Refiner specific energy, } [\frac{MWh}{1000kg}] \\ y_2 & : \text{ Refiner consistency, } [\%] \end{cases}$$

The following vector of process output variables is defined

$$y_k = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_k \in \mathbb{R}^2. \tag{4.62}$$

Subspace identification

- Input and output time series from a TMP refiner are presented in Figures 4.9 and 4.10. The time series is the result of a statistical experimental design .
- The problem is to identify the SSM matrices (A, B, C, D, E, F) (up to within a similarity transformation), including the system order (n), directly from known system input and output data vectors (or time series).
- The SSM is assumed to be on innovations form (Kalman filter).

The known process input and output data vectors from the TMP process can be defined as follows

$$\left.\begin{array}{lll} u_k & \forall & k = 1, \ldots, N \\ y_k & \forall & k = 1, \ldots, N \end{array}\right\} \textbf{Known data}$$

For the TMP refiner example we have used:

$$\begin{array}{rll} N = 1500 & [samples] & \text{for model identification} \\ 860 & [samples] & \text{for model validation} \end{array}$$

- Trends should preferably be removed from the data and the time series should be adjusted for time delays. Data preprocessing is not necessary but it often increase the accuracy of the estimated model.

The following constant trends (working points) are removed from the refiner input and output data.

$$u^0 = \begin{bmatrix} 52.3 \\ 7.0 \\ 0.58 \end{bmatrix}, \quad y^0 = \begin{bmatrix} 1.59 \\ 34.3 \end{bmatrix} \tag{4.63}$$

Organize the process output and input data vectors y_k and u_k as follows

$$\overbrace{Y = \begin{bmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_N^T \end{bmatrix} \in \mathbb{R}^{N \times m}}^{\textbf{Known} \text{ data matrix of output variables}} \tag{4.64}$$

$$\overbrace{U = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_N^T \end{bmatrix} \in \mathbb{R}^{N \times r}}^{\textbf{Known} \text{ data matrix of input variables}} \tag{4.65}$$

The problem of identifying a complete (usually) dynamic model for the process can be illustrated by the following function (similar to a matlab function).

$$\begin{bmatrix} A, & B, & C, & D, & E, & F \end{bmatrix} = \text{DSR}(Y, U, \mathcal{L}) \tag{4.66}$$

where the sixfold matrices (A, B, C, D, E, F) are the state space model matrices.

- The algorithm name **DSR** stands for Deterministic and Stochastic model Realization and identification.
- $\mathcal{L}$ is a positive integer parameter which should be specified by the user.
- The parameter $\mathcal{L}$ defines an upper limit for the unknown system order n. $n \leq \mathcal{L}m$.
- The user must chose the system order by inspection of a plot with Singular Values (SV) or Principal Angles (PA). The system order n is identified as the number of "non-zero" SV's or "non-zero" PA's.
- Note that the Kalman filter gain is given by $K = CF^{-1}$ and that the covariance matrix of the noise innovation process is given by $\mathrm{E}(v_k v_k^T) = FF^T$.
- $\mathcal{L} = 3$ and $n = 6$ where chosen in this example.

Model predictive control of TMP refiner

The following weighting matrices are used in the control objective.

$$Q_i = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}, \quad R_i = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 100 \end{bmatrix}, \; \mathcal{P}_i = 0_r, \; \forall \; i = 1, \cdots, L \tag{4.67}$$

The following horizons are used

$$\text{Horizons in MPC algorithm} \begin{cases} L = 6, & \text{the prediction horizon} \\ J = 6, & \text{the identification horizon} \end{cases}$$

The following constraints are specified

$$u_k^{\text{min}} = \begin{bmatrix} 50.0 \\ 6.0 \\ 0.5 \end{bmatrix}, \quad u_k^{\text{max}} = \begin{bmatrix} 54.0 \\ 8.0 \\ 0.7 \end{bmatrix} \; \forall \; k > 0 \tag{4.68}$$

$$\Delta u_k^{\text{min}} = \begin{bmatrix} -0.5 \\ -0.15 \\ -0.05 \end{bmatrix}, \quad \Delta u_k^{\text{max}} = \begin{bmatrix} 0.5 \\ 0.15 \\ 0.05 \end{bmatrix} \; \forall \; k > 0 \tag{4.69}$$

$$y_k^{\text{min}} = \begin{bmatrix} 1.5 \\ 33 \end{bmatrix}, \quad y_k^{\text{max}} = \begin{bmatrix} 1.7 \\ 40 \end{bmatrix} \; \forall \; k > 0 \tag{4.70}$$

Simulation results are illustrated in Figures 4.11 and 4.12.

4.25 References

Di Ruscio, D. (1997). A method for identification of combined deterministic stochastic systems. In: *Applications of Computer Aided Time Series Modeling*, Lecture Notes in Statistics 119, Eds. M. Aoki and A. M. Havenner, Springer Verlag, ISBN 0-387-94751-5.

Di Ruscio, D. and A. Holmberg (1996). Subspace identification for dynamic process analysis and modeling. *Control Systems 96*, Halifax, Nova Scotia, May 1996.

Golub, G. H. and C. F. Van Loan (1983). *Matrix Computations.* North Oxford Academic Publishers Ltd.

Larimore, W. E. (1983). System identification, reduced order filtering and modeling via canonical variate analysis. *Proc. of the American Control Conference*, San Francisco, USA, pp. 445-451.

Larimore, W. E. (1990). Canonical Variate Analysis in Identification, Filtering and Adaptive Control. *Proc. of the 29th Conference on Decision and Control*, Honolulu, Hawaii, December 1990, pp. 596-604.

Ljung, L. (1991). *System Identification Toolbox.* The Mathworks, Inc.

Faurre, P. L. (1976). Stochastic realization algorithms. In: *System Identification: Advances and Case Studies*, (eds. R. K. Mehra and D. G. Lainiotis), Academic Press.

Kung, S. Y. (1978). A new identification and Model Reduction Algorithm via Singular Value Decomposition. *Conf. on Circuits, Systems and Computers*, Pacific Grove, CA, November 1978, pp. 705-714.

Moore, B. C. (1981). Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction. *IEEE Trans. on Automatic Control*, Vol. AC-26, pp. 17-31.

Van Overschee, P. and B. De Moor (1994). N4SID: Subspace Algorithms for the Identification of Combined Deterministic Stochastic Systems. *Automatica*, vol. 30, No. 1, pp.75-94.

Van Overschee, P. (1995). *Subspace Identification: theory-implementation-application.* PhD thesis, Katholieke Universiteit Leuven, Belgium.

Van Overschee, P. and B. De Moor (1995). A Unifying Theorem for Three Subspace System Identification Algorithms. *Automatica*, vol. 31, No. 12, pp. 1853-1864.

Verhagen, M. (1994). Identification of the deterministic part of MIMO state space models given on innovations form from input output data. *Automatica*, vol. 30, No. 1, pp. 61-74.

Viberg, M. (1995). Subspace-Based Methods for the Identification of Linear Time-invariant Systems. *Automatica*, vol. 31, No. 12, pp. 1835-1851.

Wold, H. (1966). Non-linear estimation by iterative least squares procedures. *Researsch papers in statistics*, Ed. David, F. Wiley, New York, pp. 411-444.

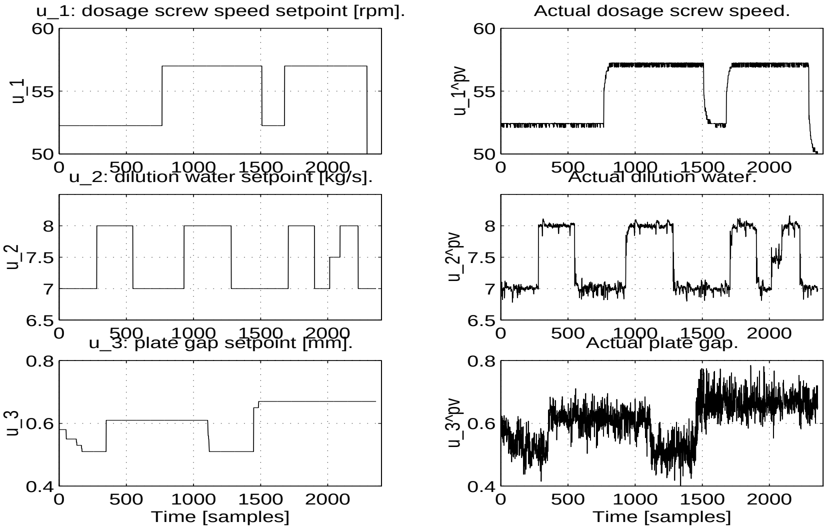


Figure 4.9: Input time series from a TMP plant at Union Bruk. The inputs are from an experimental design. The manipulable input variables are u_1, u_2 and u_3 These inputs are set-points to local input controllers. The outputs from the local controllers (controlled variables) are shown to the left and denoted u_1^{pv}, u_2^{pv} and u_3^{pv}.

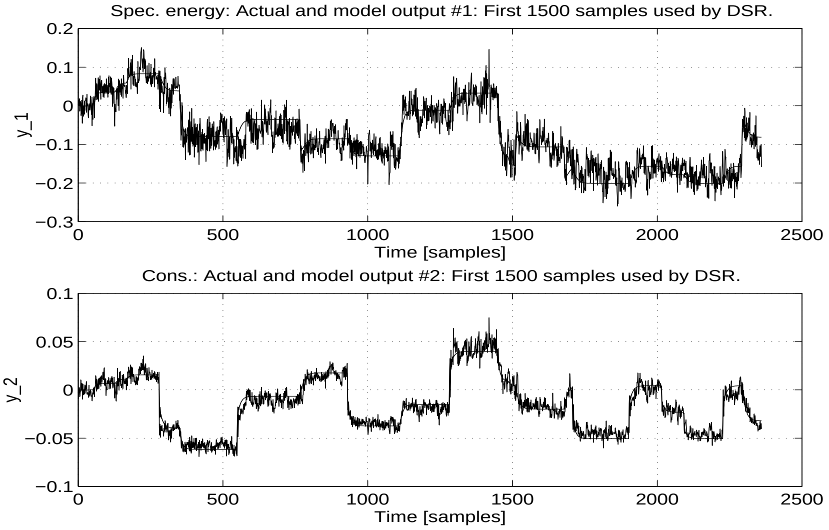


Figure 4.10: Actual and model output time series. The actual output time series is from a TMP plant at Union Bruk. The corresponding input variables are shown in Figure 4.9.

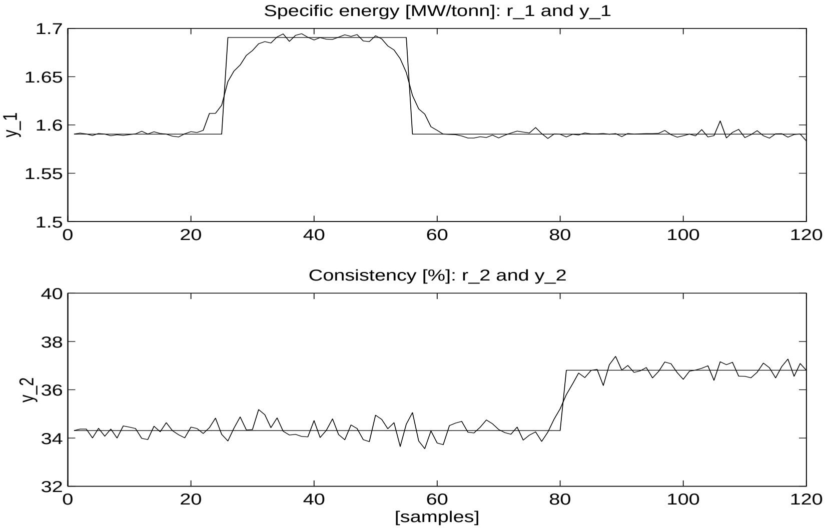


Figure 4.11: Simulation results of the MPC algorithm applied on a TMP refiner. The known references and process outputs are shown.

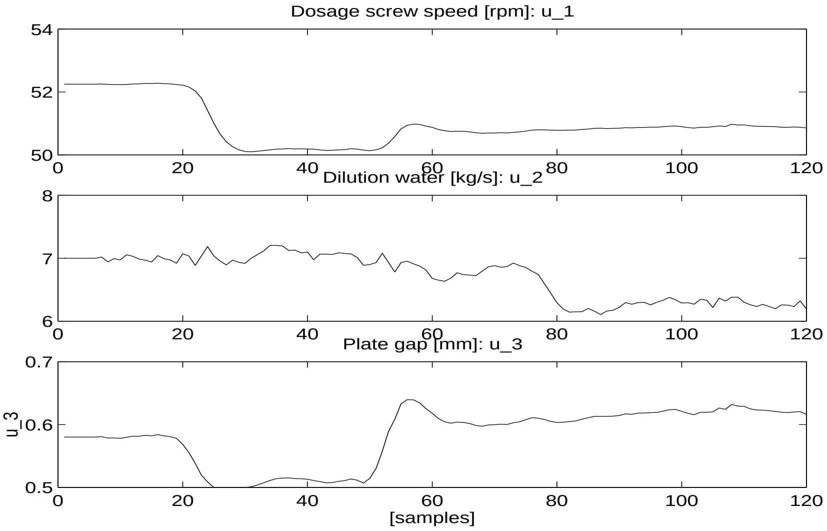


Figure 4.12: Simulation results of the MPC algorithm applied on a TMP refiner. The optimal control inputs are shown.

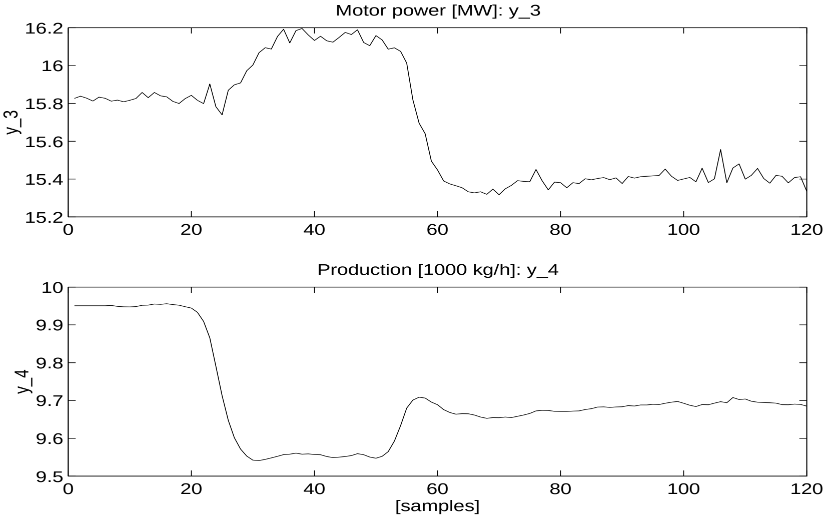


Figure 4.13: Simulation results of the MPC algorithm applied on a TMP refiner.

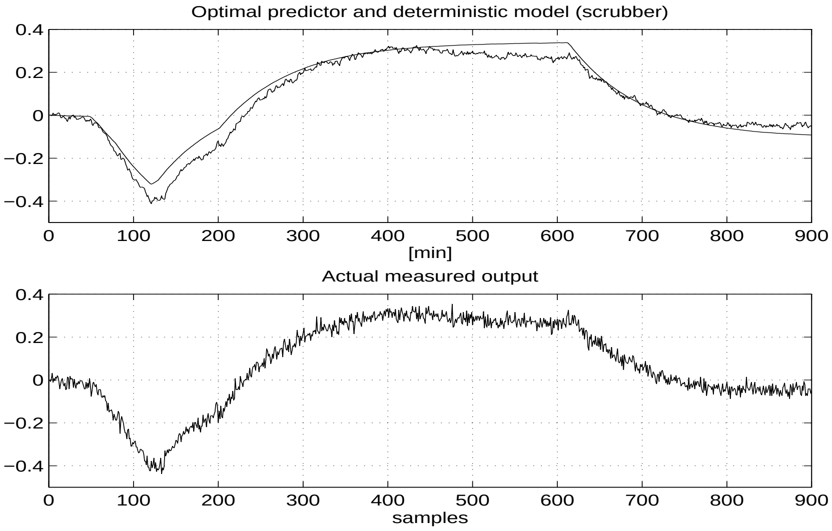


Figure 4.14: Simulated model and optimal prediction of process output (upper) and actual PH (lower).

Chapter 5

Subspace Identification for Dynamic Process Analysis and Modeling

Abstract

A recently developed subspace identification method for robust data analysis and modeling of dynamic and static systems from known observations will be presented. The method is used in order to analyze and modeling time series from an Thermo Mechanical Pulping refiner line at Norske Skog, Norway.

Keywords: Subspace identification; time series; TMP refiners; power; specific energy; system identification; practical results.

5.1 Introduction

The subspace method discussed in this work, [3] - [6], has a number of advantages compared to the traditional prediction error approaches for system identification. The method is practical and easy to use, it is non-iterative, always convergent and completely data driven. See also [8] and the survey paper [9].

The method detects system dynamics (system order) and it can be viewed as a generalization of traditional multivariate data analysis and modeling methods such as Principal Component Analysis (PCA) and Regression (PCR) and Partial Least Squares (PLS). These last methods all deals with analyzing and modeling of static relationship between variables and are well known methods in e.g., chemometrics and econometrics. However, these methods are insufficient in order to modeling data from dynamical systems, i.e., systems where the direction of time is important.

This implies that PCA, PCR and PLS deals with a special case analysis and modeling problem of the problem which is solved by the subspace methods.

The first step in the subspace method is a data compression step. The possibly large number of observations (data) are compressed to a much smaller matrix by a standard lower Left Q-orthogonal (LQ) decomposition. This smaller lower triangular matrix contains the information of the system (process and sensors) which generated the data. The system dynamics as well as the complete model for the system is extracted from this smaller matrix. A short description of how to use the method will be given in this paper.

Note that the method works for systems where the outputs are driven by both known input variables and unknown process disturbance variables. The unknown variables may be high frequency or colored noise variables. The method is believed to be a valuable tool for analyzing and modeling of data from industrial processes, in particular dynamic processes. The subspace method is used on some important processes in the pulp and paper industry in Norway, e.g. a plate column scrubber; a paper machine; the Thermo Mechanical Pulping (TMP) process.

The process variables measured on a two stage TMP refiner line at Norske Skog, Skogn, Norway are analyzed and modeled by the method. In particular, a state space model for the refiner motor power is constructed from observed data (time series). This results will be presented in the paper. The model for the 1st stage refiner has three input variables, i.e., plug screw speed, dilution water and plate gap. The power model will be presented in this paper.

The rest of this paper is organized as follows. Some preliminary system and problem definitions are stated in Section 5.2. A description of the subspace method is presented in Section 5.3. An application of the subspace method [6] is presented in Section 5.4. The actual process is a two-stage TMP refiner line at Norske Skog, Skogn, Norway. A state space model for the TMP refiner motor power is identified. Some conclusions follows in Section 5.5.

5.2 Preliminary definitions

5.2.1 System definition

Assume that the underlying system can be described by a discrete-time, time invariant, linear state space model (SSM) of the form

$$x_{k+1} = Ax_k + Bu_k + Ce_k \tag{5.1}$$

$$y_k = Dx_k + Eu_k + e_k \tag{5.2}$$

where the integer $k \geq 0$ is discrete-time, $x \in \mathbb{R}^n$ is the state vector with initial value x_0, $y \in \mathbb{R}^m$ is the system output, $u \in \mathbb{R}^r$ is the system input, $e \in \mathbb{R}^m$ is an unknown innovations process of white noise, assumed to be covariance stationary, with zero mean and covariance matrix $\mathrm{E}(e_k e_k^T) = \Delta$. The constant matrices in the SSM are of appropriate dimensions. A is the *state transition* *matrix*, B is the *external input* matrix, C is the *Kalman gain* matrix, D is the *output* matrix and E is the direct *control input to output* (feed-through) matrix. We will assume that (D, A) is an observable pair.

The innovations model, Equations (5.1) and (5.2), is discussed in e.g. [7], [1].

5.2.2 Problem definition

The problem investigated in this paper is to identify a state space model, including the system order (n), for both the deterministic part and the stochastic part of the system, i.e., the quadruple matrices (A, B, D, E) and the double matrices (C, Δ), respectively, directly from known system input and output data vectors (or time series) defined as

$$\left.\begin{array}{lll} u_k & \forall & k = 0, \ldots, N-1 \\ y_k & \forall & k = 0, \ldots, N-1 \end{array}\right\} \textbf{Known data}$$

Note that the variables can be linearly dependent as well as independent. In continuous time systems the matrix E in Equation (5.2) is usually zero. This is not the case in discrete time systems due to sampling. However, E can be forced to be zero by including a structure constraint in the algorithm, [6].

5.3 Description of the method

Given sequences with process input and output raw data. The first step in a data modeling procedure is usually to analyze the data for trends and time delays. Trends should preferably be removed from the data and the time series should be adjusted for time delays. The trend of a time series can often be estimated as the sample mean which represents some working point. Data preprocessing is not necessary but it usually increase the accuracy of the estimated model.

The simplicity of the subspace method [6] will be illustrated in the following. Assume that the data is adjusted for trends and time delays. Organize the process output and input data vectors y_k and u_k as follows

$$\overbrace{Y = \begin{bmatrix} y_0^T \\ y_1^T \\ \vdots \\ y_{N-1}^T \end{bmatrix} \in \Re^{N \times m}}^{\textbf{Known} \text{ data matrix of output variables}} \tag{5.3}$$

$$U = \overbrace{\begin{bmatrix} u_0^T \\ u_1^T \\ \vdots \\ u_{N-1}^T \end{bmatrix}}^{\textbf{Known} \text{ data matrix of input variables}} \in \Re^{N \times r} \tag{5.4}$$

The problem of identifying a complete (usually) dynamic model for the process can be illustrated by the following function (similar to a matlab function).

$$\left[\; A, \; B, \; C, \; D, \; E, \; \Delta \; \right] = \text{DSR}(Y, U, L) \tag{5.5}$$

where the sixfold matrices (A, B, C, D, E, Δ) are the state space model matrices in Equations 5.1 and 5.2. The algorithm name **DSR** stands for Deterministic and Stochastic model Realization and identification, see [3]- [6] for details. L is a positive integer parameter which should be specified by the user. The parameter L defines an upper limit for the unknown system order n. The user must chose the system order by inspection of a plot with Singular Values (SV) or Principal Angles (PA). The system order n is identified as the number of "non-zero" SV's or "non-zero" PA's. See Figure 5.1 for an illustration of the order selection.

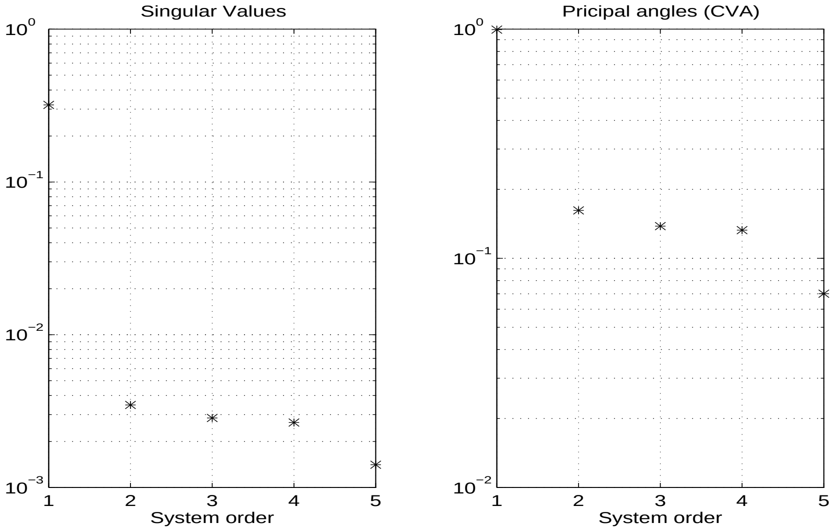


Figure 5.1: Singular Values (SV) and Principal Angles (PA) for dynamic order selection computed from time series from a PH-control loop in a plate column scrubber at Union Co., Skien, Norway. We have specified $L = 5$. A reasonable choice for the system order is $n = 1$, since the first SV is large compared to the other SV's.

5.4 Modeling the power

The main refiner motor power is believed to be one of the most important refiner measurements which is available. As an example, the motor power is the most important variable in order to estimate the consistency, the specific energy and the steam which is generated in the refiner. The motor power is an output from the refining process. The power should be modeled as a function of the refiner input variables. A combined mechanistic and empirical power model is discussed in [2]. For a CD refiner we have the following input variables; dosage screw speed n_s; plate gap for the plane zone s_1; plate gap for the conical zone s_2; flow rate of dilution water to the refiner inlet or plane zone q_{w1} and to the conical zone q_{w2}. The power model can be formulated as

$$y = y(n_s, q_{w1}, q_{w2}, s_1, s_2) \tag{5.6}$$

where y is the refiner motor power. Another procedure is to identify a model for the refiner motor power, directly from measured input and output data. This approach will be discussed in the rest of this section.

5.4.1 Refiner variables and model

The manipulable input variables are defined as follows

$$u \in \Re^3 \ := \left\{ \begin{array}{ll} u_1 & \text{Plug screw speed, } [rpm] \\ u_2 & \text{Total flow of dilution water, } [\frac{kg}{s}] \\ u_3 & \text{Plate gap, conical zone, } [mm] \end{array} \right. \tag{5.7}$$

The linearized power model is assumed to be on innovations form, Equation 5.1 and 5.2. The innovations model can be written as an optimal predictor for the power output y_t given all past process inputs and outputs. The optimal prediction of the power can be computed from

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + C(y_t - Dx_t - Eu_t) && (5.8) \\ \hat{y}_t &= Dx_t + Eu_t && (5.9) \end{aligned}$$

where $\hat{y}_t$ is the optimal prediction of the present output y_t given all past outputs $(..., y_{t-2}, y_{t-1})$ and the present and all past inputs $(..., u_{t-1}, u_t)$. The problem is to find the model matrices (A, B, C, D, E) and the covariance matrix of the innovations noise process, $\Delta = cov(e_t e_t^T)$.

We will also mention that the combined deterministic and stochastic model, Equations 5.1 and 5.2, can be separated into two parts. One deterministic part (with output y_t^d) which is driven by the known process input variables u_t and one stochastic part (with output y_t^s) driven by the unknown innovations noise process e_t. The actual output is then $y_t = y_t^d + y_t^s$. See e.g., [2] p. 36.

We will in this work focusing on modeling the 1st stage refiner motor power. It is often not clear whether a process variable should be defined as an input or output variable to a model. An example of this is the problem of modeling the 2nd stage refiner. It is assumed that the 1st stage motor power y_t contains information of the disturbances which affects the 2nd stage refiner. Hence, the inclusion of the output y_t as an additional input to the power model for the 2nd stage refiner is reasonable. We will not elaborate this further.

5.4.2 Numerical results

The results in this section is based on the time series from Norske Skog, Skogn, Norway. The following trends (working point, nominal values) are removed from the time series prior to the numerical analysis.

$y_0 = 17.69$	nominal motor power
$u_0^1 = 44.08$	nominal plug screw speed
$u_0^2 = 7.15$	nominal sum of dilution water
$u_0^3 = 0.83$	nominal plate gap CD zone

Case 1: Input variables u_1 and u_2.
The following MISO 1st stage refiner power model is determined by the use of the DSR algorithm. The parameter to DSR was $L = 1$.

$$x_{k+1} = Ax_k + B \begin{bmatrix} u_k^1 \\ u_k^2 \end{bmatrix} + Ce_k \tag{5.10}$$

$$y_k = Dx_k + E \begin{bmatrix} u_k^1 \\ u_k^2 \end{bmatrix} + e_k \tag{5.11}$$

where

$$\begin{array}{ll} A = 0.9557 & B = \begin{bmatrix} 0.0227 & -0.0558 \end{bmatrix} \\ D = 1.0 & E = \begin{bmatrix} 0.0198 & -0.0335 \end{bmatrix} \\ C = 0.4057 & \Delta = 0.0472 \end{array} \tag{5.12}$$

The steady state gain for the deterministic part, $H_d(1)$, and for the stochastic part, $H_s(1)$, is given by

$$\begin{aligned} H_d(1) &= D(I - A)^{-1}B + E \\ &= \begin{bmatrix} 0.5328 & -1.2919 \end{bmatrix} \end{aligned} \tag{5.13}$$

$$H_s(1) = D(I - A)^{-1}C + I = 10.14 \tag{5.14}$$

The signal to nose ratio is given by

$$S/N = \frac{cov(y^d)}{cov(y^s)} = 2.45 \tag{5.15}$$

The prediction error

$$J = cov(y^s) = cov(y - y^d) = 0.1367 \tag{5.16}$$

See Figures 5.2 - 5.10 for illustrations.

Case 2: Input variables u_1, u_2 and u_3.
The following MISO refiner power model is determined by the use of the DSR algorithm. The parameter to DSR is $L = 1$.

$$x_{k+1} = Ax_k + B \begin{bmatrix} u_k^1 \\ u_k^2 \\ u_k^3 \end{bmatrix} + Ce_k \tag{5.17}$$

$$y_k = Dx_k + E \begin{bmatrix} u_k^1 \\ u_k^2 \\ u_k^3 \end{bmatrix} + e_k \tag{5.18}$$

where

$$\begin{array}{ll} A = 0.9539 & B = \left[\begin{array}{ccc} 0.0236 & -0.0526 & 0.0267 \end{array}\right] \\ D = 1.0 & E = \left[\begin{array}{ccc} 0.0207 & -0.1459 & 0.014 \end{array}\right] \\ C = 0.4090 & \Delta = 0.0474 \end{array} \tag{5.19}$$

The steady state gain for the deterministic part, $H_d(1)$, and for the stochastic part, $H_s(1)$, is given by

$$\begin{aligned} H_d(1) &= D(I-A)^{-1}B + E \\ &= \left[\begin{array}{ccc} 0.5315 & -1.2860 & -1.9500 \end{array}\right] \end{aligned} \tag{5.20}$$

$$H_s(1) = D(I-A)^{-1}C + I = 9.87 \tag{5.21}$$

The signal to nose ratio is given by

$$S/N = \frac{cov(y^d)}{cov(y^s)} = 2.48 \tag{5.22}$$

The prediction error

$$J = cov(y^s) = cov(y - y^d) = 0.1354 \tag{5.23}$$

The results from this model is not plotted. The influence from the plate gap is approximately zero compared to the influence from the screw speed and the dilution water. And the model is virtually the same as the model presented above in Case 1.

5.5 Conclusion

A state space model for a TMP refiner motor power is constructed from time series. The model have three input variables, plug screw speed, dilution water and plate gap.

Bibliography

[1] M. Aoki, "State Space Modeling of Time Series," Springer-Verlag, 1990.

[2] D. Di Ruscio, *Topics in model based control with application to the thermo mechanical pulping process*, Dr. Ing. thesis. Report no. 93-46-W. Division of Engineering Cybernetics, NTH, N-7034 Trondheim, 1993.

[3] D. Di Ruscio, "A Method for identification of combined deterministic and stochastic system," *In Proceedings of the European Control Conference, ECC95*, Roma, Italy, September 5-8, 1995.

[4] D. Di Ruscio, "Methods for the identification of state space models from input and output measurements," *SYSID 94, The 10th IFAC Symposium on System Identification*, Copenhagen, July 4 - 6, 1994. Also in: *Modeling, Identification and Control*, Vol. 16, no. 3, 1995.

[5] D. Di Ruscio, "Identification and Realization of Combined Deterministic and Stochastic Systems," To be included in: *Lecture Notes in Time Series Analysis*, (edited by M. Aoki, UCLA,USA), 1996.

[6] "**DSR**: Algorithm for combined deterministic and stochastic system identification and realization of state space models from observations," *Program commercial available by Fantoft Process AS, Box 306, N-1301 Sandvika.*

[7] P. L. Faurre, "Stochastic Realization Algorithms," In: *System Identification: Advances and Case Studies*, (edited by R. K. Mehra and D. G. Lainiotis), Academic Press, 1976.

[8] W. E. Larimore, "Canonical Variate Analysis in Identification, Filtering and Adaptive Control," *Proc. of the 29th Conference on Decision and Control*, Honolulu, Hawaii, december 1990, pp. 596-604.

[9] M. Viberg, "Subspace-Based Methods for the Identification of Linear Time Invariant Systems," *Automatica*, vol. 31, No. 12, pp. 1835-1851.

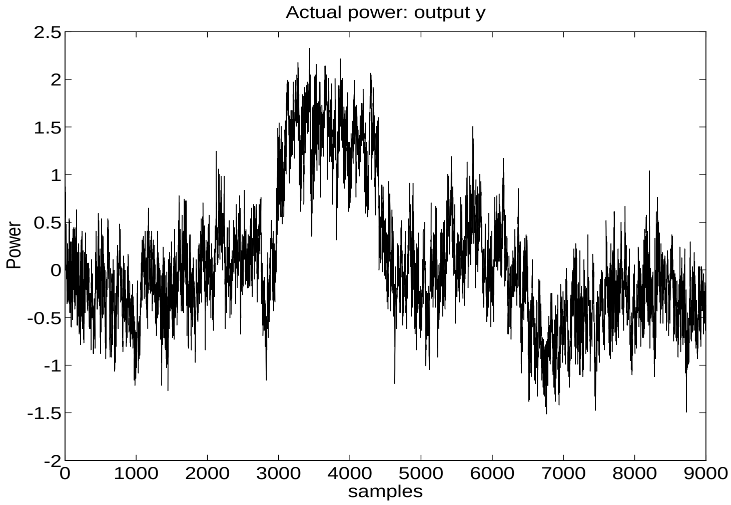


Figure 5.2: 1 st stage refiner motor power, R27.

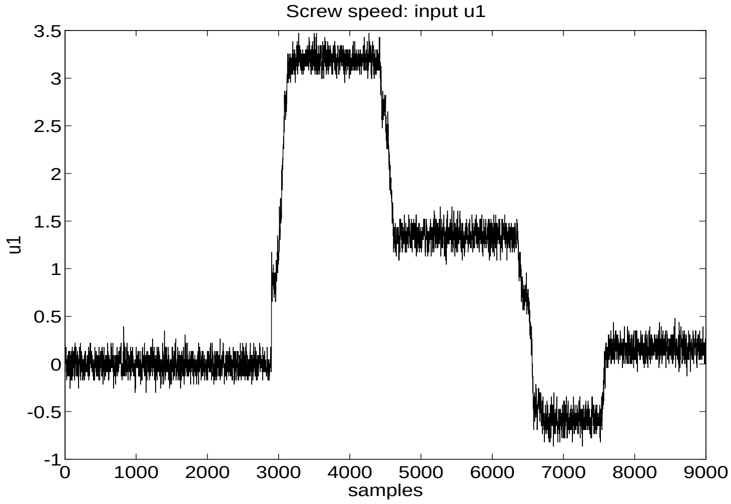


Figure 5.3: Plug screw speed, u_1.

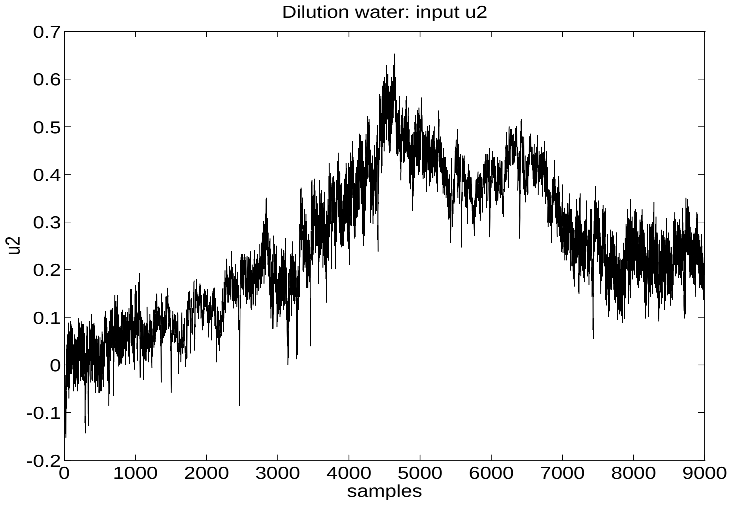


Figure 5.4: Flow of Dilution water, u_2.

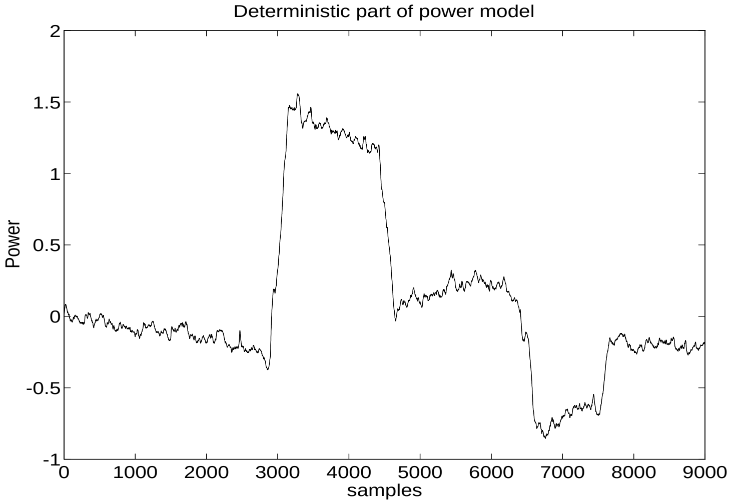


Figure 5.5: Deterministic part of the power model output, y^d.

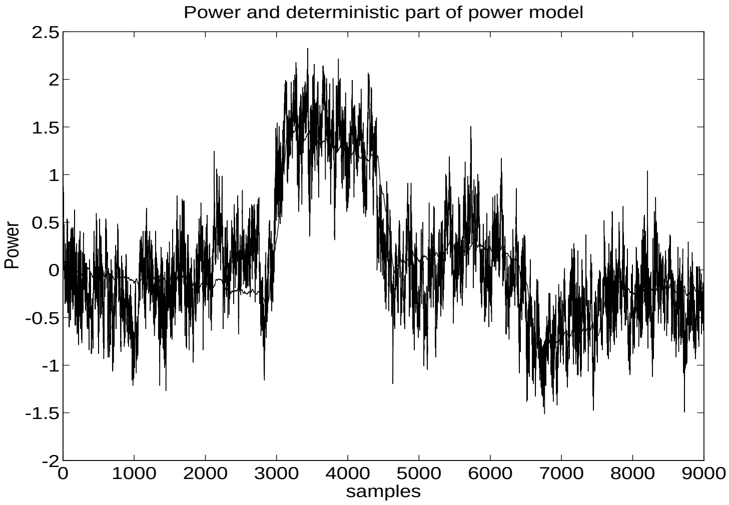


Figure 5.6: Deterministic part of the power model and actual measured power.

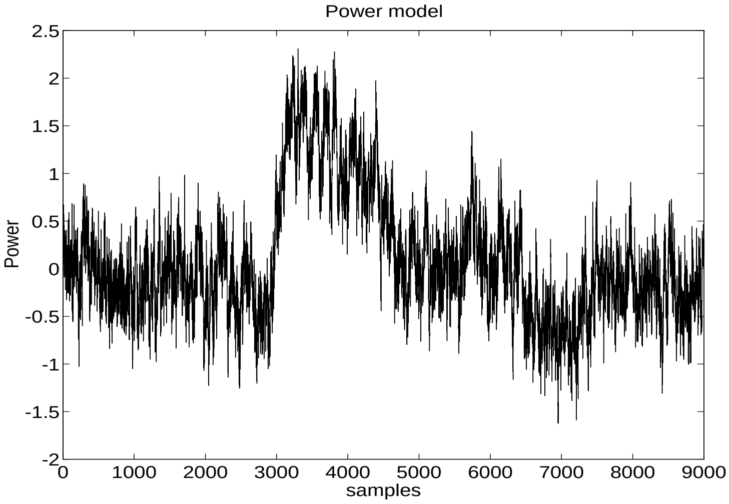


Figure 5.7: The power model, sum of deterministic part and stochastic part.

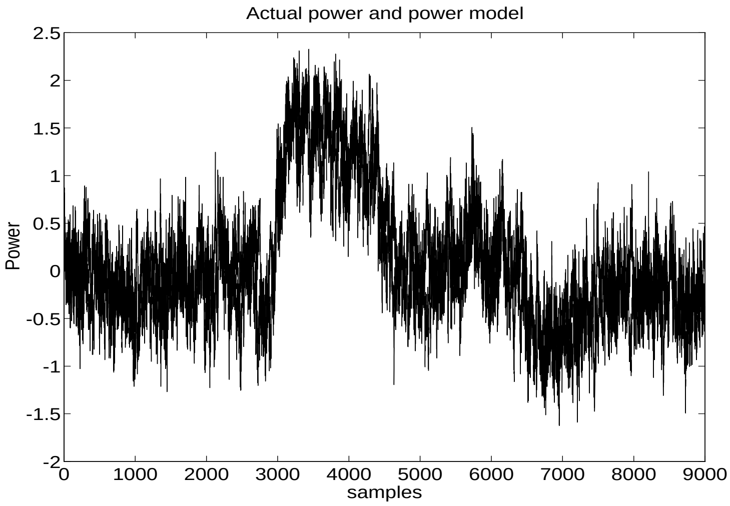


Figure 5.8: The power model and the actual measured power.

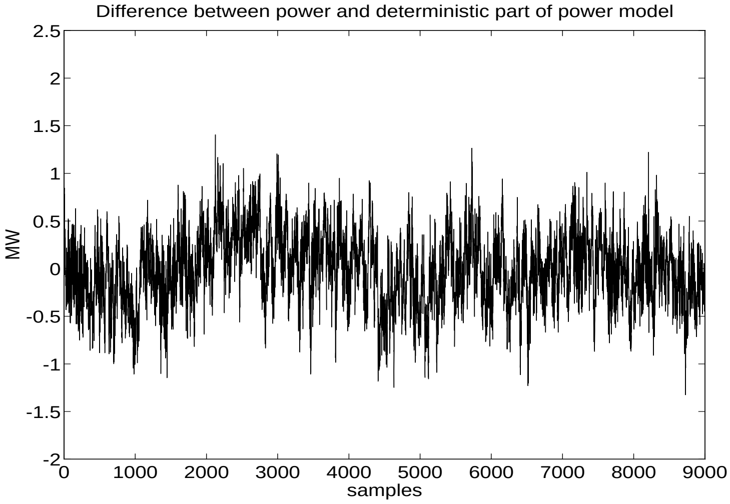


Figure 5.9: The residual between the actual measured power and the deterministic part of the estimated power model, i.e. $y^s = y - y^d$.

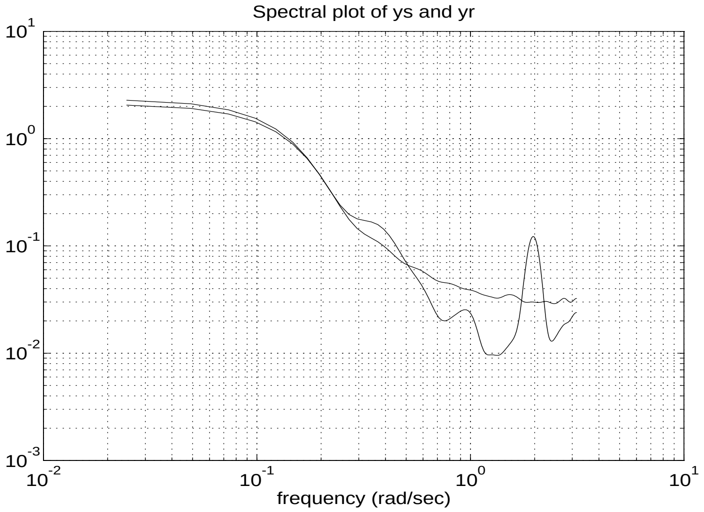


Figure 5.10: The estimated frequency function of the actual stochastic part of the power model, i.e. $y^s = y - y^d$, and $\hat{y}^d$.

Chapter 6

Dynamic Subspace Modeling (DSR) and Static Multivariate Analysis and Regression (PCA,PCR,PLS)

Abstract

It is shown that a solution to the multivariate analysis and regression problem is a special case of a recently developed subspace method for analysis and identification of combined deterministic and stochastic dynamical systems. This method detects system dynamics. A numerically stable solution to the multivariate analysis and regression problem is also presented. **Keywords:** System identification; subspace methods; dynamic systems; combined deterministic and stochastic systems; multivariate analysis and regression; PCR; PLS; data analysis and modeling.

6.1 Introduction

The theory of multivariate analysis and regression is a research field in many disciplines, e.g. in chemistry and economics. Econometrics and chemometrics are well established disciplines. A problem in many disciplines, including economics and chemistry, is to postulate relationships between variables. It is the purpose of econometrics to postulate the relationship between economic variables. In the same way one can state that a purpose of chemometrics is to postulate relationships between chemical variables. This problem is of great importance in general.

Solution procedures such as Partial Least Squares (PLS) and Principal Component regression (PCR) are effective tools for analysis and regression of parameters in systems which is static. I.e. these tools are effective to analyze and modeling variables that are static related.

On the other side it can be very ineffective to use these methods to analyze, modeling and identify (input output) relationships between variables in a dynamical system.

One goal of statistical planning of system experiments is to produce data which is rich of system information. The system input variables which are manipulable are perturbed in such a way that some invert-ability requirement of the manipulable variables is satisfied. However, it is important that all the parameters in the system is exited (in order to identify them). The system input and output variables are observed over the experimental period.

Usually all systems are dynamic in nature. In order to use PLS and PCR methods, to analyze the system input and output behavior, one have to wait for all the transients to die out in order to e.g. identify the gain from the variables which are manipulated to the observed variables. All the observations in the transient period are wasted. The result of this identification procedure usually gives a wrong conclusion about the gain. The system output is in addition to the manipulable variables driven by noise. It is known from the identification theory that the system gain can be identified before the transients in the system have died out. PLS and PCR methods can not detect dynamics and handle dynamical systems in a proper way. For such systems, methods which can detect the dynamics, including the system order are prefered.

A so called subspace method for identification of combined deterministic and stochastic systems directly from the data is presented in Di Ruscio (1994), Di Ruscio (1994a) and Di Ruscio (1995). The starting point for this method is to allow a more general model structure for the system in which the standard static multivariate analysis and regression problem is a special case.

A fundamental property of this method is that the system dynamics (system order) are identified from the data. If the system order is identified to be zero then a typical static regression problem results, in which PLS or PCR techniques can be applied. The system order is not the same as the number of principal components in PCR and number of factors (components) in PLS. The number of principal components and PLS factors are measures of the effective rank of a sequence of input (or output) variable observations. It is more or less the same as what is called persistently exiting analysis of the input signal in classical identification theory.

Section 6.2 gives a system description of both the static and the dynamic modeling problem and how the data is ordered into certain data matrices for which informations about the underlying system can be retrieved.

Section 6.4 gives a review of the solution to the more general dynamic system analysis and modeling problem. The first step is to compress the data down to the square root of some covariance matrices. The next step is to identify the system dynamics (system order). If the system order is detected or specified to be zero then, the covariance matrices for the static problem are directly extracted. This method gives a numerically stable way of identifying the system order and the matrices in a so called combined deterministic and stochastic dynamic model.

The static analysis and regression modeling problem will be discussed in Section 6.5. In Lindgren, Geladi and Wold (1993) a covariance based PLS method is presented. The first step in this method is to compute the covariance matrices which is an ill-conditioned step. The next step is then to use a modified PLS method which works on the covariance matrices.

Another solution to this special static analysis and modeling problem will be presented in this paper. This includes a numerically stable method to compress the data down to the square root of the covariance matrices, without explicitly forming the covariance matrices. In the next step, the standard PLS, SVD or PCR methods can be used.

6.2 System description and data organization

We will in this section give a system description. A system description is important because it is the fundament of the method to be presented. A poor system description usually results in limitations of any method and algorithm. We will in Section 6.2.1 give a description of the more general combined deterministic and stochastic system. Section 6.2.1 gives a description of the less general static system which is the fundament for the traditional analysis and regression tools as PCA, PCR and PLS. The latter is a special case of the system description in Section 6.2.2.

6.2.1 Combined deterministic and stochastic description

Assume that the underlying system can be described by a discrete-time, time invariant, linear state space model of the form

$$x_{k+1} = Ax_k + Bu_k + Ce_k \tag{6.1}$$

$$y_k = Dx_k + Eu_k + e_k \tag{6.2}$$

where $k \geq 0$ is discrete-time, i.e. an integer, $x \in \Re^n$ is the state vector with initial value x_0, $y \in \Re^m$ is the system output, $u \in \Re^r$ is the system input, $e \in \Re^m$ is an unknown innovations process of white noise, assumed to be covariance stationary, with zero mean and covariance matrix $\mathrm{E}(e_k e_k^T) = \Delta$. A, B, C, D and E are constant matrices of appropriate dimensions, where (D, A) is an observable pair.

A solution to the identification problem of constructing a state space model, including the system order (n), for both the deterministic part and the stochastic part of the system i.e. the quadruple (A, B, D, E) and the double (C, Δ), respectively, directly from known input and output time series u_k and y_k.

Given the system output and input time series, $y_k \forall k = 0, \dots, N-1$ and $u_k \forall k = 0, \dots, N-1$, respectively. The state space model, Equations (6.1) and (6.2), can generally be written as

$$Y_{k+1} = \tilde{A} Y_k + \tilde{B} U_k + \tilde{C} E_k \tag{6.3}$$

where

$$Y_k = \begin{bmatrix} y_k & y_{k+1} & y_{k+2} & \cdots & y_{k+K-1} \\ y_{k+1} & y_{k+2} & y_{k+3} & \cdots & y_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{k+L-1} & y_{k+L} & y_{k+L+1} & \cdots & y_{k+L+K-2} \end{bmatrix} \in \Re^{Lm \times K} \tag{6.4}$$

$$U_k = \begin{bmatrix} u_k & u_{k+1} & u_{k+2} & \cdots & u_{k+K-1} \\ u_{k+1} & u_{k+2} & u_{k+3} & \cdots & u_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{k+L-1} & u_{k+L} & u_{k+L+1} & \cdots & u_{k+L+K-2} \\ u_{k+L} & u_{k+L+1} & u_{k+L+2} & \cdots & u_{k+L+K-1} \end{bmatrix} \in \Re^{(L+1)r \times K} \tag{6.5}$$

$$E_k = \begin{bmatrix} e_k & e_{k+1} & e_{k+2} & \cdots & e_{k+K-1} \\ e_{k+1} & e_{k+2} & e_{k+3} & \cdots & e_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_{k+L-1} & e_{k+L} & e_{k+L+1} & \cdots & e_{k+L+K-2} \\ e_{k+L} & e_{k+L+1} & e_{k+L+2} & \cdots & e_{k+L+K-1} \end{bmatrix} \in \Re^{(L+1)m \times K} \tag{6.6}$$

The number of columns in Y_k, U_k and E_k are $K = N - L - k + 1$. Each column in these matrices can be interpreted as extended output, input and noise vectors, respectively. K can be viewed as the number of samples in these extended time series. We also have that $L < K < N$. L is a user specified parameter which is the number of block rows in the extended observability matrix which will be determined.

6.2.2 Purely static description

A model suitable for PLS and PCR problems can be written as the following linear steady state equation.

$$y_t^T = u_t^T E^T + e_t^T \tag{6.7}$$

where $y_t \in \Re^m$ is a vector of known output variables at observation number t. $u_t \in \Re^r$ is a vector of known input variables and $e_t \in \Re^m$ is a vector of unknown white noise variables, assumed to be covariance stationary, with zero mean and covariance matrix $\mathrm{E}(e_t e_t^T) = \Delta$.

Assume a number of N observations, i.e. $t = 1, ..., N$. The N equations can then be formed into the following matrix equation with notations which is common in multivariate regression theory.

$$\mathbf{Y} = X\mathbf{B} + \mathbf{E} \tag{6.8}$$

where $Y \in \Re^{N \times m}$ is a known matrix of observations. The vector y_t^T is also called a vector of responses at observation number t. $X \in \Re^{N \times r}$ is a known matrix of input variables. Each row in X (i.e. u_t^T) is also called a vector of predictor variables at observation number t. The matrix $\mathbf{B} \in \Re^{r \times m}$ is the matrix of unknown parameters (also called regression coefficients).

Both data matrices Y and X can be corrupted with (measurements) noise. The principal components, detected as a first step in the PCR method, are the number of linearly independent columns in the data matrix X. This possibly rank deficiency problem can e.g. be detected as the number of "non-zero" singular values which results from the Singular Value Decomposition (SVD) of the covariance matrix estimate $\frac{1}{N}X^TX$.

The (principal) components in the PLS method are computed by taking advantage of both Y and X variables.

6.3 Dependent input or output data

We will in this section illustrate how a sequence of dependent output (or input) variables can be projected to a sequence of dependent variables.

Assume that a sequence of possibly dependent input and (or) output variables are known

$$\left.\begin{array}{lllll} \mathcal{U}_k & \in & \Re^s & \forall & k=1,2,...,N \\ \mathcal{Y}_k & \in & \Re^p & \forall & k=1,2,...,N \end{array}\right\} \quad \textbf{known dependent data}$$

We will illustrate the analysis only for the output variables because the analysis for the input variables ar similar. Define the data-matrices of possibly dependent output variables as follows

$$\mathcal{Y} = \overbrace{\begin{bmatrix} \mathcal{Y}_1^T \\ \mathcal{Y}_2^T \\ \vdots \\ \mathcal{Y}_N^T \end{bmatrix}}^{p} \tag{6.9}$$

The effective rank of this data matrix can be analyzed by the SVD.

$$\mathcal{Y} = \overbrace{\begin{bmatrix} \overbrace{U_1}^{m} & U_2 \end{bmatrix}}^{p} \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^T \tag{6.10}$$

The number m independent output variables are determined by inspection of the non-zero or large singular values. A data matrix of dependent output variables can then be determined as

$$\mathbf{Y} = \overbrace{\begin{bmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_N^T \end{bmatrix}}^{m} = \mathcal{Y} V_1 S_1^{-1} \tag{6.11}$$

A sequence of dependent outputs (and inputs) can then be written as

$$\left.\begin{array}{lllll} u_k & \in & \Re^r & \forall & k=1,2,...,N \\ y_k & \in & \Re^m & \forall & k=1,2,...,N \end{array}\right\} \quad \textbf{known independent data}$$

6.4 The combined deterministic and stochastic problem

6.4.1 Data compression

Define the following QR decomposition

$$\frac{1}{\sqrt{K}}\tilde{Y} = \frac{1}{\sqrt{K}}\begin{bmatrix} U_L \\ W_1 \\ Y_L \\ Y_{L+1} \end{bmatrix} = \begin{bmatrix} R_{11} & 0 & 0 & 0 \\ R_{21} & R_{22} & 0 & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix}\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix} = RQ \quad (6.12)$$

where

$$R \quad \in \Re^{(r(L+1)+ni+2mL)\times(r(L+1)+ni+2mL)} \quad (6.13)$$

$$Q \quad \in \Re^{(r(L+1)+ni+2mL)\times K} \quad (6.14)$$

The matrix W_1 is a matrix of instrumental variables which are uncorrelated with the noise variables in E_L and sufficiently correlated with the informative part of the data.

$$W_1 = \begin{bmatrix} Y_0 \\ U_0 \end{bmatrix} \quad (6.15)$$

We can now remove the noise matrix E_L from Equation (6.3) by post multiplying with $\frac{1}{K}W_1^T$. We have from (6.12) that

$$\lim_{K\to\infty}\frac{1}{K}E_L W_1^T = (\lim_{K\to\infty}\frac{1}{K}E_L\begin{bmatrix} Q_1^T & Q_2^T \end{bmatrix})\begin{bmatrix} R_{21}^T \\ R_{22}^T \end{bmatrix} = 0 \quad (6.16)$$

Post multiplying Equation (6.3) with $[Q_1^T Q_2^T]$, using (6.16) and substituting for the corresponding R_{ij} sub-matrices from (6.12) gives

$$R_{42} = \tilde{A}R_{32} \quad (6.17)$$

$$R_{41} = \tilde{A}R_{31} + \tilde{B}R_{11} \quad (6.18)$$

The system order n and the system quadruple (A, B, D, E) are computed from (6.18).

The stochastic part of the system, defined by the matrices C and Δ, is computed from

$$R_{43} - \tilde{A}R_{33} = \tilde{C}E_k Q_3^T \quad (6.19)$$

$$R_{44} = \tilde{C}E_k Q_4^T \quad (6.20)$$

6.4.2 Identification of system dynamics

Identification of dynamics by principal component analysis

Algorithm 6.4.1 (System order, n, and the pair (D, A))
Given the matrices R_{42} and R_{32} which satisfy

$$R_{42} = \tilde{A}R_{32} \quad (6.21)$$

where

$$\tilde{A} = OA(O^TO)^{-1}O^T \tag{6.22}$$

and O is the extended observability matrix for the pair (A, D).

1. The system order n

Determine the SVD

$$R_{32} = USV^T \tag{6.23}$$

where

$$U = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \in \Re^{mL\times mL} \qquad S = \begin{bmatrix} S_n & 0 \\ 0 & 0 \end{bmatrix} \in \Re^{mL\times ni} \qquad V = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \in \Re^{ni\times ni} \tag{6.24}$$

where $S_n \in \Re^{n\times n}$ and n is the number of non-zero singular values of R_{32}, which is equal to the system order. n is determined by inspection of the non-zero diagonal elements of S or SS^T. If the system order is detected or specified to zero, then we have a static modeling problem, and the rest of this algorithm is skipped.

2. The extended observability matrix O for the pair (D, A).

The (extended) observability matrix can be taken directly as the first left part in U, i.e. U_1. We have

$$O = U(1 : Lm, 1 : n) \tag{6.25}$$

3. The system matrix A

The system matrix A can be determined as

$$A = O^TZ_{k+1}V \begin{bmatrix} S_n^{-1} \\ 0 \end{bmatrix} = U_1^TZ_{k+1}V_1S_n^{-1} \tag{6.26}$$

4. The system output matrix D

The matrix D can be taken as the $m \times n$ upper sub-matrix in the observability matrix O, i.e.

$$D = U(1 : m, 1 : n) \tag{6.27}$$

5. The extended system matrix $\tilde{A}$

We have

$$\tilde{A} = OA(O^TO)^{-1}O^T = R_{42}VS^{-1}O^T = R_{42}V_1S_n^{-1}U_1^T \tag{6.28}$$

△

System order from PLS factor analysis

Algorithm 6.4.2 (System order, n, and the pair (D, A))

Given the matrices R_{42} and R_{32} which satisfy

$$R_{42} = \tilde{A}R_{32} \tag{6.29}$$

where

$$\tilde{A} = OA(O^TO)^{-1}O^T \tag{6.30}$$

and O is the extended observability matrix for the pair (A, D).
1. The system order n
Determine the PLS weight matrices W and C, the loading matrix P and the "score" matrix T which satisfy

$$R_{32} = PT^T \tag{6.31}$$

$$R_{42} = CT^T \tag{6.32}$$

T^TT is a diagonal matrix. We have $W^TR_{32}R_{32}^TW = W^TPT^TTP^TW$. Hence, the singular values of R_{32} is the square root of the eigenvalues of PT^TTP^T. However, the zero eigenvalues should be reflected in the diagonal matrix T^TT. The system order (n) is the number of non-zero singular values of R_{32}. n is determined by inspection of the non-zero singular values of R_{32} or the the number of non-zero diagonal elements in T^TT. If the system order is detected or specified to zero, then we have a static modeling problem, and the rest of this algorithm is skipped.

Extract the following sub-matrices according to the specified order

$$W =: W(:, 1:n) \ \in \Re^{mL\times n} \quad C =: C(:, 1:n) \ \in \Re^{mL\times n} \quad P =: P(:, 1:n) \ \in \Re^{mL\times n} \tag{6.33}$$

Only the matrices W, P and C are needed in the rest of the algorithm.
2. The extended observability matrix O for the pair (D, A).
The (extended) observability matrix can be taken directly as,

$$O = W(1:Lm, 1:n) \tag{6.34}$$

3. The system matrix A
The system matrix A can be determined as

$$A = W^TC(W^TP)^{-1} = O^TC(O^TP)^{-1} \tag{6.35}$$

4. The system output matrix D
The matrix D can be taken as the $m \times n$ upper sub-matrix in the observability matrix O, i.e.

$$D = W(1:m, 1:n) \tag{6.36}$$

5. The extended system matrix $\tilde{A}$
We have

$$\tilde{A} = WAW^T = OAO^T \tag{6.37}$$

△

Note that the weight matrix W is not really necessary. We can use

$$O = P \tag{6.38}$$

$$A = (P^T P)^{-1} P^T C \tag{6.39}$$

because $C = PA$ when $O = P$.

Further research
The effect of taken the system matrix as $A = C(1:n,1:n)P(1:n,1:n)^{-1}$ should be investigated, because, in this case, A is represented in a special canonical form, e.g. with certain zeros and ones in addition to the necessary number of parameters as elements in the matrix.

6.5 The static problem

The solution to the multivariate analysis and regression problem which will be presented in this section follows as a special case of the solution to the more general combined deterministic and stochastic dynamic problem which will be reviewed in Section 6.4.

The solution presented in this section should be of interest because it only works on the square root of the covariance matrices involved, e.g. the square root of $\frac{1}{N}[YX]^T[YX]$ and thereby can be defined as numerically stable.

6.5.1 Data compression

The row dimension (number of observations) of Y and X is often much larger then the column sizes (number of variables). These large data matrices can be compressed into some much smaller matrices which contains the information of the system. Note that the procedure of directly forming the covariance matrix $\frac{1}{N}[YX]^T[YX]$ is ill-conditioned due to rounding-off errors (usually in situations when N is large). We will in this section compress the data, down to the square root of the covariance matrix. (without explicitly forming the covariance matrix).

Define the following standard QR decomposition

$$\frac{1}{\sqrt{N}}\tilde{Y} = \frac{1}{\sqrt{N}}\begin{bmatrix} X^T \\ Y^T \end{bmatrix} = \begin{bmatrix} R_{11} & 0 \\ R_{21} & R_{22} \end{bmatrix}\begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = RQ \tag{6.40}$$

where

$$R_{11} \in \Re^{r\times r} \quad R_{21} \in \Re^{m\times r} \quad R_{22} \in \Re^{m\times m} \tag{6.41}$$

$$R \in \Re^{(r+m)\times(r+m)} \quad Q \in \Re^{(r+m)\times N} \tag{6.42}$$

The solution to the total multivariate problem is given by the triangular factors R_{11}, R_{21} and R_{22}, only. The orthogonal matrix Q is not needed. This will reduce the computational effort and storage considerably, especially when the number of observations N is large compared to the number of variables.

We have directly the following equation for the regression coefficients

$$R_{21} = \mathbf{B}^T R_{11} \tag{6.43}$$

In order to solve this equation for $\mathbf{B}$, standard PLS or PCR methods can be applied.

6.5.2 The residual

The lower triangular matrix R_{22} is the square root of the residual covariance matrix. The covariance estimate of the noise (or residuals) is given by

$$\hat{\Delta} = \frac{1}{N} E^T E = R_{22} R_{22}^T \tag{6.44}$$

6.5.3 Effective rank analysis

The principal components

The lower triangular matrix R_{11} is the square root of the covariance matrix $\frac{1}{N} X^T X$. The singular values of R_{11} is equal to the square root of the eigenvalues of $\frac{1}{N} X^T X$. Hence, the principal component analysis can be made by a singular value decomposition of R_{11}. The number of principal components is the number of "non-zero" singular values of R_{11}. The "non-zero" singular values are determined by inspection. We have

$$R_{11} = USV^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} S_a & 0 \\ 0 & S_0 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^T \tag{6.45}$$

The number of principal components a, is equal to the row size of $S_a \in \Re^{a \times a}$. S_0 contains the zero singular values or singular values which are put to zero on the diagonal. Hence

$$R_{11} = U_1 S_a V_1^T \tag{6.46}$$

PLS factors

From an iterative procedure (e.g. by the Power method for computing eigenvalues and eigenvectors) the PLS method compute the weight matrices W and C and the loading matrix P, and decomposes X and Y into

$$R_{11}^T = TP^T \tag{6.47}$$

$$R_{21}^T = TC^T \tag{6.48}$$

It is possible to determine r factors from the PLS algorithm corresponding to the columns in W. The first factor and the first column in W is the maximum eigenvalue and eigenvector of the covariance matrix $\frac{1}{N}X^TYY^TX$, respectively. The effective rank index (a) is bounded by $1 \leq a \leq r$. One idea is to estimate the rank index as the number of "non-zero" PLS factors followed by cross validation. The "non-zero" factors are estimated by inspection.

6.5.4 Score and loading vector matrices

The Principal Component Analysis (PCA) algorithm may be simply calculated as an SVD of the Data matrix $X \in \Re^{N \times r}$, e.g.,

$$X = USV^T = U_1S_1V_1^T + U_2S_2V_2^T \approx U_1S_1V_1^T \tag{6.49}$$

Here, the effective rank or the number of principal components $1 \leq a \leq r$ is the number of principal components. The a larger singular values of X is then located on the diagonal of the $S_1 \in \Re^{a \times a}$ matrix. The smaller or zero singular values are similarly located in the S_2 matrix and is hence neglected.

The score vector matrix T for X is then defined as

$$T = U_1S_1. \tag{6.50}$$

The loading vector matrix P for X is defined as

$$P^T = V_1^T. \tag{6.51}$$

Hence, we have decomposed the data matrix X into

$$X = TP^T + E, \tag{6.52}$$

where $E = U_2S_2V_2^T$ is a residual matrix representing noise or error.

6.5.5 Regression

The Ordinary Least Squares (OLS) regression

Given the linear matrix equation $Y = XB + E$ withe known data matrices Y and X the OLS regression estimate of B is

$$B_{OLS} = (X^TX)^{-1}X^TY. \tag{6.53}$$

Notice that the OLS solution is minimizing the squared Frobenius matrix norm of the error $Y - BX$, i.e., ninimizing the scalar objective

$$V = ||Y - BX||_F^2 = (Y - BX)^T(Y - BX), \tag{6.54}$$

when the error $E = Y - BX$ is $\in \Re^N$ is a vector. Otherwise, equal to the trace, i.e.

$$V = ||Y - BX||_F^2 = trace((Y - BX)^T(Y - BX)). \tag{6.55}$$

Putting the gradient $\frac{dV}{dB}$ equal to zero gives the OLS solution, B_{OLS}.

The principal component regression

Using $X \approx U_1 S_1 V_1^T$ we may formulate the PCR estimate of B as

$$B_{PCR} = V_1 S_1^{-1} U_1^T Y. \tag{6.56}$$

Using the QR decomposition and the definitions in Section 6.5 we may also define

$$\begin{aligned}\hat{\mathbf{B}} &= R_{11}(R_{11}^T R_{11})^{-1} R_{21}^T & (6.57)\\ \hat{\mathbf{B}} &= (R_{11}^T)^{-1} R_{21}^T & (6.58)\end{aligned}$$

If the number of principal components is less than r, then, we are using the pseudo-inverse in the expressions for the regression matrix, i.e. we do not invert zero or almost zero singular values. Hence

$$\mathbf{B}^T = R_{21} V_1 S_a^{-1} U_1^T \tag{6.59}$$

PLS regression

A closed form formulation of the PLS regression (given univariate $Y \in \Re^N$) solution of B may be formulated as

$$B_{PLS} = W_a(W_a^T X^T X W_a)^{-1} W_a^T X^T Y, \tag{6.60}$$

where W_a is a weighting matrix equal to the Q orthogonal matrix from the QR decomposition of the controllability (Krylov) matrix $K_a = W_a R$ of the matrix pair $(X^T X, X^T Y)$. Hence, K_a is the a component controllability matrix of the matrix pair $A := X^T X$ and $B := X^T Y$ The PLS regression matrix may then also be formulated as

$$B_{PLS} = K_a(K_a^T X^T X K_a)^{-1} K_a^T X^T Y. \tag{6.61}$$

The PLS regression may also be formulated as

$$\mathbf{B} = W(P^T W)^{-1} C^T \tag{6.62}$$

where the matrices $W \in \Re^{r \times a}$, $P \in \Re^{r \times a}$, $C \in \Re^{m \times a}$, results from the PLS algorithm. Note that $W^T W = I$.

6.6 Concluding remarks

A method for analyzing and modeling input and output relationships between variables in possibly dynamical systems is presented. This method is viewed against methods such as PCR and PLS which are frequently used for modeling and analysis of static relationships between variables.

References

Di Ruscio, D. (2000). A weighted view of the partial least-squares algorithm, *Automatica, Vol.36 pp. 831-850.*

Di Ruscio, D. (1994). Methods for the identification of state space models from input and output measurements. *SYSID 94, The 10th IFAC Symposium on System Identification*, Copenhagen, July 4 - 6.

Di Ruscio, D. (1995a). A method for the identification of state space models from input and output measurements. *Modeling, Identification and Control*, Vol.16 No. 3.

Di Ruscio, D. (1995b). A Method for Identification of Combined Deterministic and Stochastic Systems: Robust implementation. *Third European Control Conference ECC95*, September 5-8, 1995.

Lindgren, F., Geladi, P. and S. Wold (1993). The kernal algorithm for PLS. *Journal of chemometrics*, Vol. 7, 45-59.

Chapter 7

On Subspace Identification of the Extended Observability Matrix

7.1 Introduction

The lower Left Q-orthogonal (LQ) (or the Q-orthogonal, upper Right (QR)) decomposition is used in order to compute certain projection matrices in the Subspace IDentification (SID) algorithms by Verhagen (1994), Van Overschee and De Moor (1994) and Di Ruscio (1995), (1997). In this paper the Partial Least Squares (PLS) method, Wold (1966), is used to compute the projection matrices. The PLS method is a simple alternative.

7.2 Definitions

7.2.1 Notation

The projection of (the row space of) a matrix A onto (the row space of) another matrix B is denoted A/B and defined as

$$A/B \overset{\text{def}}{=} AB^T(BB^T)^\dagger B, \tag{7.1}$$

where $\dagger$ denotes the Moore-Penrose pseudo-inverse. Also define

$$B^\perp \overset{\text{def}}{=} I - B^T(BB^T)^\dagger B \tag{7.2}$$

so that $BB^\perp = 0$. The projection of the row space of A onto the orthogonal complement of the row space of B is then $AB^\perp$. Hence A is decomposed into two terms as

$$A = A/B + AB^\perp, \tag{7.3}$$

where the two matrix terms A/B and $AB^\perp$ are orthogonal to each other.

7.3 Relationship between past and future data matrices

Theorem 7.3.1 (relationship between past and future data matrices)

Define $Y_{0|J}$, $U_{0|J}$ and $E_{0|J}$ as the past output, the past input and the past noise data matrices, respectively. Similarly, define $Y_{J|L+1}$, $U_{J|L+1}$ and $E_{J|L+1}$ as the future output, the future input and the future noise data matrices, respectively.

Then we have the following relationship between the past and the future data matrices

$$Y_{J|L+1} = \begin{bmatrix} H^d_{L+1} & P^d_{L+1} & \tilde{A}^J_{L+1} \end{bmatrix} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} + \begin{bmatrix} H^s_{L+1} P^s_{L+1} \end{bmatrix} \begin{bmatrix} E_{J|L+1} \\ E_{0|J} \end{bmatrix} \tag{7.4}$$

where

$$\begin{aligned} \tilde{A}^J_{L+1} &= O_{L+1} A^J O_J^\dagger && (7.5) \\ P^d_{L+1} &= O_{L+1}(C^d_J - A^J O_J^\dagger H^d_J) && (7.6) \\ P^s_{L+1} &= O_{L+1}(C^s_J - A^J O_J^\dagger H^s_J) && (7.7) \end{aligned}$$

Proof:

See Appendix 7.10.1.

7.4 The observability matrix

One important result concerning the estimation of the extended observability matrix is presented in Theorem 4.2, Equation (52), in the paper by Di Ruscio (1996). The theorem with a new proof is presented in the following.

Theorem 7.4.1 (Realization of the extended observability matrix O_{L+1})

Define the following matrix from the known data

$$Z_{J|L+1} \overset{\text{def}}{=} (Y_{J|L+1} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) U^\perp_{J|L+1}, \tag{7.8}$$

$Z_{J|L+1}$ is related to the extended observability matrix as

$$Z_{J|L+1} = O_{L+1} X^a_J \tag{7.9}$$

where

$$X^a_J \overset{\text{def}}{=} \begin{bmatrix} A^J O_J^\dagger & C^d_J - A^J O_J^\dagger H^d_J & C^s_J - A^J O_J^\dagger H^s_J \end{bmatrix} \begin{bmatrix} Y_{0|J} \\ U_{0|J} \\ E_{0|J} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} U^\perp_{J|L+1} \tag{7.10}$$

The column space of the matrix $Z_{J|L+1}$ coincides with the column space of the extended observability matrix O_{L+1} and the system order n of the SSM is given as the dimension of the column space.
$\triangle$

Proof:

See Appendix 7.10.2 and 7.10.3 for the proof of the relationship, Equations (7.8), (7.9) and (7.10).

We will in the following give a brief discussion of rank$(Z_{J|L+1})$. $X_J^a \in \mathbb{R}^{n\times K}$ is defined in (7.10). From (7.8) we have that $Z_{J|L+1} = O_{L+1}X_J^a$. The superscript a stands for autonomous. X_J^a can be interpreted as a state sequence of an autonomeous system. The matrix $Z_{J|L+1}$ is proportional to the extended observability matrix O_{L+1}. The rank of $Z_{J|L+1}$ is equal to the rank of O_{L+1} (i.e., rank$(Z_{J|L+1}) = n$) if rank$(X_J^a) = n$ and the pair (A, D) is observable. This result follows from Sylvesters inequality. We have

$$\text{rank}(O_{L+1}) + \text{rank}(X_J^a) - n \leq \text{rank}(O_{L+1}X_J^a) \leq \min\{\text{rank}(O_{L+1}), \text{rank}(X_J^a)\}$$

which gives

$$\text{rank}(X_J^a) \leq \text{rank}(Z_{J|L+1}) \leq \min\{n, \text{rank}(X_J^a)\}$$

Hence, the question is whether X_J^a lose rank below n.

We will in the following discuss some alternative formulations of X_J^a. We have from (7.10) that

$$X_J^a = \begin{bmatrix} A^J O_J^\dagger & C_J^d & C_J^s \end{bmatrix} \begin{bmatrix} Y_{0|J} - H_J^d U_{0|J} - H_J^s E_{0|J} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \\ U_{0|J} \\ E_{0|J} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} U_{J|L+1}^\perp \tag{7.11}$$

By using Equation (7.33), this can be written as

$$X_J^a = \begin{bmatrix} A^J & C_J^d & C_J^s \end{bmatrix} \begin{bmatrix} X_0 + O_J^\dagger H_J^s E_{0|J} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^\perp \\ U_{0|J} \\ E_{0|J} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} U_{J|L+1}^\perp \tag{7.12}$$

An alternative description is found by using Equations (7.36) and (7.68)

$$X_J^a = \left[\begin{array}{ccc} A^J & C_J^d & C_J^s \end{array} \right] \left[\begin{array}{c} X_0 / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] \\ U_{0|J} \\ E_{0|J} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] \end{array} \right] U_{J|L+1}^{\perp} \tag{7.13}$$

This expression relates X_J^a to the reversed extended controllability matrices C_J^d and C_J^s for the pairs (A, B) and (A, C), respectively.

At this stage our statement is that $\text{rank}(X_J^a) = n$ if:

- the pair $(A, [B, C])$ is controllable,
- the states X_0 is sufficiently correlated with the instruments (past inputs and outputs) $U_{0|J}$ and $Y_{0|J}$,
- the input is persistent exiting of a sufficient high order,
- the past noise $E_{0|J}$ is sufficiently correlated with the instruments (past data) $U_{0|J}$ and $Y_{0|J}$.

We have the following special case results.

Lemma 7.4.1 *Assume $J \to \infty$, $K \to \infty$, $J(m+r) \leq Lm$ and that A is stable.*

Then

$$Z_{J|L+1} = (Y_{J|L+1} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right]) U_{J|L+1}^{\perp} =$$

$$O_{L+1} \left[\begin{array}{cc} C_J^d & C_J^s \end{array} \right] \left[\begin{array}{c} U_{0|J} \\ E_{0|J} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] \end{array} \right] U_{J|L+1}^{\perp} \tag{7.14}$$

and

$$\textit{rank}(Z_{J|L+1}) = n \tag{7.15}$$

if and only if the pair $A, [BC]$ is controllable.
△

Still another description is found by using (7.34)

$$X_J^a = (X_J + (A^J O_J^{\dagger} H_J^s - C_J^s) E_{0|J} \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right]^{\perp}) U_{J|L+1}^{\perp} \tag{7.16}$$

From (7.16) we have that $X_J^a = X_J U_{J|L+1}^{\perp}$ when the system is deterministic. Hence, for a deterministic system rank$(X_J^a) = n$ if the inputs are persistently exiting of a sufficiently high order.

A simple formulation can be found by using (7.33) and (7.67)

$$X_J^a = X_J / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} U_{J|L+1}^{\perp} \tag{7.17}$$

Using the definition of the projection A/B, and providing the inverse exist, and define the matrix of instruments as

$$W_1 = \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{7.18}$$

then we have

$$X_J^a = X_J U_{J|L+1}^{\perp} W_1^T (W_1 U_{J|L+1}^{\perp} W_1^T)^{-1} W_1 U_{J|L+1}^{\perp} \tag{7.19}$$

The covariance can then be written as

$$\frac{1}{K-1} X_J^a (X_J^a)^T = \frac{1}{K-1} X_J U_{J|L+1}^{\perp} W_1^T (W_1 U_{J|L+1}^{\perp} W_1^T)^{-1} W_1 U_{J|L+1}^{\perp} X_J^T \tag{7.20}$$

For purely deterministic systems or for systems with a large signal to noise ratio we have

$$\frac{1}{K-1} X_J^a (X_J^a)^T = \frac{1}{K-1} X_J U_{J|L+1}^{\perp} X_J^T \tag{7.21}$$

which is positive definite. The inputs has (partly) generated the states. The states is therefore sufficiently correlated with the inputs.

Note also that Equation (7.8) can be expressed as follows

$$\begin{aligned} Z_{J|L+1} &\overset{\text{def}}{=} (Y_{J|L+1} / \begin{bmatrix} U_{J|L+1} \\ W_1 \end{bmatrix}) U_{J|L+1}^{\perp} \\ &= Y_{J|L+1} U_{J|L+1}^{\perp} W_1^T (W_1 U_{J|L+1}^{\perp} W_1^T)^{-1} W_1 U_{J|L+1}^{\perp} \\ &= (Y_{J|L+1} U_{J|L+1}^{\perp}) / (W_1 U_{J|L+1}^{\perp}) \end{aligned} \tag{7.22}$$

when $U_{J|L+1} U_{J|L+1}^T$ and $W_1 U_{J|L+1}^{\perp} W_1^T$ are non-singular matrices.

Proof:

See Appendix 7.10.4.

7.5 Standard deviation of the estimates

The subspace methods **DSR**, **CVA** and **MOESP** are shown to compute consistent (pole) estimates. However, in some circumstances the estimates shows a larger variability (standard deviation) than the estimates from **PEM**. This problem is addressed in this section.

N4SID (Van Overschee and De Moor (1995)) works in general only for white noise inputs. It is shown in Di Ruscio (1996a) that an extra projection is needed in order for the **N4SID** algorithm to give consistent estimates in the case of colored inputs. A robustified algorithm is presented in Van Overschee (1995). However, as we understand it, the **ROBUST** algorithm estimate the poles similarly as the **MOESP** algorithm.

In the case of rich exiting input signals, e.g. such as many real world inputs and of course white noise inputs etc., the subspace methods **DSR**, **CVA** and **MOESP** computes pole estimates which is comparable with those from **PEM**, both with respect to bias and variance.

For poorly exciting input signals the variance of the subspace pole estimates is often larger than the variance of the **PEM** pole estimates. See e.g. Viberg (1995).

It is therefore important to clarify the reason for the sensitivity of the estimates from subspace methods in the case of poorly exciting input signals. The question of how we can influence upon and reduce the variance of the subspace parameter estimates is important.

In order to improve the estimates in case of poorly exiting input signals we suggest to investigate the following ideas.

1. PCR, PLS or some other form of regularization for computing the projections and pseudo-inverse matrices.
2. Using regularization when computing the pseudo-inverse matrices.
3. Scaling

We will in the next section show how the Partial Least Squares (PLS) method can be used to compute the projection matrices.

7.6 PLS for computing the projections

The LQ decomposition is used in the SID algorithms by Verhagen (1994), Van Overschee and De Moor (1994) and Di Ruscio (1995), (1997).

In the following a new method for computing the projection matrices which is based on the Partial Least Squares (PLS) method is presented. Based on numerical examples, we have found that this method may reduce the variability of the pole estimates in case of poorly exciting input signals. The reason for this is that the projection matrices are computed without computing the pseudo-inverse of badly conditioned matrices. This results is to our knowledge new.

For simplicity in the following, define

$$\tilde{Y} \overset{\text{def}}{=} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \in \mathbb{R}^{p \times K} \tag{7.23}$$

where the number of rows in $\tilde{Y}$ are $p = (L+1+J)r + Jm$.

A 1st PLS decomposition is computed as follows. The arguments to the PLS algorithm are the data matrix of future outputs $Y_{J|L+1}$ and the data matrix with past inputs and outputs and future inputs $\tilde{Y}$. The PLS algorithm is iterative. The number of iterations is equal to the number of components a_1 which is bounded by $1 \leq a_1 \leq p$. Normally, a_1 can be chosen as the effective rank of $\tilde{Y}$. The problem of choosing the number of components should be discussed in some details.

We have the following decompositions

$$\tilde{Y} = PT + G \tag{7.24}$$

$$Y_{J|L+1} = CT + F \tag{7.25}$$

where $T \in \mathbb{R}^{a_1 \times K}$ is an orthonormal matrix, i.e., $TT^T = I_{a_1}$, $P \in \mathbb{R}^{p \times a_1}$ is a loading matrix for $\tilde{Y}$, $C \in \mathbb{R}^{(L+1)m \times a_1}$ is a loading matrix for $Y_{J|L+1}$. $G \in \mathbb{R}^{p \times K}$ and $F \in \mathbb{R}^{(L+1)m \times K}$ are residual matrices.

It can be shown that the projection of the row space of $Y_{J|L+1}$ onto the row space of $\tilde{Y}$ is equal to

$$Z^d_{J|L+1} \overset{\text{def}}{=} Y_{J|L+1}/\tilde{Y} = CT \tag{7.26}$$

We now have to remove the effect of future outputs from $Z^d_{J|L+1}$ in order to recover the extended observability matrix of the system. From (7.24) we have that

$$U_{J|L+1} = P_1 T + G_1 \tag{7.27}$$

where $P_1 \in \mathbb{R}^{(L+1)r \times a_1}$ is equal to the first $(L+1)r$ rows in matrix P, similarly $G_1 \in \mathbb{R}^{(L+1)r \times K}$ is equal to the first $(L+1)r$ rows in the residual matrix G.

The projection of the row space of matrix $Z^d_{J|L+1}$ onto the orthogonal complement of the row space of $U_{J|L+1}$ can then be computed as

$$Z_{J|L+1} \overset{\text{def}}{=} Z^d_{J|L+1} U^{\perp}_{J|L+1} = C - CP_1^T (P_1 P_1^T)^{\dagger} P_1 \tag{7.28}$$

However, in case of a badly conditioned matrix $P_1 P_1^T$ a second PLS can be computed with matrices C and P_1 as arguments. We have the following decompositions

$$P_1 = P_2 T_2 + G_2 \tag{7.29}$$

$$C = C_2 T_2 + F_2 \tag{7.30}$$

where $T_2 \in \mathbb{R}^{a_2 \times a_1}$ is orthonormal. $P_2 \in \mathbb{R}^{(L+1)r \times a_2}$ is a loading matrix for P_1, $C_2 \in \mathbb{R}^{(L+1)m \times a_2}$ is a loading matrix for C. $G_2 \in \mathbb{R}^{(L+1)r \times a_1}$ and $F_2 \in \mathbb{R}^{(L+1)m \times a_1}$ are residual matrices. From this we have the alternative to (7.28)

$$Z_{J|L+1} \overset{\text{def}}{=} Z^d_{J|L+1} U^{\perp}_{J|L+1} = C - C_2 T_2^T. \tag{7.31}$$

The number of components a_2 used to compute the 2nd PLS decomposition should be bounded by $1 \leq a_2 \leq a_1$.

Usually one can put $a_1 = \text{rank}(\tilde{Y})$ for the 1st PLS decomposition and $a_2 = \text{rank}(P_1)$ for the 2nd PLS decomposition. In practical situations it may be difficult to compute the (effective) rank.

One should note that the effect of choosing a_1 and a_2 smaller than the ranks of $\tilde{Y}$ and $U_{J|L+1}$, respectively, may be that the variance of the estimates is reduced. However, a bias may be included in the estimates. The problem of choosing a_1 and a_2 is in general a trade off between bias and variance. This is a form of regularization.

Monte Carlo simulations shows that the estimates when choosing $a_1 = p$ and $a_1 = \text{rank}(\tilde{Y})$, i.e., choosing the number of components in the 1st PLS equal to the row size of $\tilde{Y}$ or equal to the rank, and using (7.28) gives approximately the same results as using the LQ decomposition for computing the same projections. This will be illustrated in the next section. The effect of choosing $a_1 < \text{rank}(\tilde{Y})$ is that a bias in general is introduced in the estimates, however, the variance may be reduced.

7.7 Numerical example

The results from a Monte Carlo simulation of a MIMO system are illustrated in Figure (7.1). The number of components was chosen as $a_1 = \text{rank}(\tilde{Y})$ in the PLS decomposition used to compute the matrix $Z_{J|L+1}$. The number of samples was $N = 5000$. The input signal was chosen as

$$u_t = \begin{bmatrix} (\sin(\frac{t}{14}) + \sin(\frac{t}{7}) + \sin(\frac{t}{3}))k_1 \\ (\sin(\frac{t}{25}) + \sin(\frac{t}{10}) + \sin(\frac{t}{5}) + \sin(t))k_2 \end{bmatrix} \tag{7.32}$$

where k_1 and k_2 are positive scalar parameters. Our experience is that in case of porely exiting input signals the parameter L in the **DSR** algorithm (which is the number of block rows in the extended observability matrix and which can be interpreted as the horizon used to predict the number of states) should be chosen as small as possible in order to reduce the variance of the estimates. For the **DSR** method we have the condition $n \leq Lm$. For this example $n = 3$ and $m = 2$. Hence, the smallest possible parameter is $L = 2$. The results from the **DSR** methods obtained from using both the LQ and PLS approaches are illustrated in Figure 7.1.

The results from the **ROBUST** method by Van Overschee (1995) is also presented in Figure 7.1. The parameter for the number of block rows in this method is I. The smallest possible parameter was found to be $I = 3$ which was used in the simulations.

The results from the prediction error method PEM, Ljung (1991), are also presented in Figure 7.1.

This example shows a larger variability in the complex pole estimates from **ROBUST** than the corresponding pole estimates from **DSR**. The estimates from **DSR** is close to the estimates from **PEM**. An open question is: can the variance of the pole estimates from the SID algorithms be further reduced ?

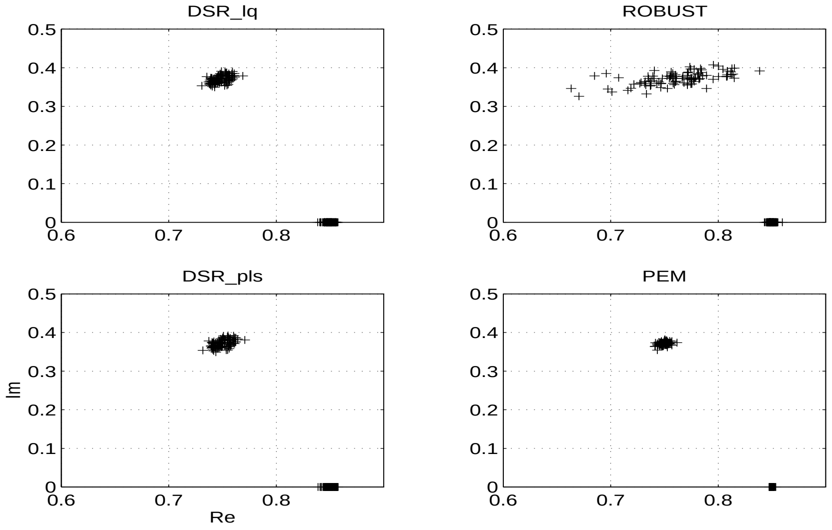


Figure 7.1: Pole estimates of a system with two inputs, two outputs and three states. **DSR** parameters $L = J = 2$ for both the LQ and PLS method for computing the projection matrices. **ROBUST** parameter $I = 3$. **PEM** stands for the prediction error method in the System Identification Toolbox for MATLAB.

7.8 Conclusion

It is shown that the partial least squares method is an alternative to the standard LQ decomposition for computing the range space of the extended observability matrix.

7.9 References

Di Ruscio, D. (1997). A method for identification of combined deterministic stochastic systems. In: *Applications of Computer Aided Time Series Modeling*, Lecture Notes in Statistics 119, Eds. M. Aoki and A. M. Havenner, Springer Verlag, ISBN 0-387-94751-5.

Di Ruscio, D. and A. Holmberg (1996). Subspace identification for dynamic process analysis and modeling. *Control Systems 96*, Halifax, Nova Scotia, May 1996.

Golub, G. H. and C. F. Van Loan (1983). *Matrix Computations.* North Oxford Academic Publishers Ltd.

Larimore, W. E. (1983). System identification, reduced order filtering and modeling via canonical variate analysis. *Proc. of the American Control Conference*, San Francisco, USA, pp. 445-451.

Larimore, W. E. (1990). Canonical Variate Analysis in Identification, Filtering and Adaptive Control. *Proc. of the 29th Conference on Decision and Control*, Honolulu, Hawaii, December 1990, pp. 596-604.

Ljung, L. (1991). *System Identification Toolbox.* The Mathworks, Inc.

Faurre, P. L. (1976). Stochastic realization algorithms. In: *System Identification: Advances and Case Studies*, (eds. R. K. Mehra and D. G. Lainiotis), Academic Press.

Kung, S. Y. (1978). A new identification and Model Reduction Algorithm via Singular Value Decomposition. *Conf. on Circuits, Systems and Computers*, Pacific Grove, CA, November 1978, pp. 705-714.

Moore, B. C. (1981). Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction. *IEEE Trans. on Automatic Control*, Vol. AC-26, pp. 17-31.

Van Overschee, P. and B. De Moor (1994). N4SID: Subspace Algorithms for the Identification of Combined Deterministic Stochastic Systems. *Automatica*, vol. 30, No. 1, pp.75-94.

Van Overschee, P. (1995). *Subspace Identification: theory-implementation-application.* PhD thesis, Katholieke Universiteit Leuven, Belgium.

Van Overschee, P. and B. De Moor (1995). A Unifying Theorem for Three Subspace System Identification Algorithms. *Automatica*, vol. 31, No. 12, pp. 1853-1864.

Verhagen, M. (1994). Identification of the deterministic part of MIMO state space models given on innovations form from input output data. *Automatica*, vol. 30, No. 1, pp. 61-74.

Viberg, M. (1995). Subspace-Based Methods for the Identification of Linear Time-invariant Systems. *Automatica*, vol. 31, No. 12, pp. 1835-1851.

Wold, H. (1966). Non-linear estimation by iterative least squares procedures. *Researsch papers in statistics*, Ed. David, F. Wiley, New York, pp. 411-444.

7.10 Appendix: proofs

7.10.1 Proof of Equation (7.4)

We have

$$Y_{J|L+1} = O_{L+1}X_J + H^d_{L+1}U_{J|L+1} + H^s_{L+1}E_{J|L+1} \tag{7.33}$$

The relationship between the "future" sequence of states, X_J, and the "past" sequence of states, X_0, is given by

$$X_J = A^J X_0 + C^d_J U_{0|J} + C^s_J E_{0|J} \tag{7.34}$$

Substituting Equation (7.34) into equation (7.33) in order to eliminate X_J gives

$$Y_{J|L+1} = O_{L+1}(A^J X_0 + C^d_J U_{0|J} + C^s_J E_{0|J}) + H^d_{L+1}U_{J|L+1} + H^s_{L+1}E_{J|L+1} \tag{7.35}$$

An expression for the sequence of past states, X_0, is determined from

$$Y_{0|J} = O_J X_0 + H^d_J U_{0|J} + H^s_J E_{0|J} \tag{7.36}$$

Equation (7.36) gives

$$X_0 = O^\dagger_J(Y_{0|J} - H^d_J U_{0|J} - H^s_J E_{0|J}) \tag{7.37}$$

Substituting Equation (7.37) into Equation (7.35) in order to eliminate X_0 gives

$$\begin{aligned} Y_{J|L+1} &= O_{L+1}A^J O^\dagger_J(Y_{0|J} - H^d_J U_{0|J} - H^s_J E_{0|J}) + O_{L+1}C^d_J U_{0|J} + O_{L+1}C^s_J E_{0|J} \\ &+ H^d_{L+1}U_{J|L+1} + H^s_{L+1}E_{J|L+1} \end{aligned} \tag{7.38}$$

which can be written as

$$\begin{aligned} Y_{J|L+1} &= O_{L+1}A^J O^\dagger_J Y_{0|J} + (O_{L+1}C^d_J - O_{L+1}A^J O^\dagger_J H^d_J)U_{0|J} \\ &+ (O_{L+1}C^s_J - O_{L+1}A^J O^\dagger_J H^s_J)E_{0|J} \\ &+ H^d_{L+1}U_{J|L+1} + H^s_{L+1}E_{J|L+1} \end{aligned} \tag{7.39}$$

which is identical to Equation (7.4), Theorem 7.3.1. **QED**.

7.10.2 Proof I of Equation (7.10)

Projection I: (remove effect of future noise, $E_{J|L+1}$)
From Equation (7.4) we have

$$Y_{J|L+1}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \begin{bmatrix} H^d_{L+1} & P^d_{L+1} & \tilde{A}^J_{L+1} \end{bmatrix} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$$

$$+P^s_{L+1}E_{0|J}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} + dE_1 \tag{7.40}$$

where

$$dE_1 \overset{\text{def}}{=} H^s_{L+1}E_{J|L+1}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{7.41}$$

and

$$\lim_{K\to\infty} dE_1 = H^s_{L+1}(\lim_{K\to\infty} E_{J|L+1}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) = 0_{(L+1)m\times K} \tag{7.42}$$

Rearranging the terms on the right hand side in Equation (7.40) gives

$$Y_{J|L+1}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} =$$

$$\begin{bmatrix} P^d_{L+1} & \tilde{A}^J_{L+1} & P^s_{L+1} \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \\ E_{0|J}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} + H^d_{L+1}U_{J|L+1} + dE_1 \tag{7.43}$$

Projection II: (remove effect of future inputs, $U_{J|L+1}$)

$$(Y_{J|L+1}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix})U^{\perp}_{J|L+1} =$$

$$\begin{bmatrix} P^d_{L+1} & \tilde{A}_{L+1} & P^s_{L+1} \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \\ E_{0|J}/\begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \end{bmatrix} U^{\perp}_{J|L+1} + dE_2 \tag{7.44}$$

where

$$dE_2 \overset{\text{def}}{=} dE_1 U^{\perp}_{J|L+1} \tag{7.45}$$

Equation (7.44) is equal to Equation (7.10) in Theorem 7.4.1. Note that $\lim_{K\to\infty} dE_2 = 0_{(L+1)m\times K}$. **QED**.

7.10.3 Proof II of Equation (7.10)

From Theorem 7.3.1, Equation (7.4) we have

$$Y_{J|L+1} = \left[\begin{array}{ccc} H^d_{L+1} & P^d_{L+1} & \tilde{A}^J_{L+1} \end{array} \right] \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] + P^s_{L+1} E_{0|J} + H^s_{L+1} E_{J|L+1} \tag{7.46}$$

The terms defined by the future inputs and future noise, $H^d_{L+1} U_{J|L+1}$ and $H^s_{L+1} E_{J|L+1}$, respectively, does not have the same column space as the extended observability matrix. These terms has to be removed. We have

$$Y_{J|L+1} \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right]^{\perp} - P^s_{L+1} E_{0|J} \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right]^{\perp} = H^s_{L+1} E_{J|L+1} + dE_1 \tag{7.47}$$

We have assumed that the noise is zero mean and white. Future noise is therefore uncorrelated with past noise and past and future inputs. Hence we have $\lim_{K\to\infty} dE_1 = 0_{(L+1)m \times K}$.

Substituting (7.47) into (7.46) in order to eliminate the term $H^s_{L+1} E_{J|L+1}$ gives

$$Y_{J|L+1} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] =$$

$$\left[\begin{array}{ccc} P^d_{L+1} & \tilde{A}^J_{L+1} & P^s_{L+1} \end{array} \right] \left[\begin{array}{c} U_{0|J} \\ Y_{0|J} \\ E_{0|J} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] \end{array} \right] + H^d_{L+1} U_{J|L+1} + dE_1 \tag{7.48}$$

Post-multiplying with $U^{\perp}_{J|L+1}$ gives

$$Z_{J|L+1} \overset{\text{def}}{=} Y_{J|L+1} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] U^{\perp}_{J|L+1} = O_{L+1} X_a + dE_1 U^{\perp}_{J|L+1} \tag{7.49}$$

where

$$X_a \overset{\text{def}}{=} \left[\begin{array}{ccc} A^J O^{\dagger}_J & C^d_J - A^J O^{\dagger}_J H^d_J & C^s_J - A^J O^{\dagger}_J H^s_J \end{array} \right] \left[\begin{array}{c} Y_{0|J} \\ U_{0|J} \\ E_{0|J} / \left[\begin{array}{c} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] \end{array} \right] U^{\perp}_{J|L+1}$$

Hence, $Z_{J|L+1}$ has the same column space as the extended observability matrix.

7.10.4 Proof of Equation (7.22)

For simplicity define

$$\tilde{Y} \stackrel{\text{def}}{=} \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \begin{bmatrix} U_{J|L+1} \\ W_1 \end{bmatrix} \tag{7.50}$$

Theorem 7.4.1, Equation (7.10), yields

$$Z_{J|L+1} = (Y_{J|L+1}/\tilde{Y})U^{\perp}_{J|L+1} = Y_{J|L+1}\tilde{Y}^T(\tilde{Y}\tilde{Y}^T)^{-1}\tilde{Y}U^{\perp}_{J|L+1} \tag{7.51}$$

Notice that

$$\tilde{Y}U^{\perp}_{J|L+1} = \begin{bmatrix} U_{J|L+1} \\ W_1 \end{bmatrix} U^{\perp}_{J|L+1} = \begin{bmatrix} 0_{(L+1)r\times K} \\ W_1 U^{\perp}_{J|L+1} \end{bmatrix} \tag{7.52}$$

The inverse can be written

$$(\tilde{Y}\tilde{Y}^T)^{-1} = \begin{bmatrix} U_{J|L+1}U^T_{J|L+1} & U_{J|L+1}W_1^T \\ W_1 U^T_{J|L+1} & W_1 W_1^T \end{bmatrix}^{-1} \stackrel{\text{def}}{=} \begin{bmatrix} I_{11} & I_{12} \\ I_{21} & \Delta^{-1} \end{bmatrix} \tag{7.53}$$

For simplicity define $U_f = U_{JL+1}$. We have

$$\begin{aligned} I_{12} &= -(U_f U_f^T)^{-1}U_f W_1^T \Delta^{-1} && (7.54) \\ \Delta &= W_1 W_1^T - W_1 U_f^T (U_f U_f^T)^{-1} U_f W_1^T = W_1 U_f^{\perp} W_1^T && (7.55) \end{aligned}$$

where we have used index f for future. Δ is the Schur complement of $U_f U_f^T$. See e.g. Kailath (1980). We now have

$$\begin{aligned} Z_{J|L+1} &= \begin{bmatrix} Y_f U_f^T & Y_f W_1^T \end{bmatrix} \begin{bmatrix} I_{11} & I_{12} \\ I_{21} & \Delta^{-1} \end{bmatrix} \begin{bmatrix} 0_{(L+1)r\times K} \\ W_1 U^{\perp}_{J|L+1} \end{bmatrix} \\ &= \begin{bmatrix} Y_f U_f^T & Y_f W_1^T \end{bmatrix} \begin{bmatrix} I_{12} \\ \Delta^{-1} \end{bmatrix} W_1 U^{\perp}_{J|L+1} \\ &= (Y_f W_1^T \Delta^{-1} + Y_f U_f^T I_{12}) W_1 U^{\perp}_{J|L+1} \\ &= (Y_f - Y_f U_f^T (U_f U_f^T)^{-1} U_f) W_1^T \Delta^{-1} W_1 U^{\perp}_{J|L+1} \\ &= Y_f U^{\perp}_{J|L+1} W_1^T \Delta^{-1} W_1 U^{\perp}_{J|L+1} && (7.56) \end{aligned}$$

7.10.5 Including constraints: regularization

From Equation 7.4 we have

$$\begin{aligned} Y_f\tilde{Y}^T &= M\tilde{Y}\tilde{Y}^T + P^s_{L+1}E_p\tilde{Y}^T + H^s_{L+1}E_f\tilde{Y}^T \\ &= M(\tilde{Y}\tilde{Y}^T+\Lambda) + P^s_{L+1}E_p\tilde{Y}^T + H^s_{L+1}E_f\tilde{Y}^T - M\Lambda \qquad (7.57) \end{aligned}$$

and

$$Y_f\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1} = M + P^s_{L+1}E_p\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1} + (H^s_{L+1}E_f\tilde{Y}^T - M\Lambda)(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1} \quad (7.58)$$

and

$$Y_f\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y} = M\tilde{Y} + P^s_{L+1}E_p\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y} + dE_1 \quad (7.59)$$

where

$$dE_1 = (H^s_{L+1}E_f\tilde{Y}^T - M\Lambda)(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y} \quad (7.60)$$

Rearranging Equation (7.59) gives

$$Y_f\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y} = \\ \begin{bmatrix} P^d_{L+1} & \tilde{A}^J_{L+1} & P^s_{L+1} \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \\ E_{0|J}\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y} \end{bmatrix} + H^d_{L+1}U_f + dE_1 \quad (7.61)$$

and finally

$$Y_f\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y}U_f^\perp = \\ \begin{bmatrix} P^d_{L+1} & \tilde{A}^J_{L+1} & P^s_{L+1} \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \\ E_{0|J}\tilde{Y}^T(\tilde{Y}\tilde{Y}^T+\Lambda)^{-1}\tilde{Y} \end{bmatrix} U_f^\perp + dE_2 \quad (7.62)$$

where

$$\begin{aligned} U_f^\perp &= I - U_f^T(U_fU_f^T+\Lambda_2)^{-1}U_f \qquad (7.63) \\ dE_2 &= dE_1U_f^\perp + \text{error term} \qquad (7.64) \end{aligned}$$

A non-zero matrix Λ will in general include a bias in the estimates and also affect the variance. Consider the following block-diagonal matrix

$$\Lambda = \begin{bmatrix} \Lambda_{11} & 0 & 0 \\ 0 & \Lambda_{22} & 0 \\ 0 & 0 & \Lambda_{33} \end{bmatrix} \quad (7.65)$$

The term $M\Lambda$ in the error equation 7.60 is then

$$M\Lambda = H^d_{L+1}\Lambda_{11} + P^s_{L+1}\Lambda_{22} + \tilde{A}^J_{L+1}\Lambda_{33} \quad (7.66)$$

Notice that $\Lambda_{11} = 0_{(L+1)r\times(L+1)r}$ gives consistent estimates.

7.10.6 Some useful results

Lemma 7.10.1 *The following equality is true*

$$U_f / \begin{bmatrix} U_f \\ W_1 \end{bmatrix} = U_f \tag{7.67}$$

Lemma 7.10.2 *The following equality is true*

$$U_f \begin{bmatrix} U_f \\ W_1 \end{bmatrix}^{\perp} = 0 \tag{7.68}$$

Proof of Lemma 7.10.1:

From the definition of A/B, Equation (7.1), we have

$$U_f / \begin{bmatrix} U_f \\ W_1 \end{bmatrix} = U_f \begin{bmatrix} U_f^T & W_1^T \end{bmatrix} \begin{bmatrix} U_f U_f^T & U_f W_1^T \\ W_1 U_f^T & W_1 W_1^T \end{bmatrix}^{-1} \begin{bmatrix} U_f \\ W_1 \end{bmatrix} \tag{7.69}$$

The inverse of a partitioned matrix is (Kailath (1980))

$$\begin{bmatrix} U_f U_f^T & U_f W_1^T \\ W_1 U_f^T & W_1 W_1^T \end{bmatrix}^{-1} =$$
$$\begin{bmatrix} (U_f U_f^T)^{-1} + (U_f U_f^T)^{-1} U_f W_1^T \Delta^{-1} W_1 U_f^T (U_f U_f^T)^{-1} & -(U_f U_f^T)^{-1} U_f W_1^T \Delta^{-1} \\ -\Delta^{-1} W_1 U_f^T (U_f U_f^T)^{-1} & \Delta^{-1} \end{bmatrix} \tag{7.70}$$

where Δ is the Schur complement of $U_f U_f^T$. See A.22. Using the expression for the inverse, Equation (7.70), then we have that

$$\begin{bmatrix} U_f U_f^T & U_f W_1^T \end{bmatrix} \begin{bmatrix} U_f U_f^T & U_f W_1^T \\ W_1 U_f^T & W_1 W_1^T \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \end{bmatrix} \tag{7.71}$$

Substituting (7.71) into (7.67) gives

$$U_f / \begin{bmatrix} U_f \\ W_1 \end{bmatrix} = \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} U_f \\ W_1 \end{bmatrix} = U_f \tag{7.72}$$

and Lemma 7.10.1 is proved. **QED**.

Proof of Lemma 7.10.2:

From the definition of $AB^{\perp}$, Equation (7.2), we have

$$U_f \begin{bmatrix} U_f \\ W_1 \end{bmatrix}^{\perp} = U_f - U_f / \begin{bmatrix} U_f \\ W_1 \end{bmatrix} = 0 \tag{7.73}$$

where we have used Lemma 7.10.1 with proof. **QED**.

7.10.7 Proof III of Equation (7.10)

The ESSM constructed from future data is

$$Y_{J+1|L} = \tilde{A}_L Y_{J|L} + \tilde{B}_L U_{J|L+1} + \tilde{C}_L E_{J|L+1} \tag{7.74}$$

Using Lemma 7.10.1 we get

$$Z^d_{J+1|L} = \tilde{A}_L Z^d_{J|L} + \tilde{B}_L U_{J|L+1} + \tilde{C}_L dE_{J|L+1} \tag{7.75}$$

where

$$Z^d_{J+1|L} \overset{\text{def}}{=} Y_{J+1|L} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{7.76}$$

$$Z^d_{J|L} \overset{\text{def}}{=} Y_{J|L} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{7.77}$$

$$dE_{J|L+1} \overset{\text{def}}{=} E_{J|L+1} / \begin{bmatrix} U_{J|L+1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{7.78}$$

Due to assumptions (ergodic noise process) the error term $dE_{J|L+1}$ tends to zero w.p.1 as the number of columns K tends to infinity.

Hence,

$$(A, B, D, E) = \arg \min_{A,B,D,E} \parallel Z^d_{J+1|L} - \tilde{A} Z^d_{J|L} - \tilde{B} U_{J|L+1} \parallel_F \tag{7.79}$$

7.10.8 Prediction of future outputs

We have the following relationship

$$Y_{J|L+1} = Z^d_{J|L+1} + Z^s_{J|L+1} \tag{7.80}$$

From Theorem 4.3, Di Ruscio (1997), we have that the innovations can be estimated from

$$F E_{J+L|1} = Z^s_{J+L|1} (Z^s_{J|L})^{\perp} \tag{7.81}$$

from the above we have that a predictor for the future outputs is given by

$$\hat{y}_{J+L|1} = Z^d_{J+L|1} (Z^s_{J|L})^{\perp} \tag{7.82}$$

This indicates that

$$y_{J+L|1} = Y_{J+L|1} (Z^s_{J|L})^{\perp} \tag{7.83}$$

The predictor can simply be written

$$\hat{y}_{J+L|1} = Z^d_{J+L|1} \tag{7.84}$$

Chapter 8

Subspace identification of the states and the Kalman filter gain

8.1 Introduction

A complete subspace identification (SID) algorithm are discussed and derived in this paper. The derivation presented is different from the other published papers on subspace identification, Van Overschee and De Moor (1994), Larimore (1990), Viberg (1995) and Van Overschee (1995) and the references therein, because we are using general input and output matrix equations which describes the relationship between the past and the future input and output data matrices.

One of the contributions in this paper is that it is shown that the Kalman filter model matrices, including the Kalman gain and the noise innovations process, of a combined deterministic and stochastic system can be identified directly from certain projection matrices which are computed from the known input and output data, without solving any Riccati or Lyapunov matrix equations. This subspace method and results was presented without proof in Di Ruscio (1995) and Di Ruscio (1997). One contribution in this paper is a complete derivation with proof. A new method for computing the matrices in the deterministic part of the system is presented. This method has been used in the DSR Toolbox for Matlab, Di Ruscio (1996), but has not been published earlier.

Furthermore, it is pointed out that the states, in general (i.e. for colored input signals), only can be computed if the complete deterministic part of the model is known or identified first. This is probably the reason for which the state based subspace algorithms which are presented in the literature does not work properly for colored input signals. The SID algorithm in Verhagen (1994) works for colored input signals. The stochastic part of the model is not computed by this algorithm. The N4SID algorithm in Van Overschee and De Moor (1994) works well and only for white input signals. The stochastic part of the model is computed by solving a Riccati equation. However, the robust modification in Van Overschee and De Moor (1995) works well also for colored input signals.

The rest of this paper is organized as follows. Some basic matrix definitions and notations are presented in Section 8.2. The problem of subspace identification of the states for both colored and white input signals is discussed in Section 8.3.1. The subspace identification of the extended observability matrix, which possibly is the most important step in any SID algorithm, are discussed in Section 8.3.2. It is proved that the Kalman filter gain matrix and the noise innovations process can be identified directly from the data in Section 8.3.3. A least squares optimal method for computing the deterministic part of the combined deterministic and stochastic system is presented in Section 8.3.4.

The problem of using subspace methods for closed loop systems are pointed out and some solutions to the problem are pointed out in section 8.4.

The main contribution in this paper is a new method for subspace system identification that works for closed loop as well as open loop systems. The method are based on the theory in Section 3 and is presented in Section 8.5. This method is probably one of the best for closed loop subspace system identification.

Some topics and remarks related to the algorithm are presented in Section 8.6. Numerical examples are provided in Section 8.7 in order to illustrate the behaviour of the algorithm both in open and closed loop. Some concluding remarks follows in Section 8.8.

8.2 Notation and definitions

8.2.1 System and matrix definitions

Consider the following state space model on innovations form

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k + Ce_k, \tag{8.1}$$

$$y_k = D\bar{x}_k + Eu_k + Fe_k, \tag{8.2}$$

where e_k is white noise with covariance matrix $\mathrm{E}(e_k e_k^T) = I_m$. One of the problems addressed and discussed in this paper is to directly identify (subspace identification) the system order, n, the state vector $\bar{x}_k \in \mathbb{R}^n$, and the matrices (A, B, C, D, E, F) from a sequence of known input and output data vectors, u_k, $\in \mathbb{R}^r$ and y_k, $\in \mathbb{R}^m$, respectively. A structure parameter, g, is introduced so that $g = 1$ when E is to be identified and $g = 0$ when E is a-priori known to be zero. This should be extended to a structure matrix G with ones and zeroes, the ones pointing to the elements in E which are to be estimated. This is not considered further here. Based on (8.1) and (8.2) we make the following definitions for further use:

Definition 8.1 (Basic matrix definitions)
The extended observability matrix, O_i, for the pair (D, A) is defined as

$$O_i \stackrel{\text{def}}{=} \begin{bmatrix} D \\ DA \\ \vdots \\ DA^{i-1} \end{bmatrix} \in \mathbb{R}^{im \times n}, \tag{8.3}$$

where the subscript i denotes the number of block rows.

The reversed extended controllability matrix, C_i^d, for the pair (A, B) is defined as

$$C_i^d \stackrel{\text{def}}{=} \begin{bmatrix} A^{i-1}B & A^{i-2}B & \cdots & B \end{bmatrix} \in \mathbb{R}^{n \times ir}, \tag{8.4}$$

where the subscript i denotes the number of block columns. A reversed extended controllability matrix, C_i^s, for the pair (A, C) is defined similar to (8.4), i.e.,

$$C_i^s \stackrel{\text{def}}{=} \begin{bmatrix} A^{i-1}C & A^{i-2}C & \cdots & C \end{bmatrix} \in \mathbb{R}^{n \times im}, \tag{8.5}$$

i.e., with B substituted with C in (8.4). The lower block triangular Toeplitz matrix, H_i^d, for the quadruple matrices (D, A, B, E)

$$H_i^d \stackrel{\text{def}}{=} \begin{bmatrix} E & 0_{m\times r} & 0_{m\times r} & \cdots & 0_{m\times r} \\ DB & E & 0_{m\times r} & \cdots & 0_{m\times r} \\ DAB & DB & E & \cdots & 0_{m\times r} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}B & DA^{i-3}B & DA^{i-4}B & \cdots & E \end{bmatrix} \in \mathbb{R}^{im \times (i+g-1)r}, \tag{8.6}$$

where the subscript i denotes the number of block rows and $i + g - 1$ is the number of block columns. Where $0_{m\times r}$ denotes the $m \times r$ matrix with zeroes. A lower block triangular Toeplitz matrix H_i^s for the quadruple (D, A, C, F) is defined as

$$H_i^s \stackrel{\text{def}}{=} \begin{bmatrix} F & 0_{m\times m} & 0_{m\times m} & \cdots & 0_{m\times m} \\ DC & F & 0_{m\times m} & \cdots & 0_{m\times m} \\ DAC & DC & F & \cdots & 0_{m\times m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ DA^{i-2}C & DA^{i-3}C & DA^{i-4}C & \cdots & F \end{bmatrix} \in \mathbb{R}^{im \times im}. \tag{8.7}$$

8.2.2 Hankel matrix notation

Hankel matrices are frequently used in realization theory and subspace system identification. The special structure of a Hankel matrix as well as some matching notations, which are frequently used througout, are defined in the following.

Definition 8.2 (Hankel matrix) *Given a (vector or matrix) sequence of data*

$$s_t \in \mathbb{R}^{nr \times ns} \; \forall \; t = 0, 1, 2, \ldots, t_0, t_0 + 1, \ldots, \tag{8.8}$$

where nr is the number of rows in s_t and nc is the number of columns in s_t.

Define integer numbers t_0, L and K and define the matrix S_t as follows

$$S_{t_0|L} \overset{def}{=} \begin{bmatrix} s_{t_0} & s_{t_0+1} & s_{t_0+2} & \cdots & s_{t_0+K-1} \\ s_{t_0+1} & s_{t_0+2} & s_{t_0+3} & \cdots & s_{t_0+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{t_0+L-1} & s_{t_0+L} & s_{t_0+L+1} & \cdots & s_{t_0+L+K-2} \end{bmatrix} \in \mathbb{R}^{Lnr \times Knc}. \tag{8.9}$$

which is defined as a Hankel matrix because of the special structure. The integer numbers t_0, L and K are defined as follows:

- *t_0 start index or initial time in the sequence, s_{t_0}, which is the upper left block in the Hankel matrix.*
- *L is the number of nr-block rows in $S_{t_0|L}$.*
- *K is the number of nc-block columns in $S_{t_0|L}$.*

A Hankel matrix is symmetric and the elements are constant across the anti-diagonals. We are usually working with vector sequences in subspace system identification, i.e., s_t is a vector in this case and hence, $nc = 1$. Examples of such vector processes, to be used in the above Hankel-matrix definition, are the measured process outputs, $y_t \in \mathbb{R}^m$, and possibly known inputs, $u_t \in \mathbb{R}^r$. Also define

$$y_{j|i} \overset{\text{def}}{=} \begin{bmatrix} y_j^T & y_{j+1}^T & \cdots & y_{j+i-1}^T \end{bmatrix}^T \in \mathbb{R}^{im}, \tag{8.10}$$

which is refereed to as an extended (output) vector, for later use.

8.2.3 Projections

Given two matrices $A \in \mathbb{R}^{i \times k}$ and $B \in \mathbb{R}^{j \times k}$. The orthogonal projection of the row space of A onto the row space of B is defined as

$$A/B = AB^T(BB^T)^\dagger B. \tag{8.11}$$

The orthogonal projection of the row space of A onto the orthogonal complement of the row space of B is defined as

$$AB^\perp = A - A/B = A - AB^T(BB^T)^\dagger B. \tag{8.12}$$

The following properties are frequently used

$$A / \begin{bmatrix} A \\ B \end{bmatrix} = A, \tag{8.13}$$

$$A/\begin{bmatrix} A \\ B \end{bmatrix}^{\perp} = 0. \tag{8.14}$$

Prof of (8.13) and (8.14) can be found in e.g., Di Ruscio (1997b). The Moore-Penrose pseudo-inverse of a matrix $A \in \mathbb{R}^{i \times k}$ where $k > i$ is defined as $A^{\dagger} = A^T(AA^T)^{-1}$. Furthermore, consistent with (8.12) we will use the definition

$$B^{\perp} = I_k - B^T(BB^T)^{\dagger}B, \tag{8.15}$$

throughout the paper. Note also the properties that $(B^{\perp})^T = B^{\perp}$ and $B^{\perp}B^{\perp} = B^{\perp}$.

8.3 Subspace system identification

8.3.1 Subspace identification of the states

Consider a discrete time Kalman filter on innovations form, i.e.,

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k + K\varepsilon_k, \tag{8.16}$$

$$y_k = D\bar{x}_k + Eu_k + \varepsilon_k, \tag{8.17}$$

where $\bar{x}_k \in \mathbb{R}^n$ is the predicted state in a minimum variance sense, $\varepsilon_k \in \mathbb{R}^m$ is the innovations at discrete time k, i.e., the part of $y_k \in \mathbb{R}^m$ that cannot be predicted from past data (i.e. known past inputs and outputs) and the present input. Furthermore, $\bar{y}_k = D\bar{x}_k + Eu_k$ is the prediction of y_k, and ε_k is white noise with covariance matrix $\Delta = \mathrm{E}(\varepsilon_k \varepsilon_k^T)$. Here $\varepsilon_k = Fe_k$ is the innovations and the model (8.1) and (8.2) is therefore equivalent with the Kalman filter (8.16) and (8.17). Furthermore, we have that $K = CF^{-1}$ and $\Delta = \mathrm{E}(\varepsilon_k \varepsilon_k^T) = FF^T$, when F is non-singular, i.e., when the system is not deterministic and when the Kalman filter exists.

A well known belief is that the states is a function of the past. Let us have a lock at this statement. The predicted state at time $k := t_0 + J$, i.e. $\bar{x}_{t_0+J}$ of a Kalman filter with the initial predicted state at $k := t_0$, i.e. $\bar{x}_{t_0}$ given, can be expressed as

$$\bar{x}_{t_0+J} = \tilde{C}_J^s y_{t_0|J} + \tilde{C}_J^d u_{t_0|J} + (A - KD)^J \bar{x}_{t_0}, \tag{8.18}$$

where $\tilde{C}_J^s = C_J(A - KD, K)$ is the reversed extended controllability matrix of the pair $(A - KD, K)$, $\tilde{C}_J^d = C_J(A - KD, B - KE)$ is the reversed extended controllability matrix of the pair $(A - KD, B - KE)$ and $\bar{x}_{t_0}$ is the initial predicted state (estimate) at the initial discrete time t_0. See (8.5) for the definition of the reversed controllability matrix. J is the *past horizon*, i.e., the number of past outputs and inputs used to define the predicted state (estimate) $\bar{x}_{t_0+J}$ at the discrete time $t_0 + J$.

Using (8.18) for different t_0, i.e. for t_0, $t_0 + 1$, $t_0 + 2$, …, $t_0 + K - 1$, gives the matrix equation

$$X_{t_0+J} = \tilde{C}_J^s Y_{t_0|J} + \tilde{C}_J^d U_{t_0|J} + (A - KD)^J X_{t_0}, \tag{8.19}$$

where

$$\begin{aligned} X_{t_0+J} &= \left[\begin{array}{cccc} \bar{x}_{t_0+J} & \bar{x}_{t_0+J+1} & \cdots & \bar{x}_{t_0+J+K-1} \end{array} \right] \in \mathbb{R}^{n \times K}, \quad (8.20)\\ X_{t_0} &= \left[\begin{array}{cccc} \bar{x}_{t_0} & \bar{x}_{t_0+1} & \cdots & \bar{x}_{t_0+K-1} \end{array} \right] \in \mathbb{R}^{n \times K}. \quad (8.21) \end{aligned}$$

where K is the number of columns in the data matrices. Note that K also is equal to the number of vector equations of the form (8.18) which is used to form the matrix version (8.19). Note also that the state matrix X_{t_0} can be eliminated from (8.19) by using the relationship

$$Y_{t_0|J} = O_J X_{t_0} + H_J^d U_{t_0|J+g-1} + H_J^s E_{t_0|J}, \quad (8.22)$$

which we have deduced from the innovations form, state space model (8.1) and (8.2). Putting $t_0 =: t_0 + J$ in (8.22) gives

$$Y_{t_0+J|L} = O_L X_{t_0+J} + H_L^d U_{t_0+J|L+g-1} + H_L^s E_{t_0+J|L}. \quad (8.23)$$

Using (8.19) to eliminate X_{t_0+J} in (8.23) gives a matrix equation which relates the future data matrices $Y_{t_0+J|L}$, $U_{t_0+J|L+g-1}$, $E_{t_0+J|L}$ and the past data matrices $Y_{t_0|J}$, $U_{t_0|J}$, $E_{t_0|J}$.

The data is usually defined at time instant (or number of observations) $k = 1, 2, \ldots, N$. Hence, $t_0 = 1$ in this case. However, we are often defining $t_0 = 0$ which corresponds to data defined at $k = 0, 1, \ldots, N-1$. The bar used to indicate predicted state is often omitted. Hence, for simplicity of notation, we define the following equations from (8.19), (8.22) and (8.23),

$$Y_{0|J} = O_J X_0 + H_J^d U_{0|J+g-1} + H_J^s E_{0|J}, \quad (8.24)$$

$$X_J = \tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J} + (A - KD)^J X_0, \quad (8.25)$$

$$Y_{J|L} = O_L X_J + H_L^d U_{J|L+g-1} + H_L^s E_{J|L}, \quad (8.26)$$

for lather use. Furthermore, (8.26) and (8.25) gives

$$Y_{J|L} = \left[\begin{array}{ccc} H_L^d & O_L \tilde{C}_J^d & O_L \tilde{C}_J^s \end{array} \right] \left[\begin{array}{c} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{array} \right] + O_L (A - KD)^J X_0 + H_L^s E_{J|L}. \quad (8.27)$$

Equation (8.27) is important for understanding a SID algorithm, because, it gives the relationship between the past and the future. Note also the terms in (8.27) which are "proportional" with the extended observability matrix O_L. From (8.27) we see that the effect from the future inputs, $U_{J|L+g-1}$, and the future noise, $E_{J|L}$, have to be removed from the future outputs, $Y_{J|L}$, in order to recover the subspace spanned by the extended observability matrix, O_L. A variation of this equation, in which the term X_0 is eliminated by using (8.22) or (8.24) is presented in Di Ruscio (1997b). Note also that (8.25) and (8.24) gives

$$X_J = \begin{bmatrix} P_J^u & P_J^y \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} - P_J^e E_{0|J}, \tag{8.28}$$

$$\begin{aligned} P_J^u &= \tilde{C}_J^d - (A-KD)^J O_J^\dagger H_J^d, & (8.29)\\ P_J^y &= \tilde{C}_J^s + (A-KD)^J O_J^\dagger, & (8.30)\\ P_J^e &= (A-KD)^J O_J^\dagger H_J^s, & (8.31) \end{aligned}$$

where we for the sake of simplicity and without loss of generality have put $g = 1$. Equation (8.28) is useful because it shows that the future states X_J is in the range of a matrix consisting of past inputs, $U_{0|J}$, and past outputs, $Y_{0|J}$ (in the deterministic case or when $J \to \infty$). Note that we have introduced the notation, P_J^u, in order to represent the influence from the *past* inputs upon the future. Combining (8.28) and (8.26) gives an alternative to (8.27), i.e. the "past-future" matrix equation,

$$Y_{J|L} = \begin{bmatrix} H_L^d & O_L P_J^u & O_L P_J^y \end{bmatrix} \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} - O_L P_J^e E_{0|J} + H_L^s E_{J|L}. \tag{8.32}$$

The two last terms in (8.32) cannot be predicted from data, i.e., because $E_{0|J}$ and $E_{J|L}$ are built from the innovations process e_k.

It is important to note that a consistent estimate of the system dynamics can be obtained by choosing L and N properly. Choosing $L_{\text{min}} \leq L$ where $L_{\text{min}} = n + \text{rank}(D) - 1$ and letting $N \to \infty$, is in general, necessary conditions for a consistent estimate of the dynamics. See Section 8.3.2 for further details.

On the other side, it is in general, also necessary to let $J \to \infty$ in order to obtain a consistent estimate of the states. The reason for this is that the term $(A - KD)^J = 0$ in this case. Hence, the effect of the initial state matrix X_0 on the future states X_J has died out. We have the following Lemma

Lemma 8.3.1 (Subspace identification of the states) *Let $K \to \infty$ in the data matrices. The projected state matrix is defined as*

$$\begin{aligned} X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} &= O_L^\dagger (\overbrace{Y_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}}^{Z_{J|L}^d} - H_L^d U_{J|L+g-1}) \\ &= \tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J} + (A-KD)^J X_0 / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}. \end{aligned} \tag{8.33}$$

Consider the case when

$$(A-KD)^J X_0 / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = 0, \tag{8.34}$$

which is satisfied when $J \to \infty$ and $(A-KD)$ is stable. This gives

$$X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = X_J, \tag{8.35}$$

and hence we have, in general, the following expression for the future states

$$X_J = O_L^\dagger(\overbrace{Y_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}}^{Z^d_{J|L}} - H_L^d U_{J|L+g-1}). \tag{8.36}$$

△

Proof 8.1 *The proof is divided into two parts.*

Part 1

The relationship between the future data matrices is given by

$$Y_{J|L} = O_L X_J + H_L^d U_{J|L+g-1} + H_L^s E_{J|L}. \tag{8.37}$$

Projecting the row space of each term in (8.37) onto the row space of $\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$ *gives*

$$Y_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = O_L X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} + H_L^d U_{J|L+g-1} + dE_1 \tag{8.38}$$

where the error term is given by

$$dE_1 = H_L^s E_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}. \tag{8.39}$$

It make sense to assume that future noise matrix $E_{J|L}$ is uncorrelated with past data and the future inputs, hence, we have that (w.p.1)

$$\lim_{K \to \infty} dE_1 = 0. \tag{8.40}$$

Part 2

Equation (8.25) gives the relationship between the future state matrix X_J and the past data matrices. Projecting the row space of each term in this equation onto the row space of $\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$ *gives*

$$X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J} + (A - KD)^J X_0 / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \quad (8.41)$$

Letting $J \to \infty$ (or assuming the last term to be zero) gives

$$X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J}. \quad (8.42)$$

Letting $J \to \infty$ and assuming the system matrix $(A - KD)$ for the predicted outputs to be stable in (8.25) shows that

$$X_J = \tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J}. \quad (8.43)$$

Comparing (8.42) and (8.43) gives

$$X_J = X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}. \quad (8.44)$$

Using (8.44) in (8.38) and solving for X_J gives (8.36). □

The condition in (8.35) is usually satisfied for large J, i.e., we have that $\lim_{J\to\infty}(A - KD)^J = 0$ when $A - KD$ is stable. Note also that the eigenvalues of $A - KD$ usually are close to zero for "large" process noise (or "small" measurements noise). Then, $(A - KD)^J$ is approximately zero even for relatively small numbers J. We will now discuss some special cases

Lemma 8.3.2 (SID of states: white input)
Consider a combined deterministic and stochastic system excited with a white input signal. Then

$$X_J = O_L^\dagger Y_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} U_{J|L+g-1}^\perp \quad (8.45)$$

when $J \to \infty$.

Proof 8.2 *This result follows from the proof of Lemma 8.3.1 and (8.36) and using that*

$$X_J U_{J|L+g-1}^\perp = X_J \quad (8.46)$$

when u_k is white and, hence, $X_0 / U_{J|L+g-1} = 0$. □

Lemma 8.3.3 (SID of states: pure stochastic system)
Consider a stochastic system. Then we simply have that

$$X_J = O_L^{\dagger} Y_{J|L}/Y_{0|J} \tag{8.47}$$

when $J \to \infty$ or when $(A - KD)^J X_0/Y_{0|J} = 0$ is satisfied.

Proof 8.3 *This result follows from the proof of Lemma 8.3.1 by putting the measured input variables equal to zero.* □

Lemma 8.3.1 shows that it is in general (i.e. for colored input signals) necessary to know the deterministic part of the system, i.e., the Toepliz matrix H_L^d in (8.36), in order to properly identify the states. This means that the matrices B and E in addition to D and A has to be identified prior to computing the states. I.e. we need to know the deterministic part of the model. However, a special case is given by Lemma 8.3.2 and Equation (8.45) which shows that the states can be identified directly when the input signals is white. Note also that the extended observability matrix O_L is needed in (8.36) and (8.45). O_L can be identified directly from the data. This is proved in the next Section 8.3.2, and this is indeed the natural step in a SID algorithm.

In the case of a white input signal or when $J \to \infty$ then, H_L^d, and the state matrix, X_J, can be computed as by the N4SID algorithm, Van Overschee and De Moor (1996). From (8.32) and (8.28) we have the following lemma

Lemma 8.3.4 (States, X_J, and Toepliz matrix H_L^d: N4SID)
The following LS solution

$$\begin{bmatrix} H_L^d & O_L P_J^u & O_L P_J^y \end{bmatrix} = Y_{J|L} \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\dagger} + dE. \tag{8.48}$$

holds in:

i) The deterministic case, provided the input is PE of order $J+L+g-1$. The error term, $dE = 0$, in this case.

ii) When $J \to \infty$, and the input is PE of infinite order. The error term, $dE = 0$, in this case.

iii) A white u_k gives a consistent estimate of H_L^d irrespective of $J > 0$. However, $O_L P_J^u$ and $O_L P_J^y$ are not consistent estimates in this case. The first $mL \times (L+g)r$ part of the error term, dE, is zero in this case.

Hence, under conditions i) and ii), $O_L P_J^u$ and $O_L P_J^y$ can be computed as in (8.48). Then the states can be consistently estimated as

$$X_J = O_L^{\dagger} \begin{bmatrix} O_L P_J^u & O_L P_J^y \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}, \tag{8.49}$$

provided conditions i) and ii) are satisfied, and $O_L^{\dagger}$ is known.

Proof 8.4 *The PE conditions in the lemma are due to the existence of the LS solution, i.e., the concatenated matrix* $\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \end{bmatrix}$ *has to be of full row rank. From (8.32) we have that the error term in the LS problem is*

$$dE = (-O_L P_J^e E_{0|J} + H_L^s E_{J|L}) \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^\dagger = -O_L P_J^e E_{0|J} \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^\dagger \tag{8.50}$$

It is clear from (8.31) that the error term $dE = 0$ when $J \to \infty$. This proves condition i) in the lemma. Furthermore, the error term, $dE = 0$, in the deterministic case because $E_{0|J} = 0$ in this case. This proves condition ii). Analyzing the error term, dE, for a white input shows that the error term is of the form

$$dE = \begin{bmatrix} 0_{mL\times(L+g)r} & dE_2 & dE_3 \end{bmatrix}^\dagger, \tag{8.51}$$

where the dE_2 and dE_3 are submatrices in dE different from zero. Note that $dE_2 = 0$ for strictly proper systems, $g = 0$, when u_k is white. This proves condition iii).

The states can then be computed by using (8.28) or (8.43), provided conditions i) or ii) are satisfied. □

One should note that in the N4SID algorithm the past horizon is put equal to the future horizon (N4SID parameter i). In order for the above lemma to give the same results as in the N4SID algorithm we have to put $i = L+1$, $J = L+1$ and $g = 1$, i.e so that $J + L = 2L + 1 = 2i$. Note that this last result does not hold in general. It holds in the deterministic case or when $J \to \infty$. The extended observability matrix O_L can be computed as presented in the next section.

8.3.2 The extended observability matrix

An important first step in the SID algorithm is the identification of the system order, n, and the extended observability matrix O_{L+1}. The reason for searching for O_{L+1} is that we have to define A from the shift invariance property, Kung (1978), or a similar method, e.g. as in Di Ruscio (1995). The key is to compute a special projection matrix from the known data. This is done without using the states. We will in this section show how this can be done for colored input signals.

Lemma 8.3.5 (SID of the extended observability matrix)
The following projections are equivalent

$$Z_{J|L+1} = (Y_{J|L+1} / \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) U_{J|L+g}^\perp \tag{8.52}$$

$$Z_{J|L+1} = (Y_{J|L+1} U_{J|L+g}^{\perp}) / (\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} U_{J|L+g}^{\perp}) \tag{8.53}$$

$$Z_{J|L+1} = Y_{J|L+1} / (\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} U_{J|L+g}^{\perp}) \tag{8.54}$$

Furthermore, $Z_{J|L+1}$ is related to the extended observability matrix O_{L+1} as

$$Z_{J|L+1} = O_{L+1} X_J^a, \tag{8.55}$$

where the "projected states" X_J^a can be expressed as

$$\begin{aligned}
X_J^a &= (X_J / \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) U_{J|L+g}^{\perp} && (8.56) \\
&= (\tilde{C}_J^d U_{0|J} + \tilde{C}_J^s Y_{0|J} - (A-KD)^J X_0 / \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}) U_{J|L+g}^{\perp} && (8.57) \\
&= (X_J - (A-KD)^J X_0 \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp}) U_{J|L+g}^{\perp} && (8.58) \\
&= (X_J + (A-KD)^J O_J^{\dagger} H_J^s E_{0|J} \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp}) U_{J|L+g}^{\perp} && (8.59)
\end{aligned}$$

Furthermore, the column space of $Z_{J|L+1}$ coincide with the column space of O_{L+1} and $n = rank(Z_{J|L+1})$ if $rank(X_J^a) = n$.

Proof 8.5 *The proof is divided into two parts. In the first part (8.52) and (8.55) with the alternative expressions in (8.56) to (8.58) are proved. In the second part the equivalence with (8.52), (8.53) and (8.54) are proved.*

Part 1 *Projecting the row space of each term in (8.26) with $L := L+1$ onto the row space of $\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$ gives*

$$Y_{J|L+1} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = O_{L+1} X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} + H_{L+1}^d U_{J|L+g-1} + dE_1 \tag{8.60}$$

where we have used (8.13). Then, w.p.1

$$\lim_{K \to \infty} dE_1 = 0, \tag{8.61}$$

where the error term, dE_1, is given by (8.39) with $L := L+1$. Removing the effect of the future input matrix, $U_{J|L+g-1}$, on (8.60) gives (8.52) and (8.55) with X_J^a as in (8.56).

Furthermore, projecting the row space of each term in (8.25) onto the row space of $\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$ *gives*

$$X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J} + (A - KD)^J X_0 / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{8.62}$$

From (8.25) we have that

$$\tilde{C}_J^s Y_{0|J} + \tilde{C}_J^d U_{0|J} = X_J - (A - KD)^J X_0. \tag{8.63}$$

Combining (8.60), (8.62) and (8.63) gives (8.52) and (8.57)-(8.58).

Part 2 *It is proved in Di Ruscio (1997) that*

$$\begin{aligned} Z_{J|L+1} &= Y_{J|L+1} / \begin{bmatrix} U_{J|L+g} \\ W \end{bmatrix} U_{J|L+g}^{\perp} \\ &= Y_{J|L+1} U_{J|L+g}^{\perp} W^T (W U_{J|L+g}^{\perp} W^T)^{-1} W U_{J|L+g}^{\perp}, \end{aligned} \tag{8.64}$$

where

$$W = \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}. \tag{8.65}$$

Using that $U_{J|L+g}^{\perp} U_{J|L+g}^{\perp} = U_{J|L+g}^{\perp}$ in (8.64) proves the equivalence between (8.53), (8.54) and (8.52). □

Lemma 8.3.6 (Consistency: Stochastic and deterministic systems)
Let $J \to \infty$, then

$$Z_{J|L+1} = O_{L+1} X_J U_{J|L+g}^{\perp}, \tag{8.66}$$

where $Z_{J|L+1}$ is defined as in Lemma 8.3.5. A sufficient condition for consistency, and that O_{L+1} is contained in the column space of $Z_{J|L+1}$, is that there are no pure state feedback.

Proof 8.6 *Letting $J \to \infty$ in (8.58) gives (8.66). This can also be proved by using (8.44) in (8.56). Furthermore, if there are pure state feedback then $X_J U_{J|L+g}^{\perp}$ will lose rank below the normal rank which is n.* □

Lemma 8.3.7 (Deterministic systems)
For pure deterministic systems we have that (8.66) can be changed to

$$Z_{J|L+1} =: Y_{J|L+1} U_{J|L+g}^{\perp} = O_{L+1} X_J U_{J|L+g}^{\perp}. \tag{8.67}$$

The extended observability matrix O_{L+1} can be computed from the column space of $Y_{J|L+1} U_{J|L+g}^{\perp}$. Furthermore, one can let $J = 0$ in the deterministic case.

Proof 8.7 *This follows from (8.66) and Lemma 8.3.5 by excluding the projection which removes the noise.* □

Lemma 8.3.8 (Stochastic systems)
For pure stochastic systems we have that (8.66) can be changed to

$$Z_{J|L+1} =: Y_{J|L+1}/Y_{0|J} = O_{L+1}X_J. \tag{8.68}$$

The extended observability matrix O_{L+1} can be computed from the column space of $Y_{J|L+1}/Y_{0|J}$.

Proof 8.8 *This follows from (8.66) and Lemma 8.3.5 by excluding the input matrices from the equations and definitions.* □

8.3.3 Identification of the stochastic subsystem

We will in this section prove that, when the extended observability matrix is known (from Section 8.3.2), the kalman filter gain matrix can be identified directly from the data. Furthermore, it is proved that the noise innovations process can be identified directly in a first step in the DSR subspace algorithm. This result was first presented in Di Ruscio (1995) without proof. Some results concerning this is also presented in Di Ruscio (2001) and (2003).

Lemma 8.3.9 (The innovations)
Define the following projection from the data

$$Z^s_{J|L+1} = Y_{J|L+1} - Y_{J|L+1}/\begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = Y_{J|L+1}\begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp}. \tag{8.69}$$

Then w.p.1 as $J \to \infty$

$$Z^s_{J|L+1} = H^s_{L+1}E_{J|L+1}. \tag{8.70}$$

Hence, the Toeplitz matrix H^s_{L+1} (with Markov matrices F, DC, ..., $DA^{L-1}C$) for the stochastic subsystem is in the column space of $\frac{1}{\sqrt{K}}Z^s_{J|L+1}$ since $\frac{1}{K}E_{J|L+1}E^T_{J|L+1} = I_{L+1\times L+1}$.

Proof 8.9 *The relationship between the future data matrices is given by*

$$Y_{J|L} = O_LX_J + H^d_LU_{J|L+g-1} + H^s_LE_{J|L}. \tag{8.71}$$

Projecting the row space of each term in (8.71) onto the row space of $\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$ *gives*

$$Y_{J|L}/\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = O_LX_J/\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} + H^d_LU_{J|L+g-1} + dE_1, \tag{8.72}$$

then, w.p.1

$$\lim_{K\to\infty} dE_1 = 0, \tag{8.73}$$

where dE_1 is given in (8.39). Furthermore,

$$\lim_{J\to\infty} X_J / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = X_J, \tag{8.74}$$

where we have used Equations (8.44) and (8.38). From (8.71), (8.72) and (8.74) we have that

$$Y_{J|L} - Y_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = H_L^s E_{J|L}. \tag{8.75}$$

Putting $L := L+1$ in (8.75) gives (8.69). □

Note the following from Lemma 3.9. The innovations can be identified directly as for $g = 1$

$$Z_{J|1}^s = F E_{J|1} = Y_{J|1} - Y_{J|1} / \begin{bmatrix} U_{J|1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{8.76}$$

or for $g = 0$ when $E = 0$

$$Z_{J|1}^s = F E_{J|1} = Y_{J|1} - Y_{J|1} / \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{8.77}$$

One should note that (8.77) holds for both open and closed loop systems. For closed loop systems it make sense to only consider systems in which the direct feed-through matrix, E, from the input u_k to the output y_k is zero. This result will be used in order to construct a subspace algorithm whic gives consisten results for close loop systems, se Section 5.

It is now possible to directly identify the matrices C and F in the innovations model (8.1) and (8.2) and K and Δ in the Kalman filter (8.16) and (8.17). Two methods are presented in the following. The first one is a direct covariance based method for computing K and Δ and the second one is a more numerically reliable "square root" based method for computing C and F.

Lemma 8.3.10 (correlation method for K and Δ) *Define the projection matrix $Z_{J|L+1}^s$ as in (8.69) and define the correlation matrix*

$$\Delta_{L+1} = \frac{1}{K} Z_{J|L+1}^s (Z_{J|L+1}^s)^T = H_{L+1}^s (H_{L+1}^s)^T. \tag{8.78}$$

where the Toepliz matrix H^s_{L+1} *can be partitioned as*

$$H^s_{L+1} = \begin{bmatrix} F & 0_{m\times Lm} \\ O_L C & H^s_L \end{bmatrix}, \tag{8.79}$$

where $C = KF$. *Hence, (8.78) can be written as*

$$\Delta_{L+1} = \begin{bmatrix} \Delta_{11} & \Delta_{12} \\ \Delta_{21} & \Delta_{22} \end{bmatrix} = \begin{bmatrix} FF^T & F(O_L C)^T \\ O_L C F^T & O_L C (O_L C)^T + H^s_L (H^s_L)^T \end{bmatrix}. \tag{8.80}$$

From this we have

$$E(\varepsilon_k \varepsilon_k^T) = FF^T = \Delta_{11} \tag{8.81}$$

and

$$K = CF^{-1} = O_L^\dagger \Delta_{21} \Delta_{11}^{-1}. \tag{8.82}$$

Lemma 8.3.11 (square-root method for C **and** F**)** *The LQ decomposition of* $\frac{1}{\sqrt{K}} Z^s_{J|L+1}$ *gives*

$$\frac{1}{\sqrt{K}} Z^s_{J|L+1} = R_{33} Q_3. \tag{8.83}$$

Then, the Toeplitz matrix H^s_{L+1}*, and the Markov matrices* F*,* $DC, \ldots, DA^{L-1}C$*, are given directly by*

$$H^s_{L+1} = R_{33}. \tag{8.84}$$

F *can be taken as one of the diagonal* $m \times m$ *block matrices in* R_{33}*, e.g. the lower left sub-matrix, i.e.*

$$F = R_{33}(Lm + 1 : (L+1)m, Lm + 1 : (L+1)m), \tag{8.85}$$

or as the mean of all the diagonals. Furthermore,

$$O_L C = R_{33}(m + 1 : (L+1)m, 1 : m). \tag{8.86}$$

The system matrix C *is given by*

$$C = O_L^\dagger O_L C = O_L^\dagger R_{33}(m + 1 : (L+1)m, 1 : m). \tag{8.87}$$

The Kalman filter gain matrix and the innovations covariance matrix are given by

$$\begin{aligned} K &= CF^{-1} && (8.88) \\ &= O_L^\dagger R_{33}(m+1 : (L+1)m, 1 : m) R_{33}^{-1}(Lm+1 : (L+1)m, Lm+1 : (L+1)m) \\ \Delta &= FF^T. && (8.89) \end{aligned}$$

8.3.4 SID of the deterministic subsystem

The parameters in the B and E matrices can be computed from an optimal least squares problem. A solution to this is given in the following

Lemma 8.3.12 (Extended state space model)
The states can be eliminated from the state space model (8.1) and (8.2) to yield the so called Extended State Space Model (ESSM)

$$Y_{J+1|L} = \tilde{A}_L Y_{J|L} + \tilde{B}_L U_{J|L+g} + \tilde{C}_L E_{J|L+1}, \tag{8.90}$$

where

$$\begin{aligned}
\tilde{A}_L &\overset{\text{def}}{=} O_L A (O_L^T O_L)^{-1} O_L^T && \in \mathbb{R}^{Lm\times Lm}, && (8.91)\\
\tilde{B}_L &\overset{\text{def}}{=} \begin{bmatrix} O_L B & H_L^d \end{bmatrix} - \tilde{A}_L \begin{bmatrix} H_L^d & 0_{Lm\times r} \end{bmatrix} && \in \mathbb{R}^{Lm\times(L+g)r}, && (8.92)\\
\tilde{C}_L &\overset{\text{def}}{=} \begin{bmatrix} O_L C & H_L^s \end{bmatrix} - \tilde{A}_L \begin{bmatrix} H_L^s & 0_{Lm\times m} \end{bmatrix} && \in \mathbb{R}^{Lm\times(L+1)m}. && (8.93)
\end{aligned}$$

Proof 8.10 *Putting $J =: J+1$ in (8.37) and substituting $X_{J+1} = AX_J + BU_J + CE_J$ into this equation gives*

$$Y_{J+1|L} = O_L A X_J + \begin{bmatrix} O_L B & H_L^d \end{bmatrix} U_{J|L+g} + \begin{bmatrix} O_L C & H_L^s \end{bmatrix} E_{J|L+1}. \tag{8.94}$$

Equation (8.37) can be solved for X_J when (A, D) is observable, i.e.,

$$X_J = O_L^\dagger (Y_{J|L} - H_L^d U_{J|L+g-1} - H_L^s E_{J|L}), \tag{8.95}$$

where $O_L^\dagger = (O_L^T O_L)^{-1} O_L^T$ is the More-Penrose pseudo inverse of O_L. Substituting (8.95) into (8.94) gives (8.90)-(8.93). □

Lemma 8.3.13 (Projection matrix for the deterministic subsystem)
Define the projection matrix

$$Z_{J|L+1}^d = Y_{J|L+1} / \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}. \tag{8.96}$$

This matrix can be partitioned into matrices $Z_{J+1|L}^d$ and $Z_{J|L}^d$ which satisfy the deterministic model

$$Z_{J+1|L}^d = \tilde{A}_L Z_{J|L}^d + \tilde{B}_L U_{J|L+g}. \tag{8.97}$$

Proof 8.11 *This follows from (8.90) and (8.13) and that*

$$\lim_{K\to\infty} \tilde{C}_L E_{J|L+1} / \begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = 0, \tag{8.98}$$

when the future inputs, $U_{J|L+g}$, the past data, $U_{0|J}$ and $Y_{J|J}$, are all independent of the future noise term, $\tilde{C}_L E_{J|L+1}$. □

Lemma 8.3.14 (SID of the system matrices B and E)
Define from (8.97) the linear equation

$$\mathcal{Y} = \tilde{B}_L \mathcal{U}, \tag{8.99}$$

where

$$\begin{aligned} \mathcal{Y} &= Z^d_{J+1|L} - \tilde{A}_L Z^d_{J|L}, & (8.100)\\ \mathcal{U} &= U_{J|L+g}. & (8.101)\end{aligned}$$

From (8.99) a least squares problem

$$\min_{B,E} \parallel \mathcal{Y} - \tilde{B}_L(B,E)\mathcal{U} \parallel_F^2 \tag{8.102}$$

for the unkown elements in B and E are defined as

$$vec(\mathcal{Y}) = \mathcal{X} vec(\begin{bmatrix} B \\ E \end{bmatrix}), \tag{8.103}$$

which can be solved for the unknown parameters as

$$vec(\begin{bmatrix} B \\ E \end{bmatrix}) = \mathcal{X}^\dagger vec(\mathcal{Y}), \tag{8.104}$$

where $\mathcal{X}^\dagger = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T$. The matrix $\mathcal{X}$ is defined as follows

$$\mathcal{X} \overset{\text{def}}{=} \sum_{i=1}^{L+g} R_i^T \otimes (E_{i-1} - \tilde{A}_L E_i) \ \in \ \mathbb{R}^{LmK \times (n+gm)r}, \tag{8.105}$$

where $\otimes$ denotes the Kronecker tensor product. The matrices R_i and E_i are defined in the following. The matrices

$$R_i \in \mathbb{R}^{r \times K} \ \forall \ i = 1, \cdots, L+g, \tag{8.106}$$

are r-block rows in the $\mathcal{U} \in \mathbb{R}^{(L+g)r \times K}$ matrix. I.e. extracted from

$$\mathcal{U} = \begin{bmatrix} R_1 \\ \vdots \\ R_{L+g} \end{bmatrix}. \tag{8.107}$$

The matrices

$$E_{i-1} \in \mathbb{R}^{Lm \times (n+mg)} \ \forall \ i = 1, \cdots, L+g, \tag{8.108}$$

are defined as follows:

$$E_0 = \begin{bmatrix} O_L & 0_{Lm \times m}, \end{bmatrix}, \quad E_{L+1} = 0_{Lm \times (n+m)}, \tag{8.109}$$

$$E_1 = \begin{bmatrix} 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \\ DA & 0_{m\times m} \\ \vdots & \vdots \\ DA^{L-2} & 0_{m\times m} \end{bmatrix}, E_2 = \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \\ \vdots & \vdots \\ DA^{L-3} & 0_{m\times m} \end{bmatrix}, E_L = \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & 0_{m\times m} \\ \vdots & \vdots \\ 0_{m\times n} & I_{m\times m} \end{bmatrix} \tag{8.110}$$

The matrix $I_{m\times m}$ denotes the $m \times m$ identity matrix.

Proof 8.12 *From (8.92) we have that $\tilde{B}_L$ is a linear function of B and E when A and D are given. The matrix $\mathcal{X}$ in the LS problem (8.103) is defined from $\tilde{B}_L \mathcal{U}$ by using the identity*

$$vec(AXB) = (B^T \otimes A) vec(X). \tag{8.111}$$

□

Note that the number of columns in $\mathcal{Y}$ and $\mathcal{U}$, which is defined in (8.100) and (8.101), can be reduced to $K =: (L+g)r$ by post-multiplying both (8.100) and (8.101) with $U_{J|L+g}^T$. However, this does not affect the estimates of B and E but will in general reduce the computation. Another variant, which should be preferred for numerical reasons, is to define $\mathcal{Y}$ and $\mathcal{U}$ from the R matrix provided by the RQ/LQ decomposition. This will also reduce the number of columns to $K =: (L+g)r$ in Lemma 8.3.14.

Note that only a matrix of size $(n+gm)r \times (n+gm)r$ has to be inverted (i.e., the matrix $\mathcal{X}^T\mathcal{X}$ in the LS solution (8.104)) in order to solve for the unknown parameters in B and E. This method combined with the LQ decomposition is found to be very efficient. The method in Lemma 8.3.14 require only that the input is exciting of order $n+gm$, and hence, independent of the user specified parameters L and J. This is consistent with the lower bound on the order of persistence of excitation for consistent estimation of an nth order possibly proper ($g=1$) linear system.

Note that the alternative strategy of first solving for $\tilde{B}_L$ in (8.97) and then extracting B and E would require the inversion of an $(L+g)r \times (L+g)r$ matrix $U_{J|L+g} U_{J|L+g}^T$. This matrix may be singular for colored input signals, and hence is not preferred.

8.4 Closed loop subspace identification

We have in Section 8.3.2 shown that the extended observability matrix O_{L+1} can be estimated from the column space of the projection matrix $Z_{J|L+1}$ as defined in (8.52). Let us look at the error term in this projection. We have

$$Z_{J|L+1} = O_{L+1} X_J^a + dZ, \tag{8.112}$$

The error term dZ is given by

$$\begin{aligned} dZ &= H_{L+1}^s (E_{J|L+1} / \begin{bmatrix} U_{J|L+g} \\ W \end{bmatrix}) U_{J|L+g}^\perp \\ &= H_{L+1}^s E_{J|L+1} U_{J|L+g}^\perp W^T (W U_{J|L+g}^\perp W^T)^{-1} W U_{J|L+g}^\perp, \\ &\approx -H_{L+1}^s E_{J|L+1} / U_{J|L+g} W^T (W U_{J|L+g}^\perp W^T)^{-1} W U_{J|L+g}^\perp, \end{aligned} \tag{8.113}$$

where W is defined in (8.65). We have in the last expression in (8.113) used that $E_{J|L+1} W^T / K \approx 0$ when the number of columns K tends to infinity. The remaining projection in the error term is then $E_{J|L+1}/U_{J|L+g}$. This term will also be approximately zero for open loop and many closed loop problems, which will be pointed out in the following. However, the term $E_{J|L+1}/U_{J|L+g}$ may be non-zero and cause biased estimates for feedback systems in which the control is directly proportional to the innovations noise. We will in the next sections discuss how to overcome this problem. We will also stress that biased estimates may be more reliable than estimates from an unbiased algorithm because the variance may be small. This is illustrated in the section of examples.

8.4.1 Closed loop subspace identification: Using a filter in the feeback loop!

Since we are allowing the inputs to be colored the question whether it is possible with feedback in the inputs have to be pointed out. An external (dither) signal, i.e., a reference signal, should be used to excite the system when collecting data for subspace identification. The natural excitations from process disturbances are often insufficient. The SID algorithm, e.g. DSR, works perfect for closed loop deterministic systems. Our simulation results also shows that the results may be good even for combined deterministic and stochastic systems, however, the results depends on the dither signal or the signal to noise ratio. If the signal to noise ratio is low then there may be a bias in the estimates. However, the variance may be small. There may also exist an "optimal" dither signal which gives very accurate subspace estimates (small bias and small variance) even for systems with a large signal to noise ratio. This will be illustrated in Example 8.7.4. A white noise or random binary signal in the reference usually gives very good closed loop identification results. Furthermore, a minimum of measurements noise is, as always, to be preferred in order to obtain good closed loop estimates.

It is believed that SID of systems with state feedback or feedback from Kalman filter states would work well, provided an external dither signal is introduced in the loop. The reason for this is that the states are "noise-free" and not correlated with the innovations noise. There are no problems by using subspace identification methods in these cases.

The key is to make the term $E_{J|L+1}/U_{J|L+g}$ small, which is equivalent to making the error term (8.39) small.

The (open loop) subspace identification methods may give biased estimates for closed loop systems as in Figures 8.1 and 8.3 when the signal-to-noise ratio is low. The reason for this is that the error term in (8.39) is not zero when the future inputs, $U_{J|L+g-1}$, are correlated with the future noise, $E_{J|L}$. Most of our simulations shows that the bias in the DSR algorithm, due to noisy closed loop data, is less than the bias in the other algorithms as N4SID, SUBID (Van Overschee and De Moor (1996)), MOESP (Verhagen (1994)).

One of our solutions to the bias problem is to include a filter, e.g. a first order low-pass filter, in the feedback path of the control system as illustrated

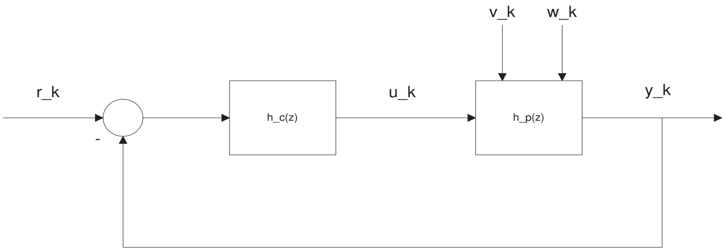


Figure 8.1: Standard feedback system with reference signal, r_k, controller represented with $u_k = h_c(z)(r_k - y_k)$ and the system represented with $h_p(z)$. Possibly process and measurements noise are represented with v_k and w_k, respectively.

in Figure 8.2. This filter will reduce or eliminate the feedback problem when using subspace identification algorithms. Hence, the input and output data, u_k and y_k, can be used directly to identify a state space model of the system. The reason for this is that the input is no longer directly proportional to the measurements noise w_k in the output $y_k = Dx_k + w_k$ (or the innovations e_k in the output $y_k = Dx_k + e_k$). This solution to the feedback problem will be illustrated in Example 8.7.3.

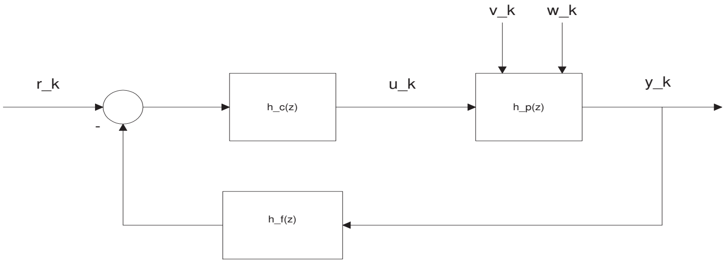


Figure 8.2: Feedback system with a filter in the feeback loop in order to eliminate problems with feeback in the data when using subspace identification algorithms. Here, u_k is the control/input signal, y_k the output signal and r_k, is the reference signal. The controller is represented with $u_k = h_c(z)(r_k - \bar{y}_k)$ where $\bar{y}_k = h_f(z)y_k$ is the filtered output. The system is represented with the transfer function $h_p(z)$. The controller is represented with the transfer function $h_c(z)$ and the filter is represented with the transfer function $h_f(z)$. Possibly process and measurements noise are represented with v_k and w_k, respectively.

Note that the control system in Figure 8.2 is a special case of the more general control configuration in Figure 8.3.

Note that SID algorithms are very useful for model predictive control in which the control input signal often is the reference signal to some local controller for the process. Hence, the SID algorithm are used to identify the closed loop model from the reference signal to the output. In this case we do not have problems with feedback in the data.

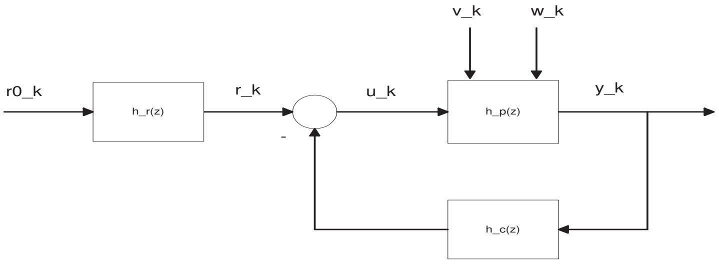


Figure 8.3: Alternative feedback system with reference signal, r_k, controller represented with $u_k = r_k - h_c(z)y_k$ and the system represented with $h_p(z)$. Possibly process and measurements noise are represented with v_k and w_k, respectively.

8.4.2 Closed loop subspace identification: Using the controller

The knowledge of the controller or the reference signal can be used to obtain consistent subspace identification algorithms for closed loop systems. Consider a linear state space model of the controller in Figure 8.1, i.e.

$$\begin{aligned} x_{k+1}^c &= A_c x_k^c + B_c(r_k - y_k), \quad (8.114)\\ u_k &= D_c x_k^c + E_c(r_k - y_k), \quad (8.115) \end{aligned}$$

where $A_c \in \mathbb{R}^{n_c \times n_c}$, $B_c \in \mathbb{R}^{n_c \times m}$, $D_c \in \mathbb{R}^{r \times n_c}$ and $E_c \in \mathbb{R}^{r \times m}$ is the state space model matrices of the controller and $x_k^c \in \mathbb{R}^n$ is the controller state vector. Note also that the results which follows also holds for the control strategy in Figure 8.3.

We will in the following assume that the controller is linear and that the quadruple (A_c, B_c, D_c, E_c) matrices in addition to the input and output data, u_k and y_k, are known. One should also note that the linear controller matrices can be exactly identified if r_k, u_k and y_k are known. The problem of identifying the controller is deterministic and one can in this case perfectly identify (A_c, B_c, D_c, E_c) by using $r_k - y_k$ as inputs and u_k as outputs by using e.g. the DSR subspace algorithm (provided that noise-free r_k's and u_k's are given).

Consider the following matrix equation obtained from the state space model (8.114) and (8.115) of the controller in Figure 8.1, i.e.

$$U_{J|L} = O_L^c X_J^c + H_L^c(R_{J|L} - Y_{J|L}). \quad (8.116)$$

We will now adopt the idea in Van Overschee and De Moor (1997) and define a signal/matrix $M_{J|L}$ from (8.116) which is orthogonal to the future noise matrix $E_{J|L}$ as

$$M_{J|L} \overset{\text{def}}{=} U_{J|L} + H_L^c Y_{J|L} = O_L X_J^c + H_L^c R_{J|L} \quad (8.117)$$

The signal/matrix 8.117 was introduced by Van Overschee and De Moor (1997) in order to solve the bias problem in the subspace identification algorithms.

Note also that a similar signal can be defined from the control system in Figure 8.3. The only difference is that the right hand side of (8.117) becomes

$R_{J|L} - O_L X_J^c$ in this case. This matrix is also orthogonal to the future noise and the closed loop subspace identification algorithm which will be presented in the following thus holds for both control systems as presented in Figures 8.1 and 8.3. The main point of introducing $M_{J|L}$ is that

$$E_{J|L}/M_{J|L} = 0, \tag{8.118}$$

which holds if $E_{J|L}$ is orthogonal to both X_J^c and $R_{J|L}$.

8.4.3 Closed loop subspace identification: Indirect method

We will in this section derive a consistent version of the closed loop subspace identification algorithm which is presented in Van Overschee and De Moor (1997). We have the following consistent projection lemma for closed loop subspace system identification

Lemma 8.4.1 (Closed loop SID)
Given the following closed loop projection

$$Z_{J|L+1} = (Y_{J|L+1}/ \begin{bmatrix} M_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix})M^{\perp}_{J|L+g} \tag{8.119}$$

Then, $Z_{J|L+1}$ is related to the extended observability matrix O_{L+1} as

$$Z_{J|L+1} = T^{-1} O_{L+1} X_J^a, \tag{8.120}$$

where $T \in \mathbb{R}^{(L+1)m \times (L+1)m}$ is a lower block Toepliz matrix given by

$$T = I_{(L+1)m} + H^d_{L+1} H^c_{L+g} \quad \text{when } g = 1 \tag{8.121}$$

$$T = I_{(L+1)m} + \begin{bmatrix} H^d_{L+1} H^c_{L+g} & 0_{(L+1)m \times m} \end{bmatrix} \quad \text{when } g = 0 \tag{8.122}$$

Furthermore, the system order is given by

$$n = rank(Z_{J|L+1}). \tag{8.123}$$

From the SVD

$$Z_{J|L+1} = U_1 S_1 V_1^T + U_2 S_2 V_2^T, \tag{8.124}$$

where the n large/dominant singular values are contained on the diagonal in S_1 and the other zero/smaller singular values on the diagonal of S_2. Furthermore, $U_1 \in \mathbb{R}^{(L+1)m \times n}$ and $U_2 \in \mathbb{R}^{(L+1)m \times ((L+1)m-n)}$.

From this we have the estimate

$$T^{-1} O_{L+1} = U_1. \tag{8.125}$$

Furthermore, the "autonomous" states are determined as $X_J^a = S_1 V_1^T$.

Proof 8.13 *A simple proof is given as follows. From (8.26) we have*

$$Y_{J|L+1} = O_{L+1}X_J + H^d_{L+1}U_{J|L+g} + H^s_{L+1}E_{J|L+1}. \tag{8.126}$$

Adding $H^d_{L+1}H^c_{L+g}Y_{J|L+g}$ on both sides of (8.126) and using the definition in (8.117) gives

$$TY_{J|L+1} = O_{L+1}X_J + H^d_{L+1}M_{J|L+g} + H^s_{L+1}E_{J|L+1}. \tag{8.127}$$

where T is as in (8.121) and (8.122). Since the matrix $\begin{bmatrix} M_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}$ is uncorrelated (orthogonal) to the future noise matrix $E_{J|L+1}$ we have that

$$TZ^d_{J|L+1} = O_{L+1}X^d_J + H^d_{L+1}M_{J|L+g} \tag{8.128}$$

where

$$Z^d_{J|L+1} =: Y_{J|L+1}/\begin{bmatrix} M_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}, \tag{8.129}$$

and

$$X^d_{J|L+1} =: X_J/\begin{bmatrix} M_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix}, \tag{8.130}$$

The lower triangular matrix T is non-singular if $I_m + EE_c$ is non-singular. Hence,

$$Z^d_{J|L+1} = T^{-1}O_{L+1}X^d_J + T^{-1}H^d_{L+1}M_{J|L+g} \tag{8.131}$$

Post-multiplication of (8.131) with $M^{\perp}_{J|L+g}$ proves (8.119) -(8.121).

Furthermore we have that

$$U_2^T Z^d_{J|L+1} = U_2^T T^{-1}H^d_{L+1}M_{J|L+g} \tag{8.132}$$

where U_2 is the left singular vectors from the SVD of $Z_{J|L+1}$ which is related to the "zero/small" singular values. Equation (8.132) is obtained by pre-multiplying (8.131) with U_2^T and using that $T^{-1}O_{L+1} = U_1$ and $U_2^T U_1 = 0$.

Equation (8.132) is a linear equation of the elements in the lower block triangular Toepliz matrix $T^{-1}H^d_{L+1}$. The solution to this problem is an important step in the closed loop subspace algorithm and therefore needs further discussion. In order to do this we write (8.132) as

$$\mathcal{Y} = \mathcal{U}\mathcal{K}M_{J|L+g}, \tag{8.133}$$

where

$$\begin{aligned} \mathcal{Y} &=: U_2^T Z_{J|L+1}^d & (8.134) \\ \mathcal{U} &=: U_2^T & (8.135) \end{aligned}$$

$$\mathcal{K} \overset{\text{def}}{=} T^{-1} H_{L+1}^d \in \mathbb{R}^{(L+1)m \times (L+g)r} \tag{8.136}$$

The matrix $\mathcal{K}$ is a lower block triangular Toepliz matrix with $m \times r$ blocks K_i $\forall\, i = 1, \ldots, L+1$. Hence, we can solve (8.132) (or equivalent (8.133) in a least squares optimal sense for the parameters in $\mathcal{K}$. An algorithm for doing this is presented in Van Overschee and De Moor (1996b) Appendix C.

The problem of solving (8.132) is very similar to the least squares problem in Section 8.3.4 for determining B and E for the open loop subspace problem.

Since H_{L+g}^c is known the matrix H_{L+1}^d is simply obtained from $\mathcal{K}$ as (when $g = 1$)

$$H_{L+1}^d = \mathcal{K}(I_{(L+1)r} - H_{L+g}^c \mathcal{K})^{-1}. \tag{8.137}$$

Finally, the extended observability matrix of the system can be obtained as (when $g = 1$)

$$O_{L+1} = (I_{(L+1)m} + H_{L+1}^d H_{L+g}^c) U_1. \tag{8.138}$$

The system matrices A and D are then computed from O_{L+1}. The system matrices B and E are computed from H_{L+1}^d and O_L.

The stochastic part of the system is determined very similar to the theory in Section 8.3.3. The difference is that the projection matrix now is given by

$$Z_{J|L+1}^s = Y_{J|L+1} - Y_{J|L+1} / \begin{bmatrix} M_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = T^{-1} H_{L+1}^s E_{J|L+1} \tag{8.139}$$

The projection matrices $Z_{J|L+1}$, $Z_{J|L+1}^d$ and $Z_{J|L+1}^s$ can effectively be computed by the LQ/QR decomposition.

The above ideas is among other details used to construct a MATLAB function, **dsr_cl.m**, for consistent closed loop subspace identification.

8.4.4 Closed loop subspace identification: Direct method

A drawback with the above algorithm is that T and H_{L+1}^d have to be identified before the extended observability matrix O_{L+1} and the system matrices A and D could be identified. We will in this section present a solution to the closed loop subspace identification problem which is more consistent with the DSR subspace algorithm in which O_{L+1} and A and D is identified directly in a first step. We have the following lemma

Lemma 8.4.2 (Direct closed loop SID)
The extended observability matrix O_{L+1} is obtained from the following projection

$$\begin{aligned} Z_{J|L+1} &= Z^d_{J|L+1}(M_{J|L+g} - H^c_{L+1}Z^d_{J|L+1})^{\perp} \\ &= Z^d_{J|L+1}(U_{J|L+g} + H^c_{L+1}Z^s_{J|L+1})^{\perp} = O_{L+1}X^a_J \end{aligned} \tag{8.140}$$

where $Z^d_{J|L+1}$ and $Z^s_{J|L+1}$ are defined in (8.129) and (8.139), respectively.

Furthermore, B and E (or also H^d_{L+1}) can be obtained from

$$Z^d_{J|L+1} = O_{L+1}X^d_J + H^d_{L+1}(M_{J|L+g} - H^c_{L+1}Z^d_{J|L+1}) \tag{8.141}$$

or from (as in the standard DSR algorithm) the equation

$$Z^d_{J+1|L+1} - \tilde{A}_L Z_{J|L} = \tilde{B}_L(M_{J|L+g} - H^c_{L+1}Z^d_{J|L+1}) \tag{8.142}$$

Proof 8.14 *From (8.128) we have that*

$$(I + H^d_{L+1}H^c_{L+g})Z^d_{J|L+1} = O_{L+1}X^d_J + H^d_{L+1}M_{J|L+g}. \tag{8.143}$$

Rearranging (8.143) gives

$$Z^d_{J|L+1} = O_{L+1}X^d_J + H^d_{L+1}(M_{J|L+g} - H^c_{L+g}Z^d_{J|L+1}) \tag{8.144}$$

The closed loop subspace algorithm which is presented in this section is very similar to the open loop subspace identification algorithm which is presented in Section 8.3. The only difference is that the projection matrices are modified to incorporate the Markov parameters of the controller (the Toepliz matrix H^c_{L+g}.

8.5 A new subspace identification method for closed and open loop systems

It was presented in Di Ruscio (1995) and proved in Di Ruscio (2001) that the noise innovation process could be identified directly from the data in a first step. This approach is valid for both open and closed loop systems.

We will in the following consider a closed loop system in which it make sense to assume that $E = 0$. Putting $g = 0$ and letting $J \to \infty$ in Equation (8.27) gives

$$Y_{J|1} = D\begin{bmatrix} \tilde{C}^d_J & \tilde{C}^s_J \end{bmatrix}\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} + FE_{J|1}. \tag{8.145}$$

Hence, the innovations can simply be identified as (for $g = 0$)

$$Z^s_{J|1} = FE_{J|1} = Y_{J|1} - Y_{J|1}/\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} \tag{8.146}$$

This result is also obtained directly from Lemma 3.9, and Di Ruscio (2001). It is clear that the above approach is valid for both open and closed loop systems since the past data, $U_{0|J}$ and $Y_{0|J}$, is uncorrelated with the future noise $E_{J|1}$.

The innovations is known and the DSR algorithm is used directly with the innovations as extra inputs in order to identify the "deterministic" model

$$\begin{aligned} x_{k+1} &= Ax_k + \left[\begin{array}{cc} B & K \end{array} \right] \left[\begin{array}{c} u_k \\ \varepsilon_k \end{array} \right], & (8.147) \\ y_k - \varepsilon_k &= Dx_k. & (8.148) \end{aligned}$$

We now simply solve a deterministic subspace identification problem in order to obtain the system matrices A, B, K and D. Any efficient subspace method, e.g. the DSR method as presented earlier in this paper, can be used.

The DSR algorithm for closed loop systems is sketched simply as follows:

1. From the known input and output data u_k and y_k $\forall$ $k = 0, 1, \ldots, N-1$ and a large past horizon, J, the future innovations matrix $Z^s_{J|1} = FE_{J|1}$ is computed from (8.146). The projection involved can be computed directly from the definition or preferably from a QR decomposition.

2. The innovations sequence ε_k $\forall$ $k = J, J+1, \ldots, N-1$ in the Kalman filter is given directly from the corresponding data in $Z^s_{J|1}$. From the known innovations sequence we form the Hankel matrix $E_{J|L+1}$.

3. Define the known input and output data sequence u_k and y_k $\forall$ $k = J, J+1, \ldots, N-1$ and form the matrices $U_{J|L}$ and $Y_{J|L+1}$.

4. The data from step 2 and 3 and a specified future horizon, L, are used to identify the state space model by an efficient subspace identification method. As a rule of thumb, chose L small such that $1 \leq n \leq Lm$. The theory in Sections 3.2 and 3.4 can be used directly by zeroing out the matrices $U_{0|J}$ and $Y_{0|J}$ from the projections.

Hence, we simply obtain the model matrices in the Kalman filter from the projection equations

$$Z_{J|L+1} = Y_{J|L+1} \left[\begin{array}{c} U_{J|L} \\ E_{J|L+1} \end{array} \right]^{\perp} \approx O_{L+1} X^a_J \tag{8.149}$$

which gives O_{L+1}, A and D and

$$Y_{J+1|L} = \tilde{A}_L Y_{J|L} + \left[\begin{array}{cc} \tilde{B}_L & \tilde{C}_L \end{array} \right] \left[\begin{array}{c} U_{J|L} \\ E_{J|L+1} \end{array} \right] \tag{8.150}$$

gives B and K according to the theory in Section 3.4. This algorithm is implemented in the **DSR_e** MATLAB function in the D-SR Toolbox for MATLAB.

8.6 Further remarks

8.6.1 Choice of algorithm parameters

There are two parameters in the algorithm, i.e., L and J. L is interpreted as the identification-horizon used to predict the number of states. J is interpreted as the horizon (into the past) used to define instruments from the data which are used to remove noise. The system order, n, which is specified or identified, is bounded by the user-specified parameter L, i.e. so that, $1 \leq n \leq mL$ where m is the number of outputs. Hence, Lm singular values are computed by the algorithm and the user may chose the system order, n, by inspection of the, n, non-zero singular values.

A rule of thumb is that L should be chosen as small as possible if the inputs are poorly exciting. The minimum identification-horizon, L_{min}, so that the pair (D, A) is observable and rank$(O_{L_{\text{min}}}) = n$ is bounded by, $\lceil \frac{n}{m} \rceil \leq L_{\text{min}} \leq n -$ rank$(D)+1$, where $\lceil \cdot \rceil$ is the ceiling function, i.e., rounding towards plus infinity. If the outputs are independent, then, we suggest to use $L_{\text{min}} = n - m + 1$ when $n \geq m$ and $L_{\text{min}} = 1$ when $n = m$. If the inputs are rich, e.g. white, then this point is not critical. In practice, it is suggested that model validation on independent data is taken into consideration when choosing the "optimal" settings for L.

The past horizon, J, may for combined deterministic and stochastic systems and for pure stochastic systems usually be chosen as $J = L + 1$ or $J = L$. Note that the estimates of C and the Kalman filter gain matrix $K = CF^{-1}$ usually becomes better when J increases. For pure deterministic systems we may chose $J = 1$. The instruments $Y_{0|J}$ and $U_{0|J}$ can also be removed from the projections, i.e., and putting $J = 0$, in this case.

The theory in this paper is the basis for the D-SR Toolbox for MATLAB which are available upon request. The toolbox consists of MATLAB functions for subspace system identification of both open and closed loop systems.

8.6.2 Choice of input signal

The subspace identification methods tends to be more sensitive to the input signal compared to e.g, the Prediction Error Method (PEM). This means that there may exist colored input signals which gives subspace estimates which are as optimal (efficient and consistent) as the PEM estimates. On the other side there may exist colored input signals where the subspace methods gives poorer results compared to the PEM. An optimal experiment for the subspace methods is in general not a white noise input, but rather a colored input signal where the frequency spectrum is optimized to excite the parameters in the system as well as possible. Our experience is also that an input signal which are minimizing the condition number of the Hankal matrix $U_{0|J}$ or $U_{J|L+g}$, is usually not an optimal input signal.

8.6.3 N4SID

The N4SID algorithms in Van Overschee and De Moor (1994) are using an oblique-projection

$$\begin{aligned} \mathcal{O}_i &= Y_{i|i} U_{i|i}^{\perp} W_p^T (W_p U_{i|i}^{\perp} W_p^T)^{-1} W_p, && (8.151) \\ W_p &= \begin{bmatrix} U_{0|i} \\ Y_{0|i} \end{bmatrix}, && (8.152) \end{aligned}$$

for the identification of the extended controllability matrix O_i, i.e., O_i is estimated from the column space of $\mathcal{O}_i$ in (8.151), e.g. using the SVD. Comparing (8.64) with (8.151) shows that the extra projection matrix $U_{i|i}^{\perp}$ are missing on the left hand side of (8.151). Hence, we conclude that in general

$$\mathcal{O}_i \neq O_i X_i^a. \tag{8.153}$$

The consequence of this is that the subspace identification theorems in Van Overschee and De Moor (1994), (1996), (1997) which are using the oblique projection, to our understanding, are wrong.

The extra projection matrix $U_{i|i}^{\perp}$ on the left hand side of (8.151) removes the influence of the future inputs on the future outputs, and is necessary in order to obtain accurate/consistent subspace estimates for colored input signals. Hence, a consistent projection is therefore

$$Z_{i|i} = \mathcal{O}_i U_{i|i}^{\perp} = O_i X_i^a U_{i|i}^{\perp}. \tag{8.154}$$

Hence, the extra projection can not be considered as a weighting matrix but as a projection matrix. One should note that the parameter i used by N4SID is related to the parameter L in DSR as $i = L + 1$. Furthermore, in N4SID the past horizon is put equal to the future horizon. Hence, this corresponds to putting $J = L + 1 = i$ in DSR.

8.7 Numerical examples

8.7.1 Example 1

Given the system (8.1) and (8.2) with the following matrices and vectors

$$A = \begin{bmatrix} 0 & 1 \\ -0.7 & 1.5 \end{bmatrix}, \; B = \begin{bmatrix} 0.25 \\ 0.625 \end{bmatrix}, \; C = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \tag{8.155}$$

$$D = \begin{bmatrix} 1 & 0 \end{bmatrix}, \; E = 1, \; F = 1. \tag{8.156}$$

The following colored input signals where used for identification

$$\begin{aligned} u_k^1 &= \sin(k) + \sin(\frac{k}{2}), && (8.157) \\ u_k^2 &= \sin(k) + \sin(\frac{k}{2}) + \sin(\frac{k}{3}), && (8.158) \\ u_k^3 &= \text{withe noise with variance } \mathrm{E}(u_k^2) = 1. && (8.159) \end{aligned}$$

The number of samples was $N = 1000$. The system was simulated 100 times, each time with the same input but with a different noise realization e_k. However, with the same covariance $\mathrm{E}(e_k^2) = 1$. The DSR parameters where chosen as $L = 2$, $J = 3$. The model structure parameter where $g = 1$. The poles of the 100 identified system matrices are illustrated in Figure 8.4, 8.5 and 8.6. From this we conclude that the method presented in this paper is almost as efficient as the PEM method, for the inputs which are considered. However, the N4SID oblique-method gives an unacceptable bias in the pole estimates for input u_k^1, i.e., because the estimated poles are unstable. From Figure 8.5 we see that the bias is eliminated but the results from N4SID are highly uncertain. However, the results from N4SID are nearly the same as DSR for the white input signal u_k^3. We also see that the variance of the subspace estimates may be smaller for a colored input signal, Figure 8.5, than for the white noise input, Figure 8.6.

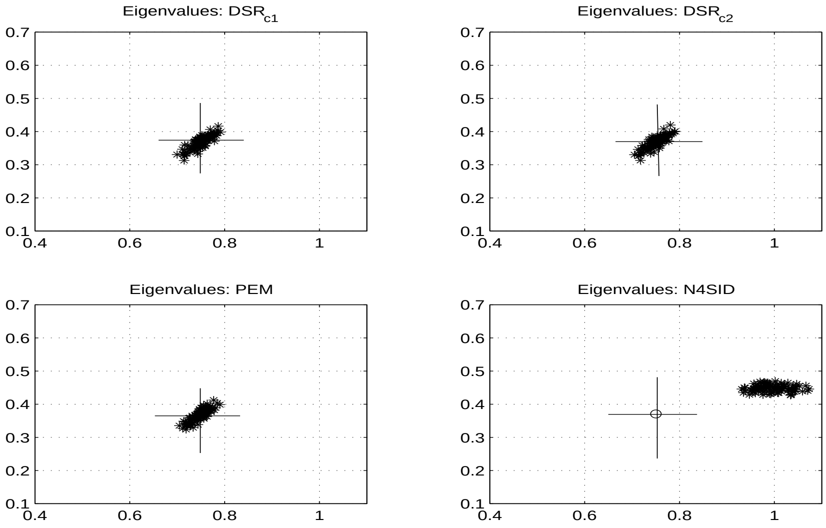


Figure 8.4: Identified poles for a Monte carlo simulation. The exact pole is marked with a cross. Input signal u_k^1 where used.

8.7.2 Example 2

We consider the following system

$$x_{k+1} = Ax_k + Bu_k + Cv_k, \tag{8.160}$$

$$y_k = Dx_k + w_k, \tag{8.161}$$

where the system matrices are the same as in Example 8.7.1. The process noise, v_k, and the measurements noise, w_k, are both white noise with standard deviation $\sqrt{\mathrm{E}(v_k^2)} = \sqrt{0.02} = 0.1458$ and $\sqrt{\mathrm{E}(w_k^2)} = \sqrt{0.002} = 0.0447$, respectively. The system is operating in closed loop. The input to the system is generated by the following discrete time PI-controller

$$u_k = K_p(r_k - y_k) + z_k, \tag{8.162}$$

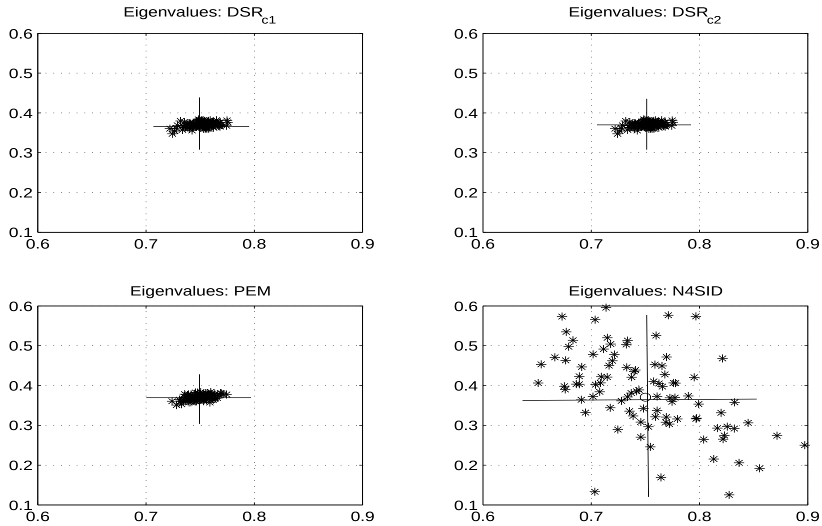


Figure 8.5: Identified poles for a Monte carlo simulation. The exact pole is marked with a cross.Input signal u_k^2 where used.

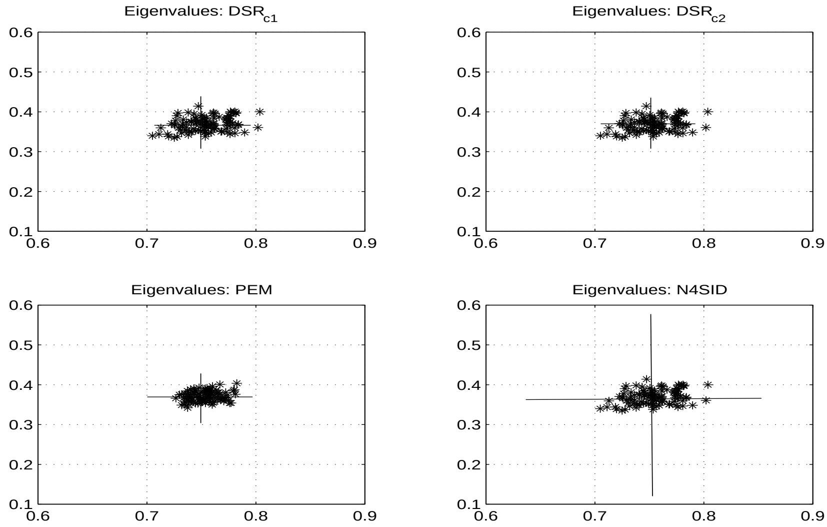


Figure 8.6: Identified poles for a Monte carlo simulation. The exact pole is marked with a cross.Input signal u_k^3 where used in this case.

where the controller state, z_k, is defined by

$$z_{k+1} = z_k + \frac{K_p}{T_i}(r_k - y_k). \tag{8.163}$$

The proportional constant is $K_p = 0.2$, the integral time is $T_i = 5$ and the reference, r_k, is taken as the binary signal in Figure 8.7.

The number of samples was $N = 1000$. The system was simulated 100 times, each time with the same reference, r_k, but with a different noise realizations v_k and w_k, but with the same variance. The DSR parameters where chosen as $L = J = 3$ and the structure parameter where $g = 0$. The subspace algorithms works perfect in the deterministic case. However, the algorithm gives a small bias in the estimates in the case of noise. The bias is negligible for this example. The pole estimates are presented in Figure 8.8.

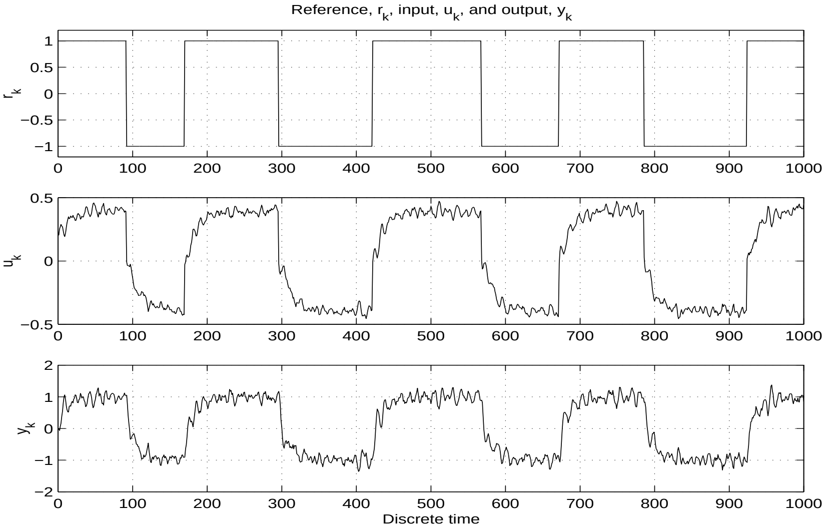


Figure 8.7: The reference signal, r_k, the input, u_k and the output y_k for two particular noise realizations v_k and w_k, used in Example 8.7.2.

8.7.3 Example 3

Consider the same closed loop example as in Example 2. We will in this example illustrate the bias-problem when using subspace identification algorithms directly from input and output data collected in closed loop (Figure 8.1). Furthermore, we will illustrate that the feedback problem can be eliminated by including a low-pass filter in the feedback as in Figure 8.2.

The process noise, v_k, and the measurements noise, w_k, are both white noise with standard deviation $\sqrt{\mathrm{E}(v_k^2)} = \sqrt{0.05} = 0.2236$ and $\sqrt{\mathrm{E}(w_k^2)} = \sqrt{0.01} = 0.1$, respectively. The pole estimates after a Monte carlo simulation is presented in Figure 8.9. We can clearly see a bias in the estimates from the (open loop) subspace identification algorithms. The bias in the DSR estimates is smaller

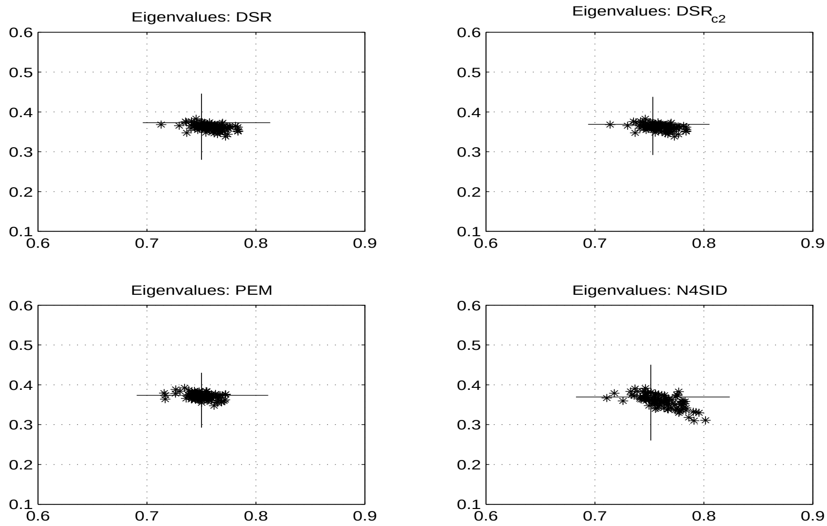


Figure 8.8: The pole estimates from the closed loop data as described in Example 8.7.2.

than the bias in the estimates from both SUBID (Van Overschee and De Moor (1996)) and N4SID. This is also the conclusion from many other simulation experiments.

Consider now the feedback system in Figure 8.2. We are using a PI-controller as in Example 8.7.2 where the filtered output is used as input to the controller. The controller equations are as follows.

$$u_k = K_p(r_k - \bar{y}_k) + z_k, \tag{8.164}$$

where the controller state, z_k, is defined by

$$z_{k+1} = z_k + \frac{K_p}{T_i}(r_k - \bar{y}_k). \tag{8.165}$$

The filter is a 1. order low-pass filter of the form

$$\begin{aligned} \bar{y}_{k+1} &= \bar{y}_k + K_f(y_k - \bar{y}_k) \\ &= (1 - K_f)\bar{y}_k + K_f y_k, \end{aligned} \tag{8.166}$$

with filter constant $K_f = 0.1$. The initial filter output is taken as $\bar{y}_0 = y_0$. Pole estimates after a Monte Carlo simulation is illustrated in Figure 8.10. We see that the pole estimates now are consistent.

8.7.4 Example 4

We will in this example search for an optimal experiment in the reference. Consider the reference $r_k = \sin(\omega k)$ for varying frequency ω. The following investigation shows that the bias in the DSR pole estimates is a function of the frequency and that the bias reach a minimum for a particular frequency.

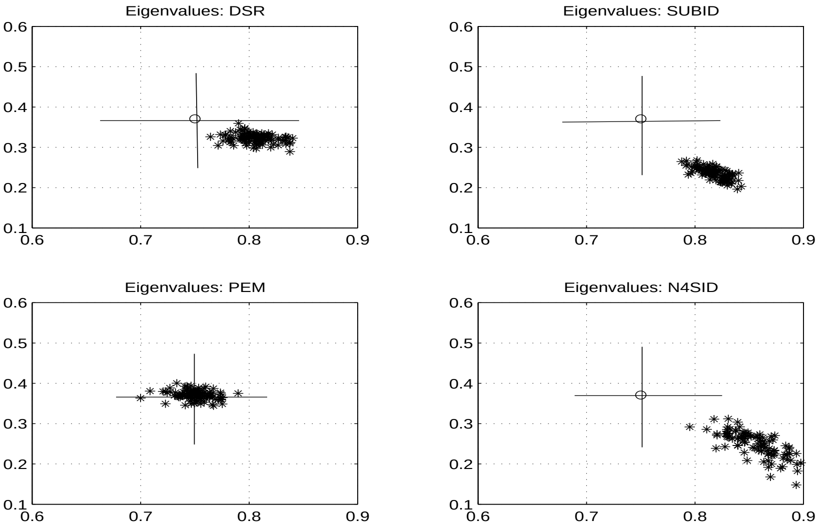


Figure 8.9: The pole estimates from the closed loop data as described in Example 8.7.3. The control system is as in Figure 8.1 with the same reference signal as in Figure 8.7.

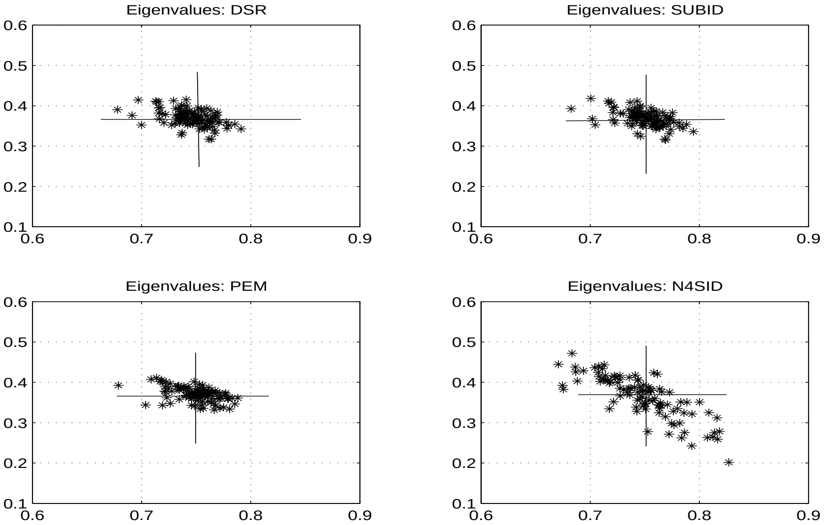


Figure 8.10: The pole estimates from the closed loop data as described in Example 8.7.3 with a filter in the feedback. The control system is as in Figure 8.2.

The reference signal which gave the smallest bias in the pole estimates is for this example found to be for $\omega = \frac{1}{1.526}$, i.e.,

$$r_k = \sin(\frac{1}{1.526}k). \tag{8.167}$$

The pole estimates from an Monte carlo experiment is illustrated in Figure 8.11. The results are very interesting because, as we see, the pole estimates from the DSR subspace identification method is more reliable than the pole estimates from the prediction error method PEM.

The process noise, v_k, and the measurements noise, w_k, are both white noise with standard deviation $\sqrt{\mathrm{E}(v_k^2)} = \sqrt{0.1} = 0.01$ and $\sqrt{\mathrm{E}(w_k^2)} = \sqrt{0.1} = 0.01$, respectively. The DSR parameters is $L = 5$, $g = 0$ and $J = 6$.

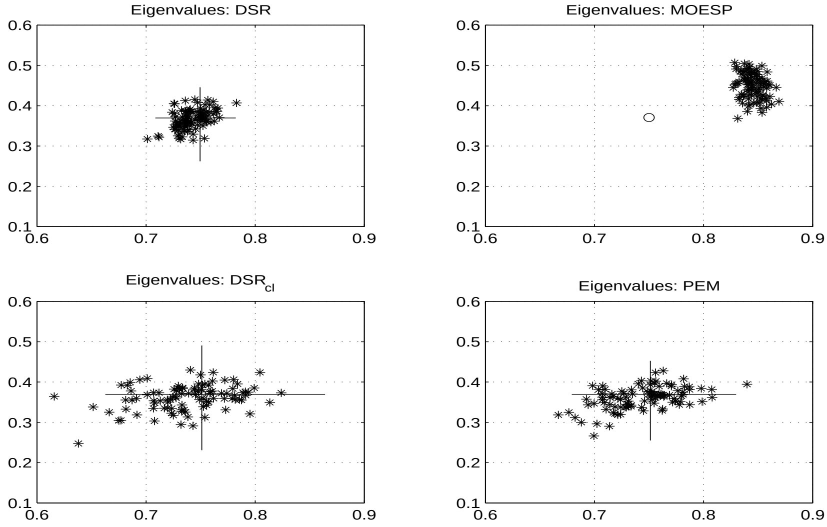


Figure 8.11: The pole estimates from the closed loop data as described in Example 8.7.4 with reference signal $r_k = sin(k/1.526)$. The control system is as in Figure 8.1.

8.7.5 Example 5

Consider the following system

$$x_{k+1} = 0.9x_k + 0.5u_k + 0.6e_k \tag{8.168}$$

$$y_k = x_k + e_k \tag{8.169}$$

with e_k white noise with unit variance. The controller is taken as

$$u_k = K_p(y_k - r_k), \tag{8.170}$$

with $K_p = 0.6$ and a reference signal

$$r_k = \sin(0.5k) + \sin(k). \tag{8.171}$$

A Monte carlo simulation study is performed with $N = 2000$ data points and $M = 100$ different noise realizations. The results are illustrated in Figures 8.12 and 8.13. As we can see, both DSR_e and PEM gives consisten results. It shows also that the subspace method, DSR_e, is as efficient that PEM is. The DSR function results in a smaller bias than MOESP. The parameters $L = 1$ and $J = 6$ was used for the DSR_e function. Parameters $L = 2$ and $J = 6$ for DSR and $i = L + 1 = 3$ for MOESP. The DSR_e function is implemented along the lines in Section 5.

The DSR algorithm gives usually less bias than MOESP for closed loop data, se Figures 8.12 and 8.13. It is very interestin that the DSR_e algorithm gives parameter estimates which are as optimal as the corresponding PEM estimates.

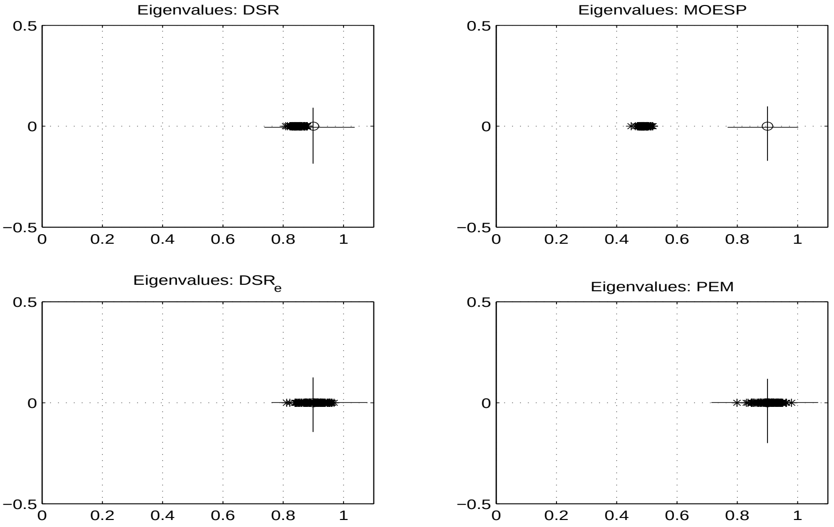


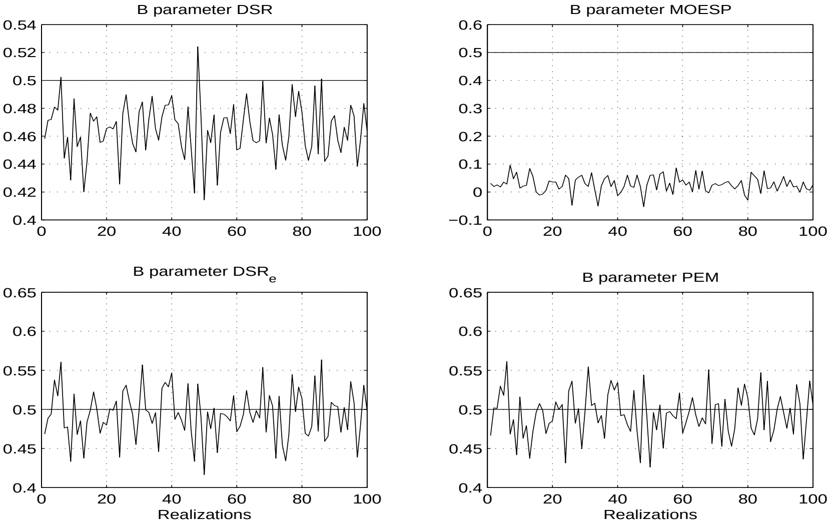
Figure 8.12: The pole estimates from the closed loop data as described in Example 8.7.5 with reference signal $r_k = sin(k) + sin(0.5k)$. The control system is as in Figure 8.1.

8.8 Conclusion

The extended observability matrix O_{L+1} can be computed directly from the column space of a projection matrix $Z_{J|L+1}$, which is defined in terms of the known data. There are in general two projections involved in order to define $Z_{J|L+1}$. One projection is used to remove the effect of noise and one projection is used to remove the effect of future inputs from the future outputs. A necessary condition for a consistent estimate of O_{L+1} is that the number of columns K in the data matrices tends to infinity.

The states are not needed in order to compute the extended observability matrix and, hence, to identify the system dynamics, i.e., the number of states n and the system matrices A and D.

An additional condition for a consistent state estimate is that the past


B parameter DSR

B parameter MOESP

B parameter DSR$_e$

B parameter PEM

Figure 8.13: The estimates of the B parameter from the closed loop data as described in Example 8.7.5 with reference signal $r_k = sin(k) + sin(0.5k)$. The control system is as in Figure 8.1.

horizon J has to tend to infinity. Furthermore, for colored input signals both the extended observability matrix O_L and the lower block triangular Toepliz matrix H_L^d has in general to be known in order to properly computing the states.

The stochastic part of the model, i.e., the Kalman filter gain matrix and the innovations covariance matrix can be identified directly from the data, i.e. from the projection matrix $Z^s_{J|L+1}$, without solving any Riccati or Lyapunov matrix equations.

The deterministic part of the model can be identified from an optimal least squares problem defined from the projection matrix $Z^d_{J|L+1}$, the extended observability matrix O_{L+1}, A and D.

Furthermore the necessary projections, $Z_{J|L+1}$, $Z^d_{J|L+1}$ and $Z^s_{J|L+1}$, which are needed in order to compute a complete state space model realization for the sixtuple matrices (A, B, D, E, C, F) (and/or K and $\Delta = \mathrm{E}(\varepsilon_k \varepsilon_k^T)$), can be computed throug a numerically stable LQ decomposition of $\begin{bmatrix} U_{J|L+g} \\ U_{0|J} \\ Y_{0|J} \\ Y_{J|L+1} \end{bmatrix}$. However, it is in general faster to compute $Z_{J|L+1}$, $Z^d_{J|L+1}$ and $Z^s_{J|L+1}$ directly from the definitions. This means that the algorithm both can be implemented as an correlation based method and a square root based method.

Finally, a method for subspace identification of closed loop systems which gives unbiased estimates is presented. Simulation results shows that the estimates are as efficient as those from the prediction error method, however, the estimates are somewhat dependent of the parameters L and J.

References

Di Ruscio, D. (2003). Subspace system identification of the Kalman filter. *Modeling, Identification and Control. Vol24, No.3, pp. 125-157.*

Di Ruscio, D. (2001). Subspace system identification of the Kalman filter gain. *The European Control Conference, ECC2001, Porto, Portogal 2001.*

Di Ruscio, D. (1997). A method for identification of combined deterministic stochastic systems. In: *Applications of Computer Aided Time Series Modeling*, Lecture Notes in Statistics 119, Eds. M. Aoki and A. M. Havenner, Springer Verlag, ISBN 0-387-94751-5.

Di Ruscio, D. (1997b). On Subspace Identification of the Extended Observability Matrix. In the proceedings of *the 36th Conference on Decision and Control 1997*, San Diego, California, December 6-14.

Di Ruscio, D. (1996). Combined Deterministic and Stochastic System Identification and Realization: DSR-a subspace approach based on observations. *Modeling, Identification and Control*, vol. 17, no.3.

Di Ruscio, D. (1995). A method for identification of combined deterministic stochastic systems: robust implementation. The *Third European Control Conference ECC95*, September 5-8, Rome, Italy.

D-SR Toolbox for MATLAB. Copyright D. Di Ruscio. Available upon request, david.di.ruscio@hit.no.

Larimore, W. E. (1990). Canonical Variate Analysis in Identification, Filtering and Adaptive Control. *Proc. of the 29th Conference on Decision and Control*, Honolulu, Hawaii, December 1990, pp. 596-604.

Kung, S. Y. (1978). A new identification and Model Reduction Algorithm via Singular Value Decomposition. *Conf. on Circuits, Systems and Computers*, Pacific Grove, CA, November 1978, pp. 705-714.

Van Overschee, P. and B. De Moor (1994). N4SID: Subspace Algorithms for the Identification of Combined Deterministic Stochastic Systems. *Automatica*, vol. 30, No. 1, pp.75-94.

Van Overschee, P. and B. De Moor (1996). *Subspace Identification for Linear Systems: theory-implementation-applications.* Kluwer Academic Publishers.

Van Overschee, P. and B. De Moor (1996b). Closed loop subspace system identification. Internal report. Katholieke Universiteit Leuven. Department Elektrotechniek. ESAT-SISTA/TR 1996-521.

Van Overschee, P. and B. De Moor (1997). Closed loop subspace system identification. In the proceedings of *the 36th Conference on Decision and Control 1997*, San Diego, California, December 6-14.

Verhagen, M. (1994). Identification of the deterministic part of MIMO state space models given on innovations form from input output data. *Automatica*, vol. 30, No. 1, pp. 61-74.

Viberg, M. (1995). Subspace-Based Methods for the Identification of Linear Time-invariant Systems. *Automatica*, vol. 31, No. 12, pp. 1835-1851.

Chapter 9

Effect of Scaling and how to Treat Trends

9.1 The data

We will consider some system input and output data vectors

$$\left.\begin{array}{lll} u_k & \forall & k=1,\cdots,N \\ y_k & \forall & k=1,\cdots,N \end{array}\right\} \text{Known system input and output data} \tag{9.1}$$

We will assume that the data can be modeled by a linear discrete time combined deterministic and stochastic state space model. The data does not necessarily have to be stationary. Constant trends or drifts is represented with non-stationary series which can be modeled by a state space model with poles on the unit circle.

9.2 The data matrices

From the known input and output series (9.1) we define the data matrices as follows

$$\overbrace{Y = \begin{bmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_N^T \end{bmatrix} \in \mathbb{R}^{N\times m}}^{\textbf{Known} \text{ data matrix of output variables}} \tag{9.2}$$

$$\overbrace{U = \begin{bmatrix} u_1^T \\ u_2^T \\ \vdots \\ u_N^T \end{bmatrix} \in \mathbb{R}^{N\times r}}^{\textbf{Known} \text{ data matrix of input variables}} \tag{9.3}$$

9.3 Scaling the data matrices

Consider the scaled system input and output data

$$\left.\begin{array}{l} u_k^s = S_u u_k \quad \forall \quad k = 1, \cdots, N \\ y_k^s = S_y y_k \quad \forall \quad k = 1, \cdots, N \end{array}\right\} \text{Known scaled input and output data.} \tag{9.4}$$

The scaled input and output data vectors can be organized into data-matrices similarly as in (9.2) and (9.3). However, the matrices which are identified should be "unscaled" in order to satisfy the underlying (unscaled) system. A procedure for incorporating scaling matrices is presented in Algorithm 9.3.1.

Algorithm 9.3.1 (Modeling procedure with scaled data matrices)
Step 1, scaling the data: From the data matrices, Equations (9.2) and (9.3), and some scaling matrices S_y and S_u, we define the scaled data matrices

$$Y^s = Y S_y^T, \tag{9.5}$$

$$U^s = U S_u^T. \tag{9.6}$$

Step 2, identification: The problem of identifying a complete linear (usually) dynamic model for the process can be solved by the DSR function (see *D-SR Toolbox for Matlab*).

$$\left[\ A,\, B^s,\, D^s,\, E^s,\, C,\, F^s\ \right] = \mathrm{DSR}(Y^s, U^s, L), \tag{9.7}$$

where L is a positive scalar.

Step 3, unscaling: The matrices

$$B = B^s S_u, \quad D = S_y^{-1} D^s, \quad E = S_y^{-1} E^s S_u, \quad F = S_y^{-1} F^s, \tag{9.8}$$

satisfy the SSM

$$x_{k+1} = A x_k + B u_k + C e_k^s, \tag{9.9}$$

$$y_k = D x_k + E u_k + F e_k^s, \tag{9.10}$$

where e_k^s has unit covariance.

Step 4, Kalman filter: The innovations form of the model (Kalman filter) is given by

$$x_{k+1} = A x_k + B u_k + K e_k, \tag{9.11}$$

$$y_k = D x_k + E u_k + e_k, \tag{9.12}$$

where K is the Kalman filter gain

$$K = C F^{-1}. \tag{9.13}$$

e_k is the innovations with covariance matrix

$$\mathrm{E}(e_k e_k^T) = F F^T. \tag{9.14}$$

△

Remark 9.1 One particular and common choice is to chose the scaling matrices S_y and S_u so that each column (variable) in the data-matrices Y^s and U^s has unit variance.

9.3.1 Proof of Step 3 in Algorithm 9.3.1

The sixfold matrices $(A, B^s, C, D^s, E^s, F^s)$ are the scaled state space model matrices (in the state space model)

$$\begin{aligned} x_{k+1} &= Ax_k + B^s u_k^s + Ce_k^s, && (9.15)\\ y_k^s &= D^s x_k + E^s u_k^s + F^s e_k^s, && (9.16) \end{aligned}$$

where e_k^s has unit covariance matrix, i.e., $\mathrm{E}(e_k^s(e_k^s)^T) = I$. Substituting for y_k^s and u_k^s given by (9.4) we get

$$\begin{aligned} x_{k+1} &= Ax_k + \overbrace{B^s S_u}^{B} u_k + Ce_k^s, && (9.17)\\ S_y y_k &= D^s x_k + E^s S_u u_k + F^s e_k^s. && (9.18) \end{aligned}$$

Assume that S_y is non-singular. Post multiply Equation (9.18) with S_y^{-1} we get

$$y_k = \underbrace{S_y^{-1} D^s}_{D} x_k + \underbrace{S_y^{-1} E^s S_u}_{E} u_k + \underbrace{S_y^{-1} F^s}_{F} e_k^s. \tag{9.19}$$

Hence, we have

$$B = B^s S_u, \quad D = S_y^{-1} D^s, \quad E = S_y^{-1} E^s S_u, \quad F = S_y^{-1} F^s. \tag{9.20}$$

To write the model on innovations (Kalman filter) form we define (from the noise term in Equation (9.19))

$$e_k = S_y^{-1} F^s e_k^s, \tag{9.21}$$

where e_k is the innovations with covariance matrix

$$\mathrm{E}(e_k e_k^T) = S_y^{-1} F^s (F^s)^T (S_y^{-1})^T, \tag{9.22}$$

because e_k^s has unit covariance.

From Equation (9.21) we get

$$e_k^s = (F^s)^{-1} S_y e_k. \tag{9.23}$$

Substituting for e_k^s into equations (9.17) and (9.19) we get the innovations form of the state space model

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + \overbrace{C(F^s)^{-1} S_y}^{K} e_k, && (9.24)\\ y_k &= Dx_k + Eu_k + e_k, && (9.25) \end{aligned}$$

where the Kalman filter gain matrix K is overbraced. **QED**.

9.3.2 Numerical Examples

Example 9.1 (Monte Carlo simulation)
Monte Carlo simulations of a MIMO system with two inputs ($r = 2$)*, two outputs* ($m = 2$) *and three states* ($n = 3$) *was worked out in order to analyze the effect of scaling the input and output data on the pole estimates. Note that, in this case, the minimal identification parameter* L *is* $L_{min} = n - m + 1 = 2$ *if the two output variables are independent. The output variables are independent in this example. The DSR parameter* L *was varyed from* $L = 2$ *to* $L = 6$*.*

Conclusions drawn when scaled data was used:

- *Scaling the inputs gave no effect upon the pole estimates. The statistical distribution of the pole estimates was unchanged.*
- *In general, output scaling of the type* $Y^s = Y S_y^T$ *where* S_y *is a general non-singular matrix, can destroy the statistical distribution of the pole estimates. However, this was only observed when* $L > L_{min}$ *where* $L_{min} = 2$*. The statistical distribution seams to depend on the chice of scaling. A question is, does there exist an optimal scaling matrix* S_y *which should be used when* $L > L_{min}$*.*
- *Irrespective of scaling the output variables, the statistical distribution of the pole estimates was unchanged when the minimal parameter* $L = L_{min} = 2$ *was used.*
- *Irrespective of choice of* L*, the statistical distribution of the pole estimates was unchanged when each column (variable) in* Y *was equally scaled, i.e., for scaling matrices* $S_y = diag(s, \cdots, s)$ *where* s *is a non-zero scalar.*

Conclusions drawn when trended data was used:

- *The system order increased by one when dtrended data (i.e., data with constant trends removed) was used. A unit pole is included in the system in order to handle the trends.*

9.4 How to handle trends and drifts

Time series usually has a trend. Such trends can be nonzero constants or mean values, low frequency noise or drifts. The trends are often aprori unknown and time varying. Assume that the output y_k from a system can be separated into three parts. One deterministic part y_k^d which is driven from the known inputs u_k, one stochastic part y_k^s driven from unknown inputs or disturbances and one part y_k^0 which represents the trend.

$$y_k = y_k^d + y_k^s + y_k^0. \tag{9.26}$$

In some applications we do not distinguish between the stochastic part y_k^s and the trend y_k^0, i.e. we treat them simultaneously. The deterministic part, the stochastic part and the trend may have common as well as separate and different dynamics.

Constant non-zero trends and drifts can be described by a state space model which has a pole on the unit circle. Low-frequency noise can be characterized by a state space model which has a pole close to the unit circle. Such trends can approximately be described by a random walk (drift)

$$x_{k+1}^0 = x_k^0 + v_k, \tag{9.27}$$

$$y_k^0 = x_k^0, \tag{9.28}$$

where v_k is assumed to be a white and Gaussian distributed disturbance. This is a non-stationary process because the system has a pole on the unit circle. Trends can often be described by non-stationary processes which has a pole on the unit circle.

Assume that the trend is purely time invariant (constant) and equal to y^0, e.g., y^0 can be the sample mean of the series y_k. The trend can then be described by a 1st order state space model with a pole equal to one (integrator)

$$x_{k+1}^0 = x_k^0, \quad x_1^0 = y^0, \tag{9.29}$$

$$y_k^0 = x_k^0, \tag{9.30}$$

where v_k is assumed to be Gaussian distributed. The output y_k^0 from this system is purely exited by the initial values y^0. The initial values which are identified are important because wrong initialization is not forgotten (because the system is equal (or close) to an integrator).

It is not necessary to remove trends from the data when using DSR. Nonzero constant trends or drifts are usually identified and represented with low frequency dynamics, i.e. poles close to the unit circle.

One simple strategy is to first remove some constant trend from the data and then identify the state space model matrices with initial values as follows,

$$\left[\; A,\, B,\, D,\, E,\, C,\, F, x_0 \;\right] = \mathrm{DSR}(Y, U, L),$$

where Y and U are the data matrices adjusted for constant trends.

The constant trends which are removed can be the sample mean, the values of the series at time zero $k = 1$ or the mean of the first, say, j samples of the series. A simple strategy for identifying constant trends , which are frequently used in practice, are given by

$$u^0 = \frac{1}{j}\sum_{k=1}^{j} u_k, \quad 1 \leq j \leq N, \tag{9.31}$$

$$y^0 = \frac{1}{j}\sum_{k=1}^{j} y_k, \quad 1 \leq j \leq N. \tag{9.32}$$

Nevertheless, even if this strategy is used, it is important to note that it can be low-frequency dynamics in the estimated state space model. It is important to note that this low-frequency dynamics often is incorporated in order to represent trends or drifts (which are not properly identified) in the output time series, it can be decoupled from the dynamics from the known inputs u_k to the outputs y_k. In the following this is illustrated with some examples.

Example 9.2 (Using trended data)
Assume a purely steady state deterministic system given by

$$y_k = Eu_k. \tag{9.33}$$

where $E \in \mathbb{R}^{m\times r}$ is the gain matrix of the system.

Assume that a sequence of N input and output samples are given. We can then define the data matrices Y and U. (Note the relationship $Y = UE^T$).

Let us use a trended output data matrix for identification, i.e.,

$$Y^s = Y - Y^0, \tag{9.34}$$

where Y^0 is the trend (sample mean).

Using DSR for identification, i.e., $\begin{bmatrix} A, & B, & D, & E, & C, & F, x_0 \end{bmatrix} = DSR(Y^s, U, L)$, we find the following 1st order state space model

$$x_{k+1} = x_k, \qquad x_1 = x_0, \tag{9.35}$$

$$\hat{y}_k^s = Dx_k + Eu_k. \tag{9.36}$$

The estimated system has a unit pole. The reason for the unit pole is to handle the trend. This means that both the trend $Dx_k = y^0$ as well as the gain matrix E are identified. We have that $\hat{y}_k^s - Dx_k = y_k$.

Example 9.3 (Non-stationary process)
Consider the system

$$y_k = eu_k + y_k^s, \tag{9.37}$$

The output y_k from this system consists of two parts. One purely steady state deterministic part eu_k where $e = -1$ and one stochastic trend y_k^s.

The trend is a drift which is described by

$$x_{k+1} = x_k + v_k, \tag{9.38}$$

$$y_k^s = x_k + w_k, \tag{9.39}$$

where v_k and w_k is serially uncorrelated and Gaussian distributed with unit variance. The system (9.38) is a so called "random walk" which has a pole on the unit circle ($a = 1$). The process is therefore not stationary.

The input to the system is exactly known and described by the random walk

$$z_{k+1} = z_k + \mu_k, \tag{9.40}$$

$$u_k = z_k, \tag{9.41}$$

where μ_k is Gaussian distributed with unit variance and serially uncorrelated with v_k and w_k.

A Monte Carlo simulation of the system was done. I.e., the system (9.37)-(9.39) was simulated $M = 500$ times, each time with new realizations for v_k and w_k but with the same input (9.41). The number of samples for each simulation was $N = 5000$,

500 *different 1st order state space models was estimated from the* 500 *sequences of input and output data (u_k, y_k, $\forall$, $k = 1, \cdots, N$).*

The innovations model for the system is

$$x_{k+1} = ax_k + bu_k + ce_k, \tag{9.42}$$

$$\hat{y}_k = x_k + eu_k + e_k. \tag{9.43}$$

where $a = 1$, $b = 0$ and $e = -1$. The "exact" Kalman filter gain matrix is $c = 0.6180$ and the variance of the innovations is $E(e_k^2) = 2.6180$.

The mean and standard deviation of the 500 *estimated set of parameters (for the innovations model) are given by*

$$\left.\begin{array}{ll} \hat{a} = 0.9994 \pm 0.0007, \quad \hat{b} = 0.0000 \pm 0.0009, & \hat{e} = -1.0017 \pm 0.0220, \\ \hat{c} = 0.6188 \pm 0.0132, \quad E(e_k^2) = 2.6083 \pm 0.0532. & \end{array}\right\} \tag{9.44}$$

Note that the estimated model parameters was scaled so that $d = 1$. The parameters was estimated by DSR with identification horizon $L = 2$ (used for identification of the number of states) and past horizon $J = 5$ (used to define instruments).

For comparison a purely steady state algorithm was used. The mean and standard deviation of the gain estimates from PLS was

$$\hat{e} = -0.9891 \pm 1.2538 \;\} \textit{ PLS with raw data} \tag{9.45}$$

$$\hat{e} = -0.9758 \pm 0.7409 \;\} \textit{ PLS with centered data} \tag{9.46}$$

The mean of the parameters is not to bad. However, the standard deviation is large which makes the estimates unreal-able.

9.5 Trends and low frequency dynamics in the data

There are basically two different approaches to deal with trends and slow disturbances in the data:

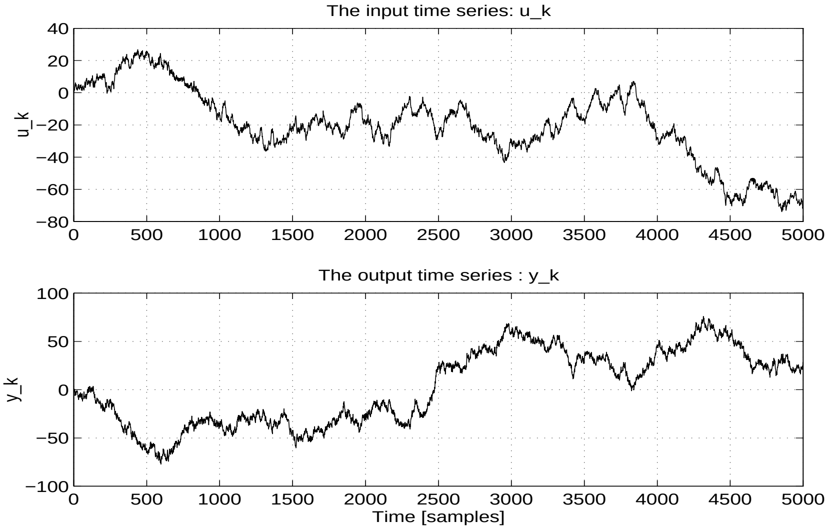


Figure 9.1: This figure shows the input used in Example 9.3 as well as the output (for one particular noise realization).

- **Removing the disturbances by explicit pretreatment of the data**. Assume that input and output trends u_t^0 and y_t^0 are given. A model is then identified based on the input and output data which are adjusted for the trends, i.e.

$$dy_t = y_t - y_t^0, \tag{9.47}$$

$$du_t = u_t - u_t^0. \tag{9.48}$$

- **Letting a noise model take care of the disturbances**. In this case the offset or trends are estimated as part of the modeling procedure. Some additional states are included in order to identify the trends.

It is important to note that the trends must satisfy the model (or system). This means that the trend either must be an equilibrium point or a steady state point.

9.6 Time varying trends

Considder the data u_t and y_t wich satisfy the system

$$x_{t+1} = Ax_t + Bu_t + Ce_t, \tag{9.49}$$

$$y_t = Dx_t + Eu_t + Fe_t, \tag{9.50}$$

and some trends u_t^0 and y_t^0 which also satisfy the system

$$x_{t+1}^0 = Ax_t^0 + Bu_t^0, \tag{9.51}$$

$$y_t^0 = Dx_t^0 + Eu_t^0. \tag{9.52}$$

In this case the deviations $du_t^0 = u_t - u_t^0$ and $dy_t^0 = y_t - y_t^0$ will also satisfy the system, i.e.

$$\begin{aligned} x_{t+1} - x_{t+1}^0 &= A(x_t - x_t^0) + B(u_t - u_t^0) + Ce_t, && (9.53) \\ y_t - y_t^0 &= A(x_t - x_t^0) + E(u_t - u_t^0) + Fe_t, && (9.54) \end{aligned}$$

This means that the adjusted data du_t and dy_t satisfy the system model (A, B, D, E, CF, F) with states $dx_t = x_t - x_t^0$.

9.7 Constant trends

Assume that the data are adjusted for some constant trends or working points u^0 and y^0. It is important to note that such trends must satisfy the steady state (static) relationship of the system, i.e.

$$y^0 = H^d u^0 \qquad (9.55)$$

where H^d is the steady state deterministic gain matrix

$$H^d = D(I - A)^{-1} + E. \qquad (9.56)$$

The data adjusted for the constant trends

$$\begin{aligned} dy_t &= y_t - y^0, && (9.57) \\ du_t &= u_t - u^0, && (9.58) \end{aligned}$$

will also satisfy the system given by (A, B, D, E, C, F). This can relatively easy be prooved in the frequency domain.

An approximation of these steady state trends wich are commonly used in practice are the sample mean

$$\begin{aligned} y^0 &= \frac{1}{N}\sum_{t=1}^{N} y_t, && (9.59) \\ u^0 &= \frac{1}{N}\sum_{t=1}^{N} u_t. && (9.60) \end{aligned}$$

Another strategy which are found from practical experiences to work well in most cases is to identify the working point from the mean of the first few samples.

Chapter 10

Validation

10.1 Model validation and fit

In order to measure the quality of a model we have to define some criteria. Consider a multivariable system with m output variables stacked on each other in the output vector y_t and r input variables stacked on each other in the input vector u_t.

We will in the following assume that we have a set of input and output time series (or observations)

$$\left.\begin{array}{c} u_t \\ y_t \end{array}\right\} \quad \forall \quad t = 1, \cdots, N \tag{10.1}$$

for model validation.

10.1.1 Criteria based on the simulated error

Assume that a deterministic model (i.e. some model matrices (A, B, D, E) and a set of input and output validation data, as defined in (10.1), are given. Simulation will then give us the *simulated outputs*

$$x_{t+1}^d = Ax_t^d + Bu_t \tag{10.2}$$

$$\hat{y}_t^d = Dx_t^d + Eu_t \tag{10.3}$$

where the initial state vector x_1^d is known/specified [1]

The difference between each actual output and each *simulated output* can be measured by the following criteria.

Definition 10.1 (Mean Square Error (MSE))
Define $[y_o]_t$ as (validation) output channel/number o and $[\hat{y}_o^d]_t$ for the simulated

[1] Note that the super-script ^{d} stands for *deterministic* and that the actual state vector x_t can be splitted into two parts, i.e., deterministic and stochastic states, satisfying $x_t = x_t^d + x_t^s$.

output number/channel o. The Mean Square Error for output number o (MSE_o), i.e. the mean square error between the output number o and the simulated output number o is defined as

$$MSE_o = \frac{1}{N}\sum_{t=1}^{N}([y_o]_t - [\hat{y}_o^d]_t)^2 \; \forall \; o = 1, \cdots, m \tag{10.4}$$

where N is the number of observations (samples). This gives a vector of MSE for the m output channels, i.e.

$$MSE = \begin{bmatrix} MSE_1 \\ \vdots \\ MSE_m \end{bmatrix} \in \; \mathbb{R}^m \tag{10.5}$$

Definition 10.2 (Relative Mean Square Error (MSER))
Define $[y_o]_t$ as (validation) output channel/number o and $[\hat{y}_o^d]_t$ for the simulated output channel/number o. The Relative Mean Square Error for output number o ($MSER_o$), i.e. the Mean Square Error between the output number o and the simulated output number o Relative to the "energy" in the signal $[y_o]_t$, is defined as follows

$$MSER_o = \frac{\frac{1}{N}\sum_{t=1}^{N}([y_o]_t - [\hat{y}_o^d]_t)^2}{\frac{1}{N}\sum_{t=1}^{N}[y_o]_t^2} = \frac{MSE_o}{\frac{1}{N}\sum_{t=1}^{N}[y_o]_t^2} \; \forall \; o = 1, \cdots, m \tag{10.6}$$

where N is the number of observations (samples). This gives a vector of MSER defined from the m output channels, i.e.

$$MSER = \begin{bmatrix} MSER_1 \\ \vdots \\ MSER_m \end{bmatrix} \in \; \mathbb{R}^m \tag{10.7}$$

10.1.2 Criteria based on the prediction error

Assume that a combined deterministic and stochastic model (i.e. some model matrices (A, B, D, E and Kalman filter gain matrix F) and a set of input and output validation data, as defined in (10.1), are given. Simulation will then give us the *optimal predictions* as follows

$$x_{t+1} = Ax_t + Bu_t + K(y_t - Dx_t - Eu_t) \tag{10.8}$$

$$\hat{y}_t = Dx_t + Eu_t \tag{10.9}$$

where the initial state vector x_1 is known/specified.

Chapter 11

Input experiment design

11.1 Experiment design for dynamic systems

Input experiment design for dynamic systems is discussed in , among others, Goodwin and Payne (1972) and Ljung (1989). We will in the following focus on Binary input Signals (BS) and Pseudo Random Binary input Signals (PRBS) and on some simple criteria for measuring the quality of different experiment design.

Assume that an input signal series

$$u_t \;\in\; \mathbb{R}^r \;\;\forall\; 1 \leq t \leq N \tag{11.1}$$

is given. From the input series (11.1) define the following input data matrix with $n+g$ block rows and $K = N - n - k$ block columns.

$$U_{k|n+g} \overset{\text{def}}{=} \overbrace{\begin{bmatrix} u_k & u_{k+1} & u_{k+2} & \cdots & u_{k+K-1} \\ u_{k+1} & u_{k+2} & u_{k+3} & \cdots & u_{k+K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{k+n+g-2} & u_{k+n+g-1} & u_{k+n+g} & \cdots & u_{k+n+K+g-3} \\ u_{k+n+g-1} & u_{k+n+g} & u_{k+n+g+1} & \cdots & u_{k+n+K+g-2} \end{bmatrix}}^{\textbf{Known}\text{ data matrix of input variables}} \in \mathbb{R}^{(n+g)r\times K} \tag{11.2}$$

Definition 11.1 (Excitation condition)
The input signal u_t defined by (11.1) is defined to be exciting of order n if and only if the matrix $U_{k|n+g}$ is non-singular, i.e.

$$rank(U_{k|n+g}) = (n+g)r \tag{11.3}$$

where g is a prescribed model structure parameter with values $g = 0$ if $E = 0_{m\times r}$ and $g = 1$ if $E \neq 0_{m\times r}$.

Definition 11.2 (Excitation condition)
The input signal u_t defined by (11.1) is defined to be exciting of order n if and

only if the matrix

$$P_n = \frac{1}{K} U_{k|n+g} U_{k|n+g}^T \in \mathbb{R}^{(n+g)r \times (n+g)r} \tag{11.4}$$

is non-singular.

A simple measure of the quality of the input signal (11.1) is the condition number of P_n or $U_{k|n+g}$. An optimal input design is an input signal with minimum condition number ($\text{cond}(U_{n|n+g})$ or $\text{cond}(P_n)$) subject to some constraints.

Following Ljung (1987) we have the following definition of a *persistent exiting* input signal

Definition 11.3 (Persistent excitation)
The input signal u_t defined by (11.1) is defined to be persistently exciting of order n if and only if the matrix

$$\lim_{K\to\infty} P_n \in \mathbb{R}^{(n+g)r \times (n+g)r} \tag{11.5}$$

is non-singular.

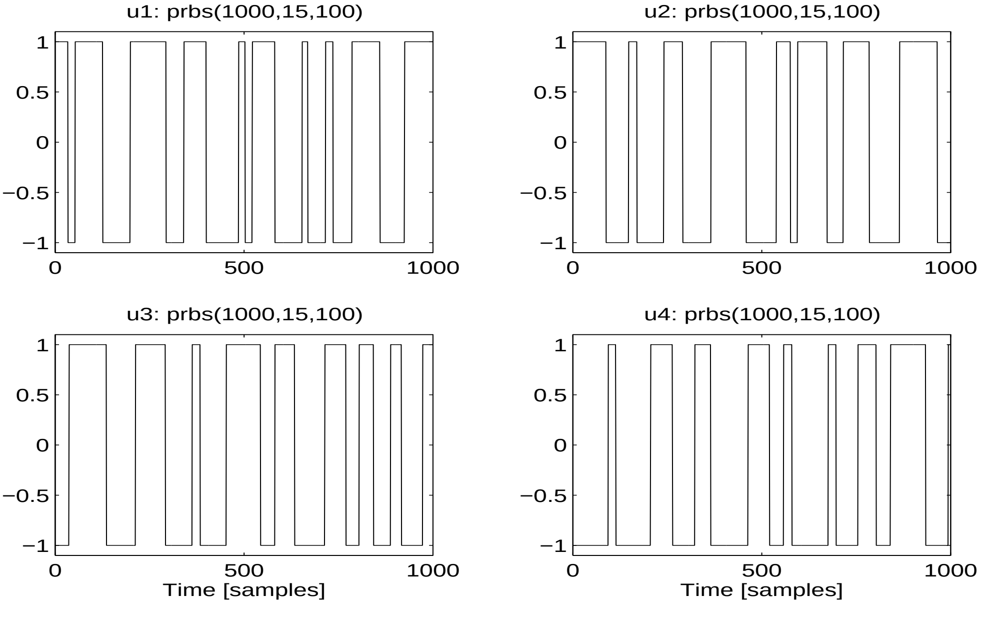


Figure 11.1: Illustration of a Pseudo Random Binary Signal (PRBS) experiment design. The input signal series are designed by four separate calls to the MATLAB function PRBS1.

Example 11.1 (optimal design)
Consider the problem of designing a binary single input experiment signal

$$u_t \ \forall \ 1 \leq t \leq N \tag{11.6}$$

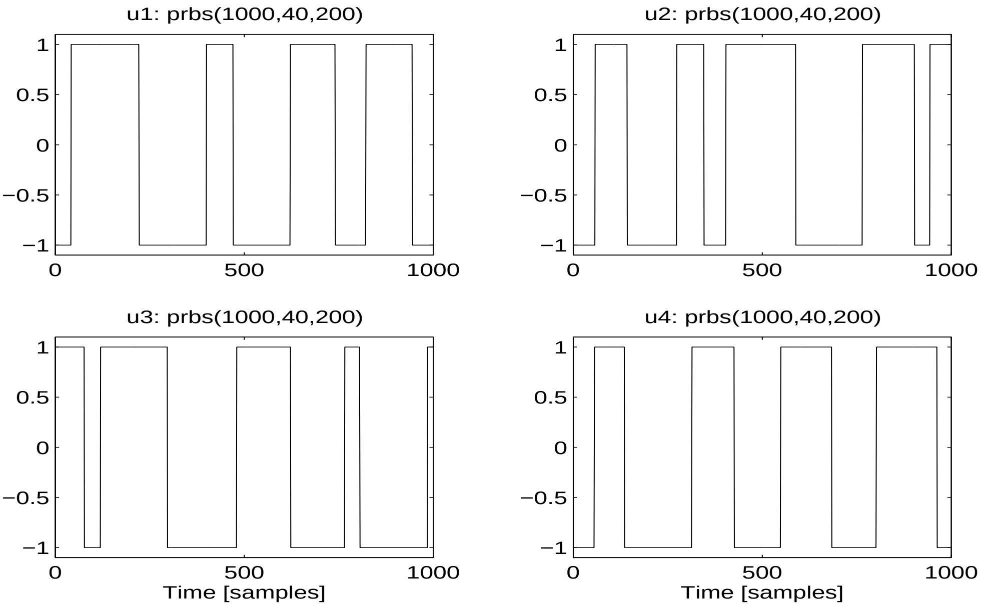


Figure 11.2: Illustration of a Pseudo Random Binary Signal (PRBS) experiment design. The input signal series are designed by four separate calls to the MATLAB function PRBS1.

with the following data

$N = 15$	*Total number of samples in* u_t
$N_T = 3$	*Total number of intervals in* u_t
$T_{min} = 2$	*Minimum sample interval*
$T_{max} = 5$	*Minimum sample interval*
$u_t^{max} = 1$	*Maximum input amplitude*
$u_t^{min} = -1$	*Minimum input amplitude*

The $N_T = 3$ *sample intervals* T_1*,* T_2 *and* T_3 *satisfy the constraint*

$$T_1 + T_2 + T_3 = N. \tag{11.7}$$

This gives only a number of two independent discrete parameters, say T_1 *and* T_2*, because* T_3 *can be computed as* $T_3 = N - T_1 - T_2$*.*

It make sense to find those integer parameters T_1 *and* T_2 *for which the condition number of* $U_{k|(n+g)}$ *is minimized. This is an integer variable optimization problem.*

The parameters T_1 *and* T_2 *is bounded from up by* $T_{max} = 5$ *and from below by* $T_{min} = 2$*. Hence,* T_1 *and* T_2 *can only have the discrete (integer) values* $2, 3, 4$ *or* 5*. This gives a total number of* $M_C = 4 \cdot 4 = 16$ *different combinations.*

The condition numbers of the matrices $U_{1|n}$ *and* P_n *are presented in the following tables for system orders* $n = 5$ *and* $n = 6$ *and for all possibile experiment design.*

Design #	T_1	T_2	T_3	*cond*(P_5)	*cond*$(U_{1\|5})$	*evaluation*
1	*2*	*2*	*11*	74.729	8.645	
2	*2*	*3*	*10*	42.695	6.534	
3	*2*	*4*	*9*	22.123	4.704	
4	*2*	*5*	*8*	22.123	4.704	
5	*3*	*2*	*10*	44.488	6.670	
6	*3*	*3*	*9*	34.487	5.873	
7	*3*	*4*	*8*	15.156	3.893	
8	*3*	*5*	*7*	15.156	3.893	
9	*4*	*2*	*9*	22.407	4.734	
10	*4*	*3*	*8*	12.159	3.487	
11	*4*	*4*	*7*	10.472	3.236	*minimum*
12	*4*	*5*	*6*	10.472	3.236	*minimum*
13	*5*	*2*	*8*	22.407	4.734	
14	*5*	*3*	*7*	12.159	3.487	
15	*5*	*4*	*6*	10.472	3.236	*minimum*
16	*5*	*5*	*5*	10.472	3.236	*minimum*

Design #	T_1	T_2	T_3	*cond*(P_6)	*cond*$(U_{1\|6})$	*evaluation*
1	*2*	*2*	*11*	∞	∞	
2	*2*	*3*	*10*	56.9469	7.5463	
3	*2*	*4*	*9*	42.1114	6.4893	
4	*2*	*5*	*8*	24.1949	4.9188	
5	*3*	*2*	*10*	109.2796	10.4537	
6	*3*	*3*	*9*	47.8914	6.9204	
7	*3*	*4*	*8*	41.2023	6.4189	
8	*3*	*5*	*7*	24.1949	4.9188	
9	*4*	*2*	*9*	81.4547	9.0252	
10	*4*	*3*	*8*	32.6288	5.7122	
11	*4*	*4*	*7*	44.8400	6.6963	
12	*4*	*5*	*6*	∞	∞	
13	*5*	*2*	*8*	26.4365	5.1416	
14	*5*	*3*	*7*	17.6257	4.1983	*minimum*
15	*5*	*4*	*6*	24.1949	4.9188	

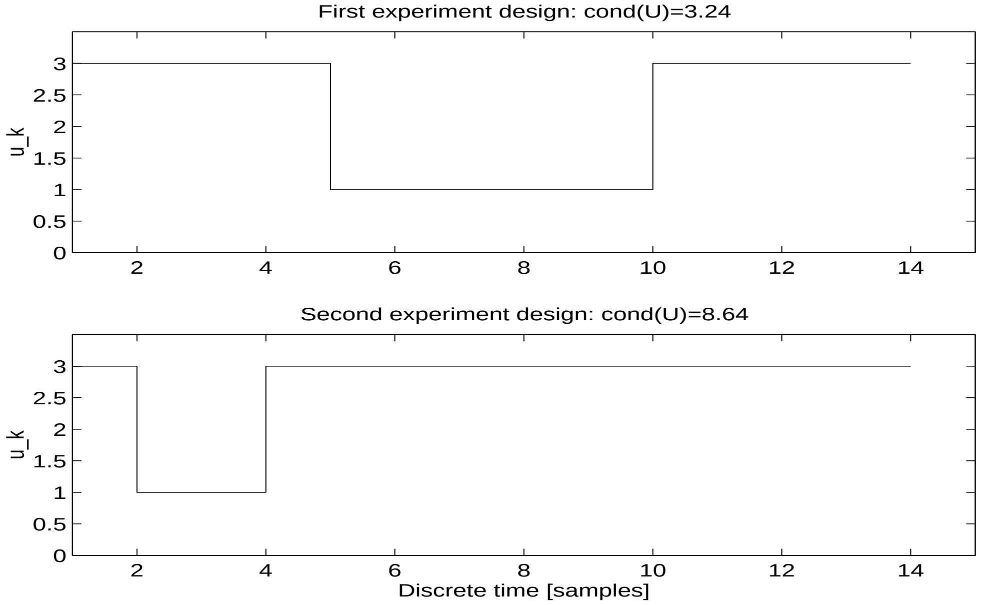


Figure 11.3: Illustration of two different binary input signal design. The design in the upper part has a condition number cond$(U_{1|5}) = 3.236$ and the design in the lower part cond$(U_{1|5}) = 8.645$. The input design in the upper part is better conditioned for identification of a 5th ($n = 5$) order dynamic model compared to the input design in the lower part. See example 11.1 for details.

Chapter 12

Special topics in system identification

12.1 Optimal predictions

For some time series we observe that the system dynamics are close to an integrator and in particular for stochastic single output systems we may find that the A parameter is close to one. In such cases we also often find that the optimal Kalman filter prediction, $\bar{y}_k$, is approximately equal to the previous output y_{k-1}. The reason for this is presented in the following proposition

Proposition 12.1 (Prediction equal to previous output)
Consider a Kalman filter for a stochastic system $x_{k+1} = Ax_k + v_k$ and $y_k = Dx_k + w_k$, i.e.,

$$\begin{aligned} \bar{x}_{k+1} &= A\bar{x}_k + K(y_k - \bar{y}_k), && (12.1)\\ \bar{y}_k &= D\bar{x}_k. && (12.2) \end{aligned}$$

Assume that $A = 1$ (or A close to one) and $DK = 1$ (or DK close to one). In this case we have that the optimal prediction, $\bar{y}_k$, at present time k is equal to the previous output y_{k-1}, i.e.,

$$\bar{y}_k = y_{k-1}. \qquad (12.3)$$

Proof 12.1
From (12.1) and (12.2) with $A = I$ we have that

$$\begin{aligned} \Delta\bar{x}_k &= \bar{x}_k - \bar{x}_{k-1} = K(y_{k-1} - \bar{y}_{k-1}), && (12.4)\\ \bar{y}_k &= y_{k-1} + D\Delta\bar{x}_k && (12.5) \end{aligned}$$

This gives

$$\bar{y}_k = \bar{y}_{k-1} + DK(y_{k-1} - \bar{y}_{k-1}), \qquad (12.6)$$

which gives Equation (12.3) when $DK = 1$. **QED**

The Kalman filter is constructed to minimize the covariance of the estimation error, i.e. the parameters in the Kalman filter is found such that, e.g. the trace of the matrix $\mathrm{E}((x_k - \bar{x}_k)(x_k - \bar{x}_k)^T)$ is minimized and usually also the prediction error criterion is minimized. Furthermore, given different models one should selected the best model from some validation criteria of the prediction error criterion $V = \mathrm{trace}((\mathrm{E}((y_k - \bar{y}_k)(y_k - \bar{y}_k)^T))$.

Proposition 12.1 may also be extended to the Kalman filter of system with known inputs.

12.2 Constraints in the subspace identification problem

It is possibile to include constraints in the OLS problem of computing the B and E matrices in the DSR algorithm, or constraints in the B, K, E and F matrices in the DSR_e algorithm. Note that the Kalman filter model matrices B, K, E and F is the DSR algorithms are computed as an optimal ordinary Least Squares (LS) problem. Hence additional constraints leads to a standard LS constrained problem which may be effectively solved. As an example we consider an output error system in which $K = 0$ in a state space model. It is possible to add constraints in the DSR_e method in order to force K to be zero.

12.3 Least squares with equality constraints

Consider an LS problem $Ax = b + e$ with p equality constraints in $Bx = d$, i.e. as follows

$$\min_{Bx=d} ||Ax - b||_F^2, \tag{12.7}$$

where $A \in \mathbb{R}^{N\times n}$, $b \in \mathbb{R}^N$, $B \in \mathbb{R}^{p\times n}$ and $d \in \mathbb{R}^p$. From the QR decomposition of B^T, i.e.

$$B^T = QR = Q \begin{bmatrix} R_1 \\ 0_{(n-p)\times p} \end{bmatrix}, \tag{12.8}$$

where $Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{n\times p}$ and hence $R_1 \in \mathbb{R}^{p\times p}$.

From this QR decomposition we define new variables y and z through

$$AQ = \begin{bmatrix} A_1 & A_2 \end{bmatrix}, \quad Q^T x = \begin{bmatrix} y \\ z \end{bmatrix}, \tag{12.9}$$

where $A_1 \in \mathbb{R}^{N\times p}$, $A_2 \in \mathbb{R}^{N\times(n-p)}$, $y \in \mathbb{R}^p$ and $z \in \mathbb{R}^{n-p}$. One point of doing these definitions is that then we have

$$Ax = AQQ^T x = \begin{bmatrix} A_1 & A_2 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix}. \tag{12.10}$$

At this stage, instead of one unknown variable x we have two unknown variables y and z, but notice that y may be solved from

$$Bx = R^T Q^T x = \left[\begin{array}{cc} R_1^T & 0 \end{array} \right] \left[\begin{array}{c} y \\ z \end{array} \right] = R_1^T y = d. \tag{12.11}$$

From this y may be determined and e.g. computed as

$$y = (R_1^T)^{-1} d, \tag{12.12}$$

if R_1^T is non singular. The inverse and the solution in (12.12) may be more efficiently computed by using that R_1^T is lower triangular. The next step is now to compute z from the unconstrained LS problem

$$\min_z ||A_2 z - (b - A_1 y)||_F^2. \tag{12.13}$$

This gives

$$z = (A_2^T A_2)^{-1} A_2^T (b - A_1 y). \tag{12.14}$$

Finally, now y and z are known and we compute the solution as

$$x = Q \left[\begin{array}{c} y \\ z \end{array} \right]. \tag{12.15}$$

This algorithm may be directly implemented in MATLAB as in the following m-file function

```
function x=ols_c(A,b,B,d)
% x=ols_c(A,x,Beq,Deq)
% Solve OLS problem Ax=b with p equality constraints as Beq*x=Deq.
% A - (N x n) matrix with N >=n
% b - n dimensional vector
% Beq - (p x n) dimensional matrix for constraints
% Deq - p dimensional vector for constraints

[p,n]=size(B);

[Q,R]=qr(B');
R1=R(1:p,1:p);
y=pinv(R1')*d;

AQ=A*Q;
A1=AQ(:,1:p); A2=AQ(:,p+1:n);

bz=b-A1*y;
z=pinv(A2'*A2)*A2'*bz;

x=Q*[y;z];
```

Example 12.1
Given the LS problem

$$\min_{x_2=0} ||Ax-b||_F^2, \tag{12.16}$$

where

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}, b = \begin{bmatrix} 7 \\ 1 \\ 3 \end{bmatrix}. \tag{12.17}$$

The constraint $x_2 = 0$ gives the constraint equation $Bx = d$ with

$$B = \begin{bmatrix} 0 & 1 \end{bmatrix}, \; d = \begin{bmatrix} 0 \end{bmatrix}. \tag{12.18}$$

using e.g. the MATLAB function **ols_c.m** *we obtain the solution*

$$x = \begin{bmatrix} 0.7143 \\ 0 \end{bmatrix}. \tag{12.19}$$

Notice that this example is in-dead trivial because since $x_2 = 0$ then x_1 may be found as $x_1 = A(:,1)^\dagger b = 0.7143$ and then $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.7143 \\ 0 \end{bmatrix}$.

Chapter 13

On the Partial Least Squares Algorithm

13.1 Notation, basic- and system-definitions

Define $y_k \in \mathbb{R}^m$ as a vector of output variables at observation number k. The output variables are some times referred to as *response variables*. Similarly a vector $x_k \in \mathbb{R}^r$ of input variables is defined. It is assumed that the vector of output variables y_k are linearly related to the vector of input variables x_k as follows

$$y_k = B^T x_k + e_k, \tag{13.1}$$

where e_k is a vector of white noise with covariance matrix $\mathrm{E}(e_k e_k^T)$ and k is the number of observation.

With N observations $k = 1, \cdots, N$ we define an output (or response) data matrix $Y \in \mathbb{R}^{N \times m}$ and an input data matrix $X \in \mathbb{R}^{N \times r}$ as follows

$$Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_N^T \end{bmatrix}, \quad X = \begin{bmatrix} x_1^T \\ \vdots \\ x_N^T \end{bmatrix}. \tag{13.2}$$

The data matrices Y and X are assumed to be known.

The linear relationship (13.1) can be written as the following linear matrix equation

$$Y = XB + E, \tag{13.3}$$

where B is a matrix with regression coefficients. $E \in \mathbb{R}^{N \times m}$ is in general an unknown matrix of noise vectors, defined as follows

$$E = \begin{bmatrix} e_1^T \\ \vdots \\ e_N^T \end{bmatrix}. \tag{13.4}$$

The linear relationship between the output (response) and the input data (or regressors) is an important assumption and restriction for the PLS as well as any LS algorithm to work.

We will in this work analyze systems with multiple output variables in the data matrix Y. This is often referred to as multivariable (or multivariate) systems.

If we only are interested in the matrix of regression coefficients B, then one should note that it (for steady state systems) usually is sufficient to consider one output at a time and only investigate single output systems. This means that the multivariable LS problem can be solved from m single output LS problems, i.e., each column in B is estimated from a separated LS problem.

Note also that (instead of modeling one output variable at a time) equation (13.3) can be transformed to an equivalent model with one output in different ways. Two possibile models with one output, which are equivalent to the multivariable model (13.3) are preseted as follows

$$\mathrm{cs}(Y) = (I_m \otimes X)\mathrm{cs}(B) + \mathrm{cs}(E), \tag{13.5}$$

$$\mathrm{cs}(Y^T) = (X \otimes I_m)\mathrm{cs}(B^T) + \mathrm{cs}(E^T), \tag{13.6}$$

where $\mathrm{cs}(\cdot)$ is the column string operator and $\otimes$ is the Kronecker product. $\mathrm{cs}(Y) \in \mathbb{R}^{Nm}$ is a column vector constructed from Y by stacking each column in Y on another. We also have $(I_m \otimes X) \in \mathbb{R}^{Nm \times rm}$ and $\mathrm{cs}(B) \in \mathbb{R}^{rm}$.

Note that (13.6) can be constructed directly from (13.1) by first writing (13.1) as

$$y_k = (x_k^T \otimes I_m)\mathrm{cs}(B^T) + e_k \tag{13.7}$$

and then combine all N equations ($k = 1, \cdots, N$) into a matrix equation of the form (13.3).

However, for the sake of completeness we will in general consider multivariable (multiple output) systems of the form (13.3). One should also note that the PLS algorithm can be used to compute projections. An example is the problem of computing the projection of the row space of Y^T onto the row space of X^T. In order to solve this problem we have to consider the general multiple output model (13.3).

13.2 Partial Least Squares

13.2.1 Interpretation of the Partial Least Squares algorithm

Define a vector t from the column space of X as $t = Xw$ where w is a weight vector with constraint $w^T w = 1$. The vector t is defined as the score vector for X. Similarly, a score vector u for Y is defined from the column space of Y as $u = Yc$ where c is a weight vector with constraint $c^T c = 1$.

The PLS algorithm can be interpreted as the problem of computing weight vectors w and c which maximize the correlation between the score vector t and the score vector u. Hence, we have the problem of maximizing the function $f(w,c) = t^Tu$ with respect to w and c subject to the orthonormal constraints on w and c. We can formulate the key part in the PLS algorithm as the following optimization problem.

Problem 13.1 (PLS optimization problem) The key part in the PLS algorithm can be formulated as an optimization problem for each component.

$$\begin{array}{ll} \text{Maximize} & \\ & f(w,c) = t^Tu, \\ \text{with respect to} & w \text{ and } c, \text{ where} \\ & t = Xw, \\ & u = Yc, \\ \text{subject to} & \\ & w^Tw = 1, \\ & c^Tc \ = 1. \end{array} \tag{13.8}$$

△

The Lagrangian associated with this optimization problem is given by

$$\begin{aligned} \mathcal{L}(w,c) &= t^Tu + \mu(1 - w^Tw) + \sigma(1 - c^Tc) \\ &= w^TX^TYc + \mu(1 - w^Tw) + \sigma(1 - c^Tc), \end{aligned} \tag{13.9}$$

where μ and σ are Lagrange multipliers associated with the constraints. The optimal solution is found by setting all possible derivatives of the Lagrangian to zero. We have

$$\frac{\partial \mathcal{L}}{\partial c} = Y^TXw - 2\sigma c = 0, \tag{13.10}$$

$$\frac{\partial \mathcal{L}}{\partial w} = X^TYc - 2\mu w = 0, \tag{13.11}$$

$$\frac{\partial \mathcal{L}}{\partial \mu} = 1 - w^Tw = 0, \tag{13.12}$$

$$\frac{\partial \mathcal{L}}{\partial \sigma} = 1 - c^Tc = 0. \tag{13.13}$$

From Equations (13.10) and (13.11) we have

$$c^TY^TXw = 2\sigma, \tag{13.14}$$

$$w^TX^TYc = 2\mu. \tag{13.15}$$

By transposing Equation (13.15) and comparing the result with Equation (13.14) shows that the Lagrange multipliers are equal, i.e., $\mu = \sigma$.

From this it is clear that w and c are given by the following SVD problem

$$c^TY^TXw = \bar{s}, \tag{13.16}$$

or equivalently (by transposing (13.16))

$$w^TX^TYc = \bar{s}, \tag{13.17}$$

where $\bar{s} = 2\sigma = 2\mu$ is the singular value associated with X^TY or Y^TX. The weight vectors w and c can be taken as singular vectors of X^TY or Y^TX. Furthermore, one should take w and c as the singular vectors associated with the maximal singular value of X^TY (or Y^TX). The reason for this is that the objective

$$f(w, c) = t^Tu = w^TX^TYc = \bar{s} \tag{13.18}$$

is maximized when $\bar{s}$ is the maximum singular value of X^TY. Note also that if $r > m$ then we should compute the economy size SVD of X^TY otherwise we should work with Y^TX. Further details of this SVD problem will be presented and commented on in the next section.

From Equation (13.10) we have $c = Y^TXw/(2\sigma)$. Substituting this into Equation (13.11) gives an eigenvector problem for w. Similarly, from Equation (13.11) we have $w = X^TYc/(2\mu)$. Substituting this into Equation (13.10) gives an eigenvector problem for c. Hence, we have the following eigenvector/eigenvalue problems associated with the PLS algorithm

$$\begin{aligned} X^TYY^TXw &= \lambda w, \quad \text{with } w^Tw = 1, & (13.19)\\ Y^TXX^TYc &= \lambda c, \quad \text{with } c^Tc = 1, & (13.20) \end{aligned}$$

where the eigenvalue λ is related to the Lagrangian multipliers in (13.9) as $\lambda = 4\sigma\mu$. Note also that λ is related to the maximum singular value $\bar{s}$ of (13.16) or (13.17) as $\lambda = \bar{s}^2$.

Computing the orthonormal weight vectors w and c

From Section 13.2.1 it is clear that both weight vectors w and c can be computed as the left and right singular vectors of the matrix X^TY, respectively. We will in the following assume that $r > m$. Otherwise we should work with Y^TX. In this case c is the left singular vector and w is the right singular vector associated with Y^TX, respectively.

The key step in the PLS algorithm is the following singular value decomposition

$$X^TY = \mathcal{U}\mathcal{S}\mathcal{V}^T, \tag{13.21}$$

where $\mathcal{U} \in \mathbb{R}^{r\times r}$ is a matrix of left orthonormal singular vectors, $\mathcal{V} \in \mathbb{R}^{m\times m}$ is a matrix of right orthonormal singular vectors and $\mathcal{S} \in \mathbb{R}^{r\times m}$ is a diagonal matrix of singular values. Note that a reduced singular value decomposition can be computed when $r > m$. In this case $\mathcal{U} \in \mathbb{R}^{r\times m}$, $\mathcal{S} \in \mathbb{R}^{m\times m}$ and $\mathcal{V} \in \mathbb{R}^{m\times m}$.

Define the maximum (largest) singular value of X^TY as $\bar{s}$. Hence, $\bar{s}$ is equal to the upper left (first diagonal) element of the matrix $\mathcal{S}$ of singular values. The weight vector w is given by the left singular vector associated with the largest singular value. Similarly, the weight vector c is given by the right singular vector associated with the largest singular value. We have

$$\begin{aligned} w &= \mathcal{U}(:,1), & (13.22)\\ c &= \mathcal{V}(:,1), & (13.23)\\ \bar{s} &= \mathcal{S}(1,1). & (13.24) \end{aligned}$$

The weight vectors are orthonormal, i.e., they satisfy $w^Tw = 1$ and $c^Tc = 1$. The reason for this is that the columns of the SVD matrices $\mathcal{U}$ and $\mathcal{V}$ are orthonormal.

Equation (13.21) is central in the PLS algorithm. This can be shown as follows. From the SVD in (13.21) it is clear that the eigenvector problems in (13.19) and (13.20) can be written as

$$\begin{aligned} X^TYY^TX\mathcal{U} &= \mathcal{U}\mathcal{S}\mathcal{S}^T, & (13.25)\\ Y^TXX^TY\mathcal{V} &= \mathcal{V}\mathcal{S}^T\mathcal{S}, & (13.26) \end{aligned}$$

where $\mathcal{S}\mathcal{S}^T \in \mathbb{R}^{r\times r}$ is a diagonal matrix with the eigenvalues of X^TYY^TX along the diagonal and $\mathcal{S}^T\mathcal{S} \in \mathbb{R}^{m\times m}$ is a diagonal matrix with the eigenvalues of Y^TXX^TY along the diagonal.

After w and c are computed, the score vectors can be computed as follows

$$\begin{aligned} t &= Xw, & (13.27)\\ u &= Yc. & (13.28) \end{aligned}$$

The maximum correlation between the score vectors t and u is

$$\max_{w,c} f(w,c) = t^Tu = w^TX^TYc = w^T\mathcal{U}\mathcal{S}\mathcal{V}^Tc = \bar{s}, \qquad (13.29)$$

where $\bar{s}$ is the maximum singular value of X^TY, i.e. equal to the upper left diagonal element in S. This is the solution to the PLS optimization problem (13.8).

Remark 13.1 *For one component, the maximum correlation between the t score vector and the u score vector is equal to the maximum singular value $\bar{s}$ of X^TY, i.e., $t^Tu = \bar{s}$.*

We have given an interpretation of the PLS algorithm for one component (also for the first component in the general PLS algorithm). The following algorithm illustrates the central part of the general PLS algorithm.

Algorithm 13.2.1 (Central part of the PLS algorithm)

```
$X_1 = X$
$for$    $i = 1, a$                                          Loop for all components $1 \leq a \leq r$
         $X_i^T Y = \mathcal{U}\mathcal{S}\mathcal{V}^T$                      The Singular Value Decomposition
         $w_i = \mathcal{U}(:,1)$                            The weight vector for $X$
         $c_i = \mathcal{V}(:,1)$                            The weight vector for $Y$
         $\bar{s}_i = \mathcal{S}(1,1)$                      The largest singular value of $X^T Y$
         $t_i = X_i w_i$                                     The score vector for $X$
         $X_{i+1} = (I_N - \frac{t_i t_i^T}{t_i^T t_i}) X_i$        Rank one reduction of $X$
$end$
```

This results in a (score vector) matrix $T \in \mathbb{R}^{N \times a}$ with orthogonal columns, weight matrices $C \in \mathbb{R}^{m \times a}$ and $W \in \mathbb{R}^{r \times a}$ with orthonormal columns and a diagonal matrix $S \in \mathbb{R}^{a \times a}$ of maximum singular values $\bar{s}_i$ of $X_i^T Y$. The matrices are defined as follows

$$T = \begin{bmatrix} t_1 & \cdots & t_a \end{bmatrix}, \; C = \begin{bmatrix} c_1 & \cdots & c_a \end{bmatrix}, \; W = \begin{bmatrix} w_1 & \cdots & w_a \end{bmatrix}, \tag{13.30}$$

$$S = \begin{bmatrix} \bar{s}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & & \bar{s}_a \end{bmatrix}. \tag{13.31}$$

△

The details in the algorithm will be discussed in the next sections.

Remark 13.2 *For each new component a rank one reduced matrix X_i is used in the SVD problem. Similarly it is common practice to use a rank one reduced matrix $Y_{i+1} = (I_N - \frac{t_i t_i^T}{t_i^T t_i}) Y_i$. In this case the SVD of the matrix $X_i^T Y_i$ is computed at each iteration. However, rank one reduction of Y is not necessary. Further details will be discussed in the next Section 13.2.1.*

Definition 13.1 *Define a score matrix U of score vectors for Y computed from a rank one reduced matrix Y_i as follows*

$$U = \begin{bmatrix} u_1 & \cdots & u_a \end{bmatrix} \in \mathbb{R}^{N \times a}, \tag{13.32}$$

where the score vectors are given by

$$u_i = Y_i c_i \; \in \mathbb{R}^N. \tag{13.33}$$

△

Remark 13.3 *The maximum correlation matrix between the score vector matrix T and the score vector matrix U is equal to a lower triangular matrix L, i.e.,*

$$T^T U = L \ \in \ \mathbb{R}^{a \times a}, \tag{13.34}$$

if the score matrix U is computed as in definition 13.1, i.e., from a rank one reduced (undeflated) matrix Y_i.

The diagonal element number i of L is equal to the maximum singular value $\bar{s}_i$ of $X_i^T Y$, i.e., $l_{ii} = \bar{s}_i$, where l_{ii} is diagonal element number i in matrix L. This means that

$$tr(T^T U) = tr(S), \tag{13.35}$$

where S is defined by (13.31).

Proof of remark 13.3

The statement that L given by (13.34) is a lower triangular matrix can be proved as follows. The upper triangular elements of $T^T U = L$ are given by

$$l_{i,i+1} = t_i^T u_{i+1} = t_i^T Y_{i+1} c_{i+1} = t_i^T (I - \frac{t_i t_i^T}{t_i^T t_i}) Y_i c_{i+1} = (t_i^T - t_i^T) Y_i c_{i+1} = 0 \tag{13.36}$$

QED.

Definition 13.2 *Define a score matrix $\tilde{U}$ for Y computed from an undeflated Y matrix as follows*

$$\tilde{U} = YC \ \in \mathbb{R}^{N \times a}. \tag{13.37}$$

Each score vector is defined as

$$\tilde{u}_i = Y c_i \ \in \mathbb{R}^N. \tag{13.38}$$

△

Remark 13.4 *The maximum correlation matrix between the score vector matrix T and the score matrix $\tilde{U}$ is in general equal to a full matrix M defined as*

$$T^T \tilde{U} = M \ \in \ \mathbb{R}^{a \times a}, \tag{13.39}$$

when matrix $\tilde{U}$ is computed without Y deflation, i.e., as in Definition 13.2.

The diagonal element number i of M is equal to the maximum singular value $\bar{s}_i$ of $X_i^T Y$, i.e., $m_{ii} = \bar{s}_i$, where ii is diagonal element number i in matrix M. Hence

$$tr(T^T \tilde{U}) = tr(S), \tag{13.40}$$

where S is defined by (13.31).
△

Gram-Schmidt orthogonalization

We have shown how the weight vectors w and c, and the score vectors t and u are computed (for one component) from the SVD of X^TY. See Section 13.2.1.

The next step in the PLS algorithm is a rank one reduction of X and Y. (However, one should note that rank one reduction of Y is not necessary). This step is very central in the PLS algorithm and should therefore be discussed in detail. Define the following (rank one reduced) matrices from X, Y and the t-score vector.

$$X_{i+1} = (I_N - \frac{t_i t_i^T}{t_i^T t_i})X_i, \tag{13.41}$$

$$Y_{i+1} = (I_N - \frac{t_i t_i^T}{t_i^T t_i})Y_i, \tag{13.42}$$

where $X_1 = X$ and $Y_1 = Y$. In the PLS algorithm, the weight vectors w and c and the score vectors t and u, for the next component, are computed based on the rank one reduced matrices X_{i+1} and Y_{i+1}.

The score vector t_{i+1} (for the next component) is computed based on X_{i+1} and Y_{i+1}. This means that a new SVD of $X_{i+1}^T Y_{i+1}$ is computed. The weight vectors w_{i+1}, c_{i+1} and the score vectors t_{i+1}, u_{i+1} can be computed as shown in Section 13.2.1.

t_{i+1} is orthogonal to t_i, i.e., $t_{i+1}^T t_i = 0$. The rank one reduction, Equation (13.41) and (13.42), is similar to the Gram-Schmidt orthogonalization process.

The rank one reduction process can be written in terms of so called loading vectors. The loading vector p_i for X_i is defined as

$$p_i^T = \frac{t_i^T X_i}{t_i^T t_i}. \tag{13.43}$$

The rank one reduction of X, Equation (13.41), can then be written as

$$X_{i+1} = X_i - t_i p_i^T. \tag{13.44}$$

Using that $t_i = X_i w_i$ we have the following alternative to (13.43),

$$p_i^T = \frac{w_i^T X_i^T X_i}{w_i^T X_i^T X_i w_i}. \tag{13.45}$$

The rank one reduction of X_i, Equation (13.41), can then be written as

$$X_{i+1} \quad = \quad X_i(I - w_i p_i^T). \tag{13.46}$$

Equations (13.41), (13.44) and (13.46) can be used to formulate different PLS implementations.

The loading vector $\bar{c}_i$ for Y is defined as

$$\bar{c}_i^T = \frac{t_i^T Y_i}{t_i^T t_i}. \tag{13.47}$$

The rank one reduction of Y_i, Equation (13.42), can then be written as

$$Y_{i+1} = Y_i - t_i \bar{c}_i^T. \tag{13.48}$$

By using $t_i = X_i w_i$ we can find alternative expressions for (13.47) and (13.48).

We will know show that the loading vector $\bar{c}_i$ is related to the weight vector c_i, i.e. related to the SVD of $X_i^T Y_i$. For convenience in the final PLS algorithm we define a scaled weight (loading) vector $\bar{c}_i$ for Y_i as follows

$$\bar{c}_i = \frac{\bar{s}_i}{t_i^T t_i} c_i, \tag{13.49}$$

where $\bar{s}_i$ is the largest singular value of $X_i^T Y_i$, i.e. the upper left (diagonal) element in the matrix S of singular values.

Proof of equation (13.49)

For the sake of simplicity the subscript i is neglected in the following proof. The loading vector can be written as

$$\bar{c} = \frac{Y^T t}{t^T t} = \frac{Y^T X w}{t^T t}, \tag{13.50}$$

where we have used that $t = Xw$. Using that $Y^T X = \mathcal{V}\mathcal{S}^T\mathcal{U}^T$ we have

$$\bar{c} = \frac{Y^T X w}{t^T t} = \frac{1}{t^T t}\mathcal{V}\mathcal{S}^T\mathcal{U}^T w. \tag{13.51}$$

The matrix $\mathcal{U}$ with left singular vectors for $X^T Y$ has orthonormal columns. The weight vector w is equal to the first column in $\mathcal{U}$. Hence we have

$$\mathcal{U}^T w = \begin{bmatrix} w^T \\ u_2^T \\ \vdots \\ u_r^T \end{bmatrix} w = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{13.52}$$

Using (13.52) we have that

$$\mathcal{S}^T\mathcal{U}^T w = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_m \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} s_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \bar{s} \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{13.53}$$

By using (13.53) we have that

$$\mathcal{V}\mathcal{S}^T\mathcal{U}^T w = \begin{bmatrix} c & v_2 & \cdots & v_m \end{bmatrix} \begin{bmatrix} \bar{s} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \bar{s}c. \tag{13.54}$$

Finally, we have that the loading vector $\bar{c}$ can be written as

$$\bar{c} = \frac{1}{t^T t}\mathcal{V}\mathcal{S}^T\mathcal{U}^T w = \frac{\bar{s}}{t^T t}c. \tag{13.55}$$

QED.

Remark 13.5 Note that it is not necessary to compute the rank one reduction of both matrices X and Y as shown in Equation (13.41) and (13.42), in order to compute the SVD of $X_i^T Y_i$ for each component. This can be shown as follows

$$X_{i+1}^T Y_{i+1} = X_i^T (I - \frac{t_i t_i^T}{t_i^T t_i})^T (I - \frac{t_i t_i^T}{t_i^T t_i}) Y_i. \tag{13.56}$$

The matrix $I - t_i t_i^T/(t_i^T t_i)$ is symmetric. We also have that $(I - t_i t_i^T/(t_i^T t_i))^n = I - t_i t_i^T/(t_i^T t_i)$ for positive integers n. Hence

$$X_{i+1}^T Y_{i+1} = X_i^T (I - \frac{t_i t_i^T}{t_i^T t_i}) Y_i. \tag{13.57}$$

This means that

$$X_{i+1}^T Y_{i+1} = X_{i+1}^T Y = X^T Y_{i+1}. \tag{13.58}$$

Algorithm 13.2.2 (PLS algorithm which does not reduce Y)
In the following algorithm Y is not reduced at each iteration. The price to pay is that the u-score vectors are not computed.

```
$X_1 = X$
$for$    $i = 1, a$                                             Loop for all components $1 \leq a \leq r$
         $X_i^T Y$     $=$  $\mathcal{U}\mathcal{S}\mathcal{V}^T$                  The Singular Value Decomposition
         $w_i$         $=$  $\mathcal{U}(:,1)$                     The weight vector for $X$
         $c_i$         $=$  $\mathcal{V}(:,1)$                     The weight vector for $Y$
         $\bar{s}_i$   $=$  $\mathcal{S}(1,1)$                     The largest singular value of $X^T Y$
         $t_i$         $=$  $X_i w_i$                              The score vector for $X$
         $p_i$         $=$  $\frac{X_i^T t_i}{t_i^T t_i}$          Loading vector for $X$.                   (13.59)
         $d_i$         $=$  $\frac{s_1}{t_i^T t_i}$                Scaling coefficient
         $\bar{c}_i$   $=$  $d_i c_i$                              Scaled weight (loading) vector for $Y$
         $X_{i+1}$     $=$  $(I - \frac{t_i t_i^T}{t_i^T t_i})X_i$  Rank one reduction of $X$
$end$
$W = \left[\begin{array}{ccc} w_1 & \cdots & w_a \end{array}\right]$, $\bar{C} = \left[\begin{array}{ccc} \bar{c}_1 & \cdots & \bar{c}_a \end{array}\right]$,
$P = \left[\begin{array}{ccc} p_1 & \cdots & p_a \end{array}\right]$, $T = \left[\begin{array}{ccc} t_1 & \cdots & t_a \end{array}\right]$.
```

This algorithm decomposes X into $X = TP^T + G$, where G is the residual matrix which is equal to $E = X_{a+1}$. Y is decomposed into $Y = T\bar{C}^T + F$. The residual can be computed as $F = Y - T\bar{C}^T$. The matrix of regression coefficients can be computed as $B_{\text{PLS}} = W(P^T W)^{-1}\bar{C}^T$. Note also that the rank one reduction of X can be computed as $X_{i+1} = X_i - t_i p_i^T$ or $X_{i+1} = X_i(I - w_i p_i^T)$.
△

Monte carlo simulation with PLS used on dynamical systems indicates that additional rank reduction of Y gives improved numerical precision. However, the effect of removing the rank reduction step on Y with respect to numerical precision should be further investigated.

The partial regression

We will here study the rank one reduction of X and Y which is carried out for each component in the PLS algorithm. This will in turn give us an expression for the PLS estimate of the regression coefficients.

Consider the rank one reduction (deflation) process of X and Y

$$X_{i+1} = X - tp^T, \quad p^T = \frac{t^T X}{t^T t}, \tag{13.60}$$

and

$$Y_{i+1} = Y - t\bar{c}^T, \quad \bar{c}^T = dc^T, \tag{13.61}$$

where we have defined the scalar d as

$$d = \frac{\bar{s}}{t^T t}. \tag{13.62}$$

Using that $t = Xw$ we have

$$X_{i+1} = X(I - wp^T), \tag{13.63}$$
$$Y_{i+1} = Y - Xw\bar{c}^T = Y - Xwdc^T. \tag{13.64}$$

Comparing (13.64) with the assumed linear relationship $Y = XB + E$ between X and Y, we have that the matrix of regression coefficients (for one component only) is given by

$$B_1 = w\bar{c}^T = wdc^T = \frac{\bar{s}}{t^T t} wc^T. \tag{13.65}$$

The prediction of Y (for one component only) is then given by

$$\hat{Y}_1 = XB_1 = Xw\bar{c}^T. \tag{13.66}$$

Hence, if the rank one reduction of Y is carried out at each step, it can be interpreted as the residual

$$Y_{i+1} = Y - \hat{Y}_1. \tag{13.67}$$

Example 13.1 (PLS with two components)
Assume two components $a = 2$. The residual can be written as

$$Y_3 = Y_2 - t_2\bar{c}_2^T = Y_2 - X_2 w_2 \bar{c}_2^T = Y_2 - X_2 B_2, \tag{13.68}$$

where we have defined $B_2 = w_2\bar{c}_2^T$. Substituting for $Y_2 = Y - XB_1$ and $X_2 = X(I - w_1p_1^T)$ gives

$$Y_3 = Y - XB_1 - X(I - w_1p_1^T)B_2 = Y - X(B_1 + (I - w_1p_1^T)B_2). \quad (13.69)$$

The PLS estimate of the matrix of regression coefficients B can be written as

$$B_{\text{PLS}} = B_1 + (I - w_1p_1^T)B_2 = w_1\bar{c}_1^T + (I - w_1p_1^T)w_2\bar{c}_2^T, \quad (13.70)$$

where we have used that $B_1 = w_1\bar{c}_1^T$ and $B_2 = w_2\bar{c}_2^T$. This can be written in matrix form as follows

$$B_{\text{PLS}} = \overbrace{\left[\begin{array}{cc} w_1 & w_2 \end{array}\right]}^{W} \overbrace{\left[\begin{array}{cc} 1 & -p_1^Tw_2 \\ 0 & 1 \end{array}\right]}^{(P^TW)^{-1}} \overbrace{\left[\begin{array}{cc} \bar{c}_1 & \bar{c}_2 \end{array}\right]^T}^{\bar{C}^T}. \quad (13.71)$$

It can in general be shown that $B_{\text{PLS}} = W(P^TW)^{-1}\bar{C}^T$ as indicated. The loading matrix $\bar{C}$ can be written as

$$\bar{C} = \overbrace{\left[\begin{array}{cc} c_1 & c_2 \end{array}\right]}^{C} \overbrace{\left[\begin{array}{cc} \frac{\bar{s}_1}{t_1^Tt_1} & 0 \\ 0 & \frac{\bar{s}_2}{t_2^Tt_2} \end{array}\right]}^{\bar{S}}. \quad (13.72)$$

The matrix of regression coefficients can then be written as

$$B_{\text{PLS}} = \overbrace{\left[\begin{array}{cc} w_1 & w_2 \end{array}\right]}^{W} \overbrace{\left[\begin{array}{cc} 1 & -p_1^Tw_2 \\ 0 & 1 \end{array}\right]}^{(P^TW)^{-1}} \overbrace{\left[\begin{array}{cc} \frac{\bar{s}_1}{t_1^Tt_1} & 0 \\ 0 & \frac{\bar{s}_2}{t_2^Tt_2} \end{array}\right]}^{\bar{S}} \overbrace{\left[\begin{array}{cc} c_1 & c_2 \end{array}\right]^T}^{C^T}. \quad (13.73)$$

It can in general be shown that $B_{\text{PLS}} = W(P^TW)^{-1}\bar{S}C^T$, where $\bar{S}$ is a diagonal matrix with $\bar{s}_i/(t_i^Tt_i)$ on the diagonal.
$\triangle$

The PLS algorithm step for step

The following algorithm is presented in order to illustrate the computations involved in the PLS algorithm. Refined PLS algorithms will be presented later.

Algorithm 13.2.3 (PLS computations)

```
$X_1 = X,\quad Y_1 = X$
$for$    $i = 1, a$                                          Loop for all components $1 \leq a \leq r$
         $X_i^T Y_i = USV^T$                                 The Singular Value Decomposition
         $w_i = U(:,1)$                                      The weight vector for $X$
         $c_i = V(:,1)$                                      The weight vector for $Y$
         $\bar{s}_i = S(1,1)$                                The largest singular value of $X^T Y$
         $t_i = X_i w_i$                                     The score vector for $X$
         $p_i = \frac{X_i^T t_i}{t_i^T t_i}$                 Loading vector for $X$.
         $d_i = \frac{\bar{s}_i}{t_i^T t_i}$                 Scaling coefficient
         $\bar{u}_i = Y_i c_i / d_i$                         Scaled score vector $u = Yc$ for $Y$
         $\bar{c}_i = d_i c_i$                               Scaled weight (loading) vector for $Y$
         $X_{i+1} = (I - \frac{t_i t_i^T}{t_i^T t_i}) X_i$   Rank one reduction of $X$
         $Y_{i+1} = (I - \frac{t_i t_i^T}{t_i^T t_i}) Y_i$   Rank one reduction of $Y$
$end$
```

△

Note that the rank one reduction of Y only is needed when the u-score vector is needed.

13.2.2 The Partial Least Squares decomposition

The rank one reduction step $X_{i+1} = X_i - t_i p_i^T \; \forall \; i = 1, \cdots, a$ which is carried out for each component in the iterative PLS algorithm, can be written as follows

$$X_{a+1} = X - t_1 p_1^T - \cdots t_a p_a^T. \tag{13.74}$$

The residual matrix is here defined as $G = X_{a+1}$. A similar rank one reduction for Y, i.e. $Y_{i+1} = Y_i - t_i \bar{c}_i^T \; \forall \; i = 1, \cdots, a$ can be carried out. This results in

$$Y_{a+1} = Y - t_1 \bar{c}_1^T - \cdots t_a \bar{c}_a^T. \tag{13.75}$$

The residual is defined as $F = Y_{a+1}$. Note however that the reduction of Y is not necessary in order to compute the decomposition.

Equations (13.74) and (13.75) can be written in matrix form. Hence, the PLS algorithm decomposes X and Y as follows

$$\begin{aligned} X &= TP^T + G, && (13.76) \\ Y &= T\bar{C}^T + F, && (13.77) \end{aligned}$$

where $G \in \mathbb{R}^{N \times r}$ and $F \in \mathbb{R}^{N \times m}$ are residual matrices. $T \in \mathbb{R}^{N \times a}$ is a matrix of score vectors associated with the X matrix. Note that T is an orthogonal matrix, i.e. $T^T T$ is a diagonal matrix. $P \in \mathbb{R}^{r \times a}$ is a matrix of loading vectors associated with X. $\bar{C} \in \mathbb{R}^{m \times a}$ is a matrix of scaled weight vectors (or loading vectors) associated with Y.

The loading matrix (scaled weight matrix) $\bar{C}$ for Y can be decomposed into

$$\bar{C} = CS\Lambda_t^{-1} = Y^T T \Lambda_t^{-1}, \tag{13.78}$$

where C is a matrix with orthonormal columns c_i, i.e., $c_i^T c_i = 1$. See the Appendix, Equation (13.185) for further details. c_i can be taken from the right singular vectors of $X_i^T Y$. S is a diagonal matrix with the maximum singular values $\bar{s}_i$ of $X_i^T Y$ on the diagonal. Λ_t is a diagonal matrix with $t_i^T t_i$ on the diagonal. Note that $t_i = X_i w_i$ and that

$$\Lambda_t = T^T T. \tag{13.79}$$

The loading matrix P for X can then be decomposed into

$$P = P_t \Lambda_t^{-1}, \tag{13.80}$$

where P_t is the unscaled loading matrix with columns formed from the unscaled loading vectors, i.e.,

$$p_{t,i} = X_i^T t_i. \tag{13.81}$$

We will in the next section use the above relations to formulate a PLS decomposition with an orthonormal score vector matrix for X.

By combining Equations (13.80) and (13.81) we find that the P loading matrix for X, in general can be expressed as $P = X^T T (T^T T)^{-1}$. This expression for P can be proved by putting the gradient matrix of $\parallel G \parallel_F^2$ with respect to $\text{tr}(P)$ equal to zero. These results are presented in the following lemma.

Lemma 13.2.1 (PLS decomposition in terms of the T-score matrix) *The P loading matrix for X and the $\bar{C}$ loading matrix for Y can in general be expressed as*

$$\begin{aligned} P &= X^T T (T^T T)^{-1}, & (13.82)\\ \bar{C} &= Y^T T (T^T T)^{-1}, & (13.83)\\ & & (13.84) \end{aligned}$$

where we have assumed that $T^T T$ is non-singular.

The PLS decomposition, Equations (13.76) and (13.77), can in general be expressed as

$$\begin{aligned} X &= T(T^T T)^{-1} T^T X + G, & (13.85)\\ Y &= T(T^T T)^{-1} T^T Y + F. & (13.86) \end{aligned}$$

The $T \in \mathbb{R}^{N \times a}$ score vector matrix for X can in general be expressed as

$$T = XW(P^T W)^{-1} = XW(T^T X W)^{-1}(T^T T). \tag{13.87}$$

△

Orthonormal score vector matrix

In some applications, e.g. in subspace identification algorithms for dynamic systems, orthonormal decompositions are prefered. The PLS decomposition, Equations (13.76) and (13.77), can be rewritten as

$$\begin{aligned} X &= T_o P_o^T + G, & (13.88)\\ Y &= T_o C_o^T + F, & (13.89) \end{aligned}$$

where

$$\begin{aligned} T_o &= T\Lambda_t^{-\frac{1}{2}}, & (13.90)\\ P_o &= P\Lambda_t^{\frac{1}{2}} = P_t\Lambda_t^{-\frac{1}{2}}, & (13.91)\\ C_o &= \bar{C}\Lambda_t^{\frac{1}{2}} = CS\Lambda_t^{-\frac{1}{2}}. & (13.92) \end{aligned}$$

The scaled score vector T_o is orthonormal, i.e. $T_o^T T_o = I_a$. Note that $\Lambda_t = T^T T$ is a diagonal matrix.

Note also the following alternative description of the orthonormal PLS decomposition

$$\begin{aligned} X &= \overbrace{T\Lambda_t^{-\frac{1}{2}}}^{T_o}\overbrace{\Lambda_t^{-\frac{1}{2}}P_t^T}^{P_o} + G, & (13.93)\\ Y &= T\Lambda_t^{-\frac{1}{2}}\underbrace{\Lambda_t^{-\frac{1}{2}}SC^T}_{C_o} + F. & (13.94) \end{aligned}$$

13.2.3 The Partial Least Squares regression and prediction

We will in this section discuss some alternative methods for computing the PLS estimate of the matrix of regression coefficients B.

From Equation (13.77) we have that the prediction of Y is given by

$$\hat{Y} = T\bar{C}^T. \quad (13.95)$$

Similarly, from Equation (13.76) we have that the prediction of X (or the information in X used for modeling Y) is given by

$$\hat{X} = TP^T. \quad (13.96)$$

The matrix of PLS regression coefficients is given by

$$\hat{B} = W(P^T W)^{-1}\bar{C}^T. \quad (13.97)$$

Note that only the weight matrix W, loading matrix $\bar{C}$ and the loading matrix P are needed in order to compute B. Hence, the score vector matrices T and U are not explicitly needed. T is implicitly computed in order to define P and $\bar{C}$. U need not to be computed in order to compute $\hat{B}$. Note also that the matrix

P^TW which have to be inverted is of size $a \times a$. P^TW is an upper triangular matrix with ones on the diagonal. This should be utilized when computing $\hat{B}$ for numerical efficiency.

The matrix of PLS regression coefficients can equivalently be written as

$$\hat{B} = W(P_t^TW)^{-1}SC^T = W(T^TXW)^{-1}SC^T, \tag{13.98}$$

where $P_t^TW = T^TXW$ is an upper triangular matrix with $t_i^Tt_i$ on the diagonal. Se also Lemma 13.2.1.

Proof of Equation (13.97)

The prediction of Y can be written as

$$\hat{Y} = \hat{X}\hat{B}. \tag{13.99}$$

Substituting for $\hat{X} = TP^T$ and $\hat{B}$ given by (13.97) gives

$$\hat{Y} = TP^TW(P^TW)^{-1}\bar{C}^T = T\bar{C}^T, \tag{13.100}$$

which is equivalent to Equation (13.95).
QED.

The matrix of regression coefficients can be computed recursively during the PLS iterations. This is numerically preferable because we in this case do not have to explicitly inverting the upper triangular matrix P^TW. We have

$$\hat{B} = \sum_{i=1}^{a}(I_r - w_{i-1}p_{i-1}^T)w_i\bar{c}_i^T, \quad w_0 = 0, \; p_0 = 0. \tag{13.101}$$

This formula can be proved from the PLS updating expressions $Y_{i+1} = Y_i - X_iw_i\bar{c}_i^T$ and $X_{i+1} = X_i(I - w_ip_i^T)$. We also have the following alternative expression

$$\hat{B} = \sum_{i=1}^{a}(I_r - w_{i-1}p_{i-1}^T)w_i\frac{\bar{s}_i}{t_i^Tt_i}c_i^T, \quad w_0 = 0, \; p_0 = 0. \tag{13.102}$$

Remark 13.6 (Prediction) Note that the prediction of Y usually is computed as

$$\hat{Y} = X\hat{B}, \tag{13.103}$$

when X is not used for computing $\hat{B}$. Hence, Equation (13.103) is used for validation and prediction.

13.2.4 Computing projections from the PLS decomposition

Consider the basic linear relationship

$$Y^T = B^T X^T + E^T. \tag{13.104}$$

The projection of (the row space) of a matrix A onto (the row space) of a matrix B is defined as follows

$$A/B = AB^T(BB^T)^\dagger B, \tag{13.105}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo inverse. We also define the projection of the row space of A onto the orthogonal complement of the row space of B as

$$AB^\perp = A - AB^T(BB^T)^\dagger B. \tag{13.106}$$

This implies that A is decomposed into two terms which are orthogonal to each other, i.e.

$$A = A/B + AB^\perp. \tag{13.107}$$

Then we have the following projection of Y^T onto X^T.

$$Y^T/X^T = B^T X^T + E^T/X^T. \tag{13.108}$$

The projection of E^T onto X^T is assumed to be zero. This is the case when $N \to \infty$ and the noise process e_k is white and serially uncorrelated with x_i. The projection can then be computed as

$$Y^T/X^T = \hat{B}^T \hat{X}^T. \tag{13.109}$$

Substituting for the decomposition $\hat{X} = TP^T$ and the estimate of the regression matrix $\hat{B} = W(P^TW)^{-1}\bar{C}^T$ into (13.109) gives

$$Y^T/X^T = \bar{C}(P^TW)^{-T}W^TPT^T = \bar{C}T^T. \tag{13.110}$$

Using the orthonormal PLS decomposition $Y = T_oC_o^T$ and $X = T_oP_o^T$ we have the following alternative description

$$Y^T/X^T = C_oP_o^T(P_oP_o^T)^{-1}\overbrace{P_oT_o^T}^{X^T}. \tag{13.111}$$

where we have used (13.110). Comparing (13.111) with (13.108) shows that

$$B^T = C_oP_o^T(P_oP_o^T)^{-1} \quad \text{and} \quad B = (P_oP_o^T)^{-1}P_oC_o^T \quad , \tag{13.112}$$

when $P_oP_o^T \in \mathbb{R}^{r\times r}$ is non-singular.

Note also that the projection of Y^T onto the orthogonal complement of X^T is equal to the residual G^T from the PLS decomposition.

13.2.5 A semi standard PLS implementation

The version of the PLS algorithm which is presented in this section is denoted semi standard because it is based on the SVD.

Algorithm 13.2.4 (Standard PLS implementation with SVD)

```
$X_1 = X, \quad Y_1 = Y$
$for$   $i = 1, a$                                       Loop for all components $1 \le a \le r$
        $X_i^T Y_i = USV^T$                              The Singular Value Decomposition
        $w_i = U(:,1)$                                   The weight vector for $X$
        $t_i = X_i w_i$                                  The score vector for $X$
        $\bar{c}_i = \frac{Y_i^T t_i}{t_i^T t_i}$        Scaled weight vector for $Y$
        $\bar{u}_i = \frac{Y_i \bar{c}_i}{\bar{c}_i^T \bar{c}_i}$   Score vector for $Y$
        $p_i = \frac{X_i^T t_i}{t_i^T t_i}$              Loading vector for $X$
        $X_{i+1} = X_i - t_i p_i^T$                      Updating of $X$, (rank one reduction of $X$)
        $Y_{i+1} = Y_i - t_i \bar{c}_i^T$                Updating of $Y$, (rank one reduction of $Y$)
$end$
```

The loading matrix P, the weight matrix W and the loading matrix $\bar{C}$ are then given by

$$P = \begin{bmatrix} p_1 & \cdots & p_a \end{bmatrix}, \quad W = \begin{bmatrix} w_1 & \cdots & w_a \end{bmatrix}, \quad \bar{C} = \begin{bmatrix} \bar{c}_1 & \cdots & \bar{c}_a \end{bmatrix} \tag{13.113}$$

The matrix of regression coefficients can then be computed as $B_{\text{PLS}} = W(P^T W)^{-1} \bar{C}^T$. Note that $P^T W$ is upper triangular with ones on the diagonal.

The algorithm also computes the score vector matrices

$$T = \begin{bmatrix} t_1 & \cdots & t_a \end{bmatrix}, \quad \bar{U} = \begin{bmatrix} \bar{u}_1 & \cdots & \bar{u}_a \end{bmatrix}. \tag{13.114}$$

△

13.2.6 The number of components a

The number of variables in X is r. The number of PLS components a (or factors) is bounded from above by r. Hence,

$$1 \le a \le r.$$

The problem of choosing the number of components a is in general a trade of between, on the one hand, “rank decision” based on the score matrices U and T or the largest singular values $\bar{s}_i$ of $X_i^T Y_i$, $i = 1, \cdots, r$, and, on the other hand, model validation on independent data.

The PLS algorithm is constructed in order to maximize the linear relationship between the u_i score for Y_i and the t_i score for X_i. Recall that the u_i score vector lies in the column space of Y_i and that the t_i score vector lies in the column space of X_i. See Problem (13.1). In order to chose the effective number of components a one can plot the score vector u_i against the score vector t_i for all $i = 1, \cdots, a$ and choose a from the plot where there is no linear relationship between u_a and t_a.

Another strategy is simply to choose a as the number of "non zero" significant singular values $\bar{s}_i \; \forall \; i = 1, \cdots, r$.

The problem of choosing a can of course be done recursively during the PLS iteration. One procedure is to plot u_i against t_i at each PLS iteration and by inspection decide if a new component should be incorporated. Another procedure is to at each PLS iteration check if $\bar{s}_i$ is significant different from zero.

We end this section by pointing out that it often make sense to be parsimonious when choosing the number of components.

13.3 Further interpretations of the PLS algorithm

13.3.1 General results

The results in this sections are general in the sense that it holds for all components a. However, we will restrict our results to $1 \leq a \leq r$ where r is the number of X variables.

Remark 13.7 *We have the following general expression for the Squared Prediction Error (SPE) when the PLS algorithm is used*

$$V_N = \parallel Y - X\hat{B} \parallel_F^2 = \parallel Y - T\bar{C}^T \parallel_F^2 \tag{13.115}$$

which can be expressed as

$$V_N = tr(Y^T Y) - 2tr(S\Lambda_t^{-1} T^T Y C) + tr(CS\Lambda_t^{-1} SC^T) \tag{13.116}$$

by using that $\bar{C} = CS(T^T T)^{-1}$. Furthermore, (13.116) can be evaluated as

$$\begin{aligned} V_N(B) = \parallel Y - X\hat{B} \parallel_F^2 &= tr(Y^T Y) - tr(S^2 \Lambda_t^{-1}) \\ &= tr(Y^T Y) - \sum_{i=1}^{a} \frac{\bar{s}_i^2}{t_i^T t_i}, \end{aligned} \tag{13.117}$$

where we have assumed that $\Lambda_t = T^T T$ is non-singular. a is the number of components which usually is bounded as $1 \leq a \leq r$. For each new component which is incorporated, the variance in the output variables is reduced by the positive term $\frac{\bar{s}_i^2}{t_i^T t_i}$. It is assumed that $t_i^T t_i \neq 0$.
△

Remark 13.8 *Putting the gradient matrix of Equation (13.116) with respect to C equal to zero, gives the following general PLS condition*

$$SC^T = T^T Y. \tag{13.118}$$

This expression is to our knowledge new.

We have also derived a quite different and general expression for the SPE as follows.

Remark 13.9 *Substituting Equation (13.118) into (13.116) shows that the SPE can be written as*

$$V_N = tr(Y^TY) - tr(Y^TT\Lambda_t^{-1}T^TY) = tr(Y^TY) - tr(Y^TT_0T_0^TY). \quad (13.119)$$

The first term on the right hand side is the sum of variances for each output (response) variable (times $N-1$). The second term is the sum of variances for the predicted outputs (times $N-1$), i.e.,

$$\hat{Y} = X\hat{B} = T\backslash Y = T\Lambda_t^{-1}T^TY = T_oT_o^TY, \quad (13.120)$$

where T_o is the orthogonal score matrix. Note that for finite N we have $E(\hat{y}_k\hat{y}_k^T) \approx \frac{1}{N-1}\hat{Y}^T\hat{Y}$ and $(N-1)E(\hat{y}_k\hat{y}_k^T) \approx tr(\hat{Y}^T\hat{Y})$.

The second term on the right hand side of (13.119) can be computed from the SVD of the matrix

$$\Lambda_t^{-\frac{1}{2}}T^TY = T_o^TY = \tilde{\mathcal{U}}\tilde{\mathcal{S}}\tilde{\mathcal{V}}^T. \quad (13.121)$$

Then,

$$V_N = tr(Y^TY) - \sum_{i=1}^{\min(a,m)} \tilde{s}_i^2. \quad (13.122)$$

Note the difference in the second term on the right hand side of Equations (13.117) and (13.122). The number of singular values $\bar{s}$ is a but the number of singular values $\tilde{s}_i$ is $\min(a,m)$.
△

13.3.2 Special case results

The results in this section holds in general only for one component, i.e., $a = 1$.

Remark 13.10 *We have the following expression for the Squared Prediction Error (SPE) when $a = 1$.*

$$V_N = \parallel Y - XB_1 \parallel_F^2 = tr(Y^TY) - \frac{\bar{s}^2}{t^Tt}, \quad (13.123)$$

where $T = t = Xw$ and $\bar{s}$ is the maximum singular value of X^TY. We have assumed that t^Tt is non-singular.
△

Remark 13.11 *We have the following expression for the squared Frobenius norm of $u-t$*

$$\| u-t \|_F^2 = tr(Y^TY) + tr(X^TX) - 2tr(W^TX^TYC), \tag{13.124}$$

where $t=XW$ and $u=YC$. Hence, in the PLS algorithm the weight vectors are computed as

$$\min_{W,C} \| t-u \|_F^2 = \max_{W,C}(t^Tu). \tag{13.125}$$

This gives an alternative interpretation of the basic PLS optimization Problem 13.1.
△

Remark 13.12 *In methods to solve least squares problems it is often useful to have some knowledge of the norm of the solution. An example is the Tikhonov regularized solution or the least squares solution with a quadratic inequality side constraint on the solution $\hat{B}$. See Hansen (1992).*

For PLS with one component ($a=1$) we have

$$\| B_1 \|_F = \frac{\bar{s}}{t^Tt} = \frac{\| X^TY \|_F}{w^TX^TXw}. \tag{13.126}$$

Note that, in this case, the above also holds for the 2-norm of the solution, i.e., by replacing $\| \cdot \|_F$ in (13.126) with $\| \cdot \|_2$.

13.4 A covariance based PLS implementation

From the basic linear Equation (13.3) we have

$$X^TY = X^TXB + X^TE \tag{13.127}$$

which is the normal equation for the LS problem. We will in the following present a version of the PLS algorithm which works with X^TY and X^TX rather than X and Y.

The weights w and c are computed from the SVD of X^TY. First, initialize $(X^TX)_i = X^TX$ and $(X^TY)_i = X^TY$, then compute the weights w_i and c_i from the SVD of $(X^TY)_i$ as follows,

$$(X^TY)_i = USV^T, \quad w_i = U(:,1), \quad c_i = U(:,1), \quad \bar{s}_i = S(1,1). \tag{13.128}$$

Compute the loading vector p_i as

$$p_i = \frac{(X^TX)_i w_i}{z_i}, \quad \text{where} \quad z_i = w_i^T(X^TX)_i w_i. \tag{13.129}$$

The loading vector $\bar{c}_i$ can be computed as

$$\bar{c}_i = \frac{(X^TY)_i^T w_i}{z_i} = \frac{\bar{s}_i}{z_i} c_i. \tag{13.130}$$

Finally, the correlation matrix X^TY and the covariance matrix X^TX are updated directly as

$$\begin{aligned}(X^TX)_{i+1} &= (I - w_ip_i^T)^T(X^TX)_i(I - w_ip_i^T), &(13.131)\\ (X^TY)_{i+1} &= (I - w_ip_i^T)^T(X^TY)_i. &(13.132)\end{aligned}$$

Equations (13.128) to (13.132) are computed for all $i = 1, \cdots, a$ components, where $1 \leq a \leq r$. This procedure results in the matrices W, $\bar{C}$ and P. The matrix of regression coefficients can then be computed as $B_{\rm PLS} = W(P^TW)^{-1}\bar{C}$. Remark that the product P^TW is upper triangular with ones on the diagonal. This structure should be used for efficient computation of $(P^TW)^{-1}$.

The covariance based PLS implementation is very simple. It is very fast compared to the traditional PLS when the number of observations N is very large compared to the number of input variables r and the number of output variables m.

The initialization of the covariance matrix X^TX and the correlation matrix X^TY can be a problem due to rounding off errors when the number of observations N is large. The algorithm which is presented in the next section is numerically robust.

13.5 A square root PLS algorithm

Consider the QR decomposition of the concatenated matrix

$$\left[\begin{array}{cc} X & Y \end{array}\right] = \left[\begin{array}{cc} Q_1 & Q_2 \end{array}\right] \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22} \end{array}\right]. \qquad (13.133)$$

From this we have that $X = Q_1R_{11}$ and $Y = Q_1R_{12} + Q_2R_{22}$. Substituting into the linear equation $Y = XB + E$ gives the normal equation

$$R_{12} = R_{11}B, \qquad (13.134)$$

where $R_{11} \in \mathbb{R}^{r\times r}$ is upper triangular and $R_{12} \in \mathbb{R}^{r\times m}$.

A standard PLS algorithm can then be used to decompose R_{11} and R_{12} as follows

$$\begin{aligned}R_{11} &= TP^T + G, &(13.135)\\ R_{12} &= T\bar{C}^T + F &(13.136)\end{aligned}$$

and to compute the weighting matrix W. The PLS estimate of B is then $B_{\rm PLS} = W(P^TW)^{-1}\bar{C}^T$.

The above algorithm is numerically robust in the sense that it avoid possibly rounding off errors when computing X^TY. The reason for this is that the square root PLS algorithm instead works on $X^TY = R_{11}^TR_{12}$. However, the square root PLS algorithm is not so fast as the covariance based PLS algorithm.

13.6 Undeflated PLS

The PLS algorithm is iterative. At each iteration a rank one reduced (deflated) matrix X_i is computed. Usually a rank one reduction (deflation) of Y is also carried out. Deflation of Y is not necessary in order to compute the regression matrix B. The reason is that $X_i^T Y_i = X_i^T Y$. In the following an LS method without deflation is presented. This method is denoted UDPLS. The results from UDPLS are in general different from the results from PLS.

Problem 13.2 (UDPLS optimization problem)

$$\begin{array}{ll} \text{Maximize} & \\ & f(W, C) = \mathrm{tr}(T^T U), \\ \text{with respect to} & W \text{ and } C, \text{ where} \\ & T = XW, \\ & U = YC, \\ \text{subject to} & \\ & W^T W = I, \\ & C^T C \;\; = I. \end{array}$$

The Lagrangian associated with this optimization problem is given by

$$\mathcal{L}(w, c) = \mathrm{tr}(W^T X^T Y C) + \mathrm{tr}[\Gamma^T (I - W^T W)] + \mathrm{tr}[\Sigma^T (I - C^T C)], \quad (13.137)$$

where Γ and Σ are matrices with Lagrange multipliers associated with the constraints. The optimal solution is found by setting all possible gradient matrices of the Lagrangian to zero. We have

$$\frac{\partial \mathcal{L}}{\partial W} = (C^T Y^T X - 2\Gamma^T W^T)^T = 0, \;\Rightarrow X^T Y C = 2W\Gamma. \quad (13.138)$$

$$\frac{\partial \mathcal{L}}{\partial C} = (W^T X^T Y - 2\Sigma^T C^T)^T = 0, \;\Rightarrow Y^T X W = 2C\Sigma. \quad (13.139)$$

These derivatives are restricted to symmetric matrices Γ and Σ.

For non-singular Σ we have $C = 0.5 Y^T X W \Sigma^{-1}$, which gives

$$X^T Y Y^T X W = 4W\Gamma\Sigma, \quad (13.140)$$

$$Y^T X X^T Y C = 4C\Sigma\Gamma. \quad (13.141)$$

The key step is the following SVD

$$X^T Y = \mathcal{U}\mathcal{S}\mathcal{V}^T. \quad (13.142)$$

This gives

$$\begin{aligned} X^T Y Y^T X \mathcal{U} &= \mathcal{U}\mathcal{S}\mathcal{S}^T, & (13.143) \\ Y^T X X^T Y \mathcal{V} &= \mathcal{V}\mathcal{S}^T\mathcal{S}. & (13.144) \end{aligned}$$

Chose W and C as

$$\begin{aligned} W &= \mathcal{U}, & (13.145) \\ C &= \mathcal{V}. & (13.146) \end{aligned}$$

Note that

$$\begin{aligned} T &= XW, & (13.147) \\ T^T U &= W^T X^T Y C = S. & (13.148) \end{aligned}$$

The matrix of regression coefficients are given by

$$\hat{B}_{\text{UDPLS}} = W(T^T T)^{-1} S C^T = \mathcal{U}(\mathcal{U}^T X^T X \mathcal{U})^{-1} S \mathcal{V}. \tag{13.149}$$

where one should use only the a first components of T and $\mathcal{U}$. The estimate can also be written as

$$\hat{B}_{\text{UDPLS}} = \mathcal{U}(\mathcal{U}^T X^T X \mathcal{U})^{-1} \mathcal{U}^T X^T Y. \tag{13.150}$$

Note that the above method is equivalent to the ordinary PLS algorithm when $a = 1$.

13.7 Total Least Squares and Truncated Total Least Squares

Total Least Squares (TLS) and Truncated Total Least Squares (TTLS) are methods for solving over-determined linear systems of equations. To be specific, TLS is a technique to solve models of the type $Y = XB + E$, where not only the left hand side Y is subject to errors but also X is subject to errors. This model is known by statisticians as the *errors in variables model*. This model is also known as a bi-linear model because Y is linear in both X and B but both B and X is unknown, since X is subject to errors. Software for TLS and TTLS are presented in Hansen (1992).

13.7.1 Total Least Squares

Consider the Total Least Squares (TLS) problem

$$\min_{Z,B} f(Z, B), \tag{13.151}$$

where

$$f(Z, B) = \|X - Z\|_F^2 + \|Y - ZB\|_F^2, \tag{13.152}$$

where $\|\cdot\|_F$ is the Frobenius norm. The solution to this optimization problem can be found from the SVD of the compound (or concatenated) matrix $[X \;\; Y]$. We have

$$\begin{bmatrix} X & Y \end{bmatrix} = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix} \begin{bmatrix} V_{11}^T & V_{21}^T \\ V_{12}^T & V_{22} \end{bmatrix}, \tag{13.153}$$

where $U_1 \in \mathbb{R}^{N\times r}$, $U_2 \in \mathbb{R}^{N\times m}$, $S_1 \in \mathbb{R}^{r\times r}$, $S_2 \in \mathbb{R}^{m\times m}$, $V_{11} \in \mathbb{R}^{r\times r}$, $V_{12} \in \mathbb{R}^{r\times m}$, $V_{21} \in \mathbb{R}^{m\times r}$ and $V_{22} \in \mathbb{R}^{m\times m}$. The solution is given by

$$\begin{aligned} B_{\text{TLS}} &= -V_{12}V_{22}^{-1}, && (13.154) \\ Z &= U_1 S_1 V_{11}^T. && (13.155) \end{aligned}$$

In order to prove this results one should note that $\|A\|_F^2 = \text{tr}(A^T A)$. The results in this section are from Moor and David (199?).

Proof of TLS solution

An argumentation/proof of the above TLS solution is given in the following. First, note that the model (13.3) can be equivalently written as

$$\begin{bmatrix} X & Y \end{bmatrix} \begin{bmatrix} -B \\ I_m \end{bmatrix} = E. \qquad (13.156)$$

From the SVD (13.153) we have

$$\begin{bmatrix} X & Y \end{bmatrix} = U_1 S_1 \begin{bmatrix} V_{11} \\ V_{21} \end{bmatrix}^T + U_2 S_2 \begin{bmatrix} V_{12} \\ V_{22} \end{bmatrix}^T. \qquad (13.157)$$

From this we have

$$\begin{bmatrix} X & Y \end{bmatrix} \begin{bmatrix} V_{12} \\ V_{22} \end{bmatrix} = U_2 S_2 \qquad (13.158)$$

and

$$\begin{bmatrix} X & Y \end{bmatrix} \begin{bmatrix} V_{12}V_{22}^{-1} \\ I_m \end{bmatrix} = U_2 S_2 V_{22}^{-1}. \qquad (13.159)$$

Comparing (13.159) with (13.156) we get the TLS estimate of the regression matrix (13.154). From this we also have an estimate of the residual matrix E.

13.7.2 Truncated Total Least Squares

The TLS solution presented in the above Section can be extended to problems where X is nearly rank deficient. A technique to solve this problem is Truncated Total Least Squares (TTLS). See e.g., Fierro *et al*, Hansen (1992). Basically, the idea is to only retain the $1 \le a \le r$ largest singular values of the compound matrix $\begin{bmatrix} X & Y \end{bmatrix}$ in the matrix S_1. The TTLS solution is similar to the TLS solution, exept that we now have

$$V_{22} \in \mathbb{R}^{(r-a+m)\times m}, V_{12} \in \mathbb{R}^{r\times(r-a+m)} \qquad (13.160)$$

and

$$B_{\text{TTLS}} = -V_{12}V_{22}^{\dagger}. \qquad (13.161)$$

Remark 13.13 *Note also that the above TLS and TTLS methods can be modified to work directly on the normal Equation (13.127). In this case we take the SVD of the concatenated matrix $[X^TX \;\; X^TY]$. The solution are then found from the SVD and the following formulation of the normal equation*

$$\begin{bmatrix} X^TX & X^TY \end{bmatrix} \begin{bmatrix} -B \\ I_m \end{bmatrix} = X^TE. \tag{13.162}$$

13.8 Performance and PLS

Motivated by the TLS performance objective (13.152) we suggest the following performance objective to be minimized

$$f(P,B) = \|X - TP^T\|_F^2 + \|Y - TP^TB\|_F^2. \tag{13.163}$$

Let us first study the second part which is a measure of the residual F. For one component we have

$$\|F\|_F^2 = \|Y - tp^TB\|_F^2. \tag{13.164}$$

Substituting for $B_{\text{PLS}} = w(p^Tw)^{-1}\bar{c}^T$ and $\bar{c} = Y^Tt/(t^Tt)$ gives (or simply replace the right hand side of (13.164) with the Gram-Schmidt orthogonalization process, Equation (13.42), which is central in the PLS algorithm)

$$\|F\|_F^2 = \|(I - \frac{tt^T}{t^Tt})Y\|_F^2. \tag{13.165}$$

Using the relationship between the Frobenius norm and trace we have

$$\|F\|_F^2 = \text{tr}[Y^T(I - \frac{tt^T}{t^Tt})Y] \tag{13.166}$$

13.9 Optimal solution

From the discussion in this work we have that the PLS solution is of the form

$$B = W(W^TX^TXW)^{-1}W^TX^TY \tag{13.167}$$

where $W \in \mathbb{R}^{r\times a}$ and a is the number of components bounded by $1 \leq a \leq r$. W is a so called weighting matrix. Different regression methods gives different weighting matrices.

The squared Frobenius norm of the residual $Y - XB$ is given by

$$V(W) = \|Y - XB\|_F^2 = \text{tr}(Y^TY) - \text{tr}(Y^TXW(W^TX^TXW)^{-1}W^TX^TY)\tag{13.168}$$

Assume first for simplicity that W is equal to a vector w. The squared Frobenius norm of the residual is in this case given by

$$V(w) = \|Y - XB\|_F^2 = Y^TY - \frac{Y^TXww^TX^TY}{w^TX^TXw} = Y^TY - \frac{w^TX^TYY^TXw}{w^TX^TXw}\tag{13.169}$$

where we also have assumed that Y is a vector.

The minimum weight vector w can be found by putting the gradient of $V(w)$ with respect to w equal to zero. The gradient is given by

$$\frac{dV(w)}{dw} = -\frac{2X^TYY^TXw(w^TX^TXw) - w^TX^TYY^TXw(2X^TXw)}{(w^TX^TXw)^2} \qquad (13.170)$$

Putting the gradient equal to zero gives

$$X^TYY^TXw = \frac{w^TX^TYY^TXw}{w^TX^TXw}X^TXw \qquad (13.171)$$

This is a generalized eigenvalue problem, i.e. $\lambda_1 = \frac{w^TX^TYY^TXw}{w^TX^TXw}$ is a generalized eigenvalue of the square matrices X^TYY^TX and X^TX and w is the corresponding generalized eigenvector.

From this we have that a solution in general (i.e. when $W \in \mathbb{R}^{r\times a}$ and the number of components satisfy $1 \leq a \leq r$) can be computed by the following generalized eigenvalue problem

$$X^TYY^TXW = X^TXW\Lambda \qquad (13.172)$$

where $\Lambda \in \mathbb{R}^{r\times r}$ is a diagonal matrix with the generalized eigenvalues on the diagonal. $W \in \mathbb{R}^{r\times r}$ is the corresponding generalized eigenvector matrix. W and Λ can e.g. be computed by the MATLAB function eig$(\cdot,\cdot)$, i.e.

$$\left[\begin{array}{cc} W, & \Lambda \end{array}\right] = \mathrm{eig}(X^TYY^TX, X^TX) \qquad (13.173)$$

The weight matrix corresponding to the first a generalized eigenvalues is then given by

$$W := W(:, 1:a); \qquad (13.174)$$

Note that it is possibile to compute only the a first generalized eigenvectors.

The norm of the residual is given by

$$V(W) = \|Y - XB\|_F^2 = \mathrm{tr}(Y^TY) - \mathrm{tr}(\Lambda(1:a, 1:a)). = \mathrm{tr}(Y^TY) - \sum_{i=1}^{a}\lambda_i \qquad (13.175)$$

13.10 Numerical examples

Example 13.2
Consider the following example from Hansen (1992)

$$\overbrace{\begin{bmatrix} 0.27 \\ 0.25 \\ 3.33 \end{bmatrix}}^{Y} = \overbrace{\begin{bmatrix} 0.16 & 0.10 \\ 0.17 & 0.11 \\ 2.02 & 1.29 \end{bmatrix}}^{X} \overbrace{\begin{bmatrix} 1.00 \\ 1.00 \end{bmatrix}}^{B} + \overbrace{\begin{bmatrix} 0.01 \\ -0.03 \\ 0.02 \end{bmatrix}}^{E}. \qquad (13.176)$$

The problem adressed is to find the best estimate of B from given data matrices X and Y and knowledge of the model structure (13.3).

$$B_{\mathrm{LS}} = \begin{bmatrix} 7.01 \\ -8.40 \end{bmatrix}, \; \| B_{\mathrm{LS}}\|_2 = 10.94, \; \| Y - XB_{\mathrm{LS}}\|_2 = 0.02. \quad (13.177)$$

$$B_{\mathrm{PLS}} = \begin{bmatrix} 1.1703 \\ 0.7473 \end{bmatrix}, \; \| B_{\mathrm{PLS}}\|_2 = 1.3885, \; \| Y - XB_{\mathrm{PLS}}\|_2 = 0.0312. \quad (13.178)$$

$$B_{\mathrm{TTLS}} = \begin{bmatrix} 1.1703 \\ 0.7473 \end{bmatrix}, \; \| B_{\mathrm{TTLS}}\|_2 = 1.3885, \; \| Y - XB_{\mathrm{TTLS}}\|_2 = 0.0327. \quad (13.179)$$

A major difficulty with the above ordinary least squares solution B_{LS} in (13.177) is that its norm is significantly grater than the norm of the exact solution, which is $\|B\|_2 = \sqrt{2}$. One component ($a = 1$) was specified for the PLS and TTLS algorithms. The PLS and TTLS solutions are almost similar for this example.

13.11 Conclusion

The PLS estimate of the matrix of regression coefficients can be computed as

$$B_{\mathrm{PLS}} = W(W^T X^T XW)^{-1} W^T X^T Y, \tag{13.180}$$

where $W \in \mathbb{R}^{r \times a}$ is a weight matrix. If the weight matrix W is square and non-singular then this expression reduces to the ordinary least squares estimate.

The PLS algorithm is in general not optimal with respect to (minimizing) the Squared Prediction error, i.e., the PLS estimate of the regression coefficients gives in general not minimum of $\| Y - XB_{\mathrm{PLS}} \|_F$. This can be shown by counterexample. The optimal solution can be written in terms of an optimal weight matrix W which is derived and presented in Section (13.9).

References

De Moor, B, and J. David (199?). Total least squares and the algebraic Riccati equation. Katholieke Universiteit Leuven, B-3001 Leuven, Belgium. Internal report.

Fierro, R. D., G. H. Golub, P. C. Hansen and D. P. O'Leary (199?) Regularization by truncated total least squares.

Hansen, P. C. (1992). Regulatrization Tools. A Matlab Package for Analysis and Solution of Discrete Ill-Posed Problems. Danish Computing Centre for Research and Education, DK-2800 Lyngby, Denmark.

Höskuldsson, A. (1996). *Prediction Methods in Science and technology*, COLOURSCAN Warsaw, ISBN 87-985941-0-9.

13.12 Appendix: proofs

13.12.1 Proof of Equation (13.115)

The Squared Prediction error (SPE) can be expressed as

$$\begin{aligned} V_N(B) = \| Y - T\bar{C}^T \|_F^2 &= \mathrm{tr}\{(Y - T\bar{C}^T)^T(Y - T\bar{C}^T)\} \\ &= \mathrm{tr}\{(Y^T - CS\Lambda_t^{-1}T^T)(Y - T\Lambda_t^{-T}S^TC^T)\}, \end{aligned} \tag{13.181}$$

where we have used Equation (13.78). We have

$$V_N(B) = \| Y - T\bar{C}^T \|_F^2 = \mathrm{tr}(Y^TY) - 2\mathrm{tr}(CS\Lambda_t^{-1}T^TY) + \mathrm{tr}(CS\Lambda_t^{-1}T^TT\Lambda_t^{-T}S^TC^T), \tag{13.182}$$

and

$$V_N(B) = \mathrm{tr}(Y^TY) - 2\mathrm{tr}(S\Lambda_t^{-1}T^TYC) + \mathrm{tr}(\Lambda_t^{-1}S^TC^TCS) \tag{13.183}$$

We have from Definition 13.2 and Remark 13.4 that the diagonal of matrix $M = T^TYC = T^T\tilde{U}$ is equal to the diagonal of matrix S. We also have that the matrix C^TC have ones on the diagonal, S is a diagonal matrix and Λ_t is a diagonal matrix. This gives

$$V_N(B) = \mathrm{tr}(Y^TY) - 2\mathrm{tr}(S^2\Lambda_t^{-1}) + \mathrm{tr}(\Lambda_t^{-1}S^2) = \mathrm{tr}(Y^TY) - \mathrm{tr}(\Lambda_t^{-1}S^2) \tag{13.184}$$

QED.

Note that a general condition for PLS is given by

$$SC^T = T^TY. \tag{13.185}$$

This condition is to our knowledge new. Equation (13.185) can be derived by putting the gradient matrix $dV_N(B)/d\mathrm{tr}(C) = 0$, i.e., putting the gradient matrix of (13.183) to zero. Using this condition we derive the following expression for the decomposition of Y

$$Y = T\Lambda_t^{-1}T^TY + F. \tag{13.186}$$

Hence, the projection of the row space of Y^T onto the row space of T^T is equal to

$$Y^T/T^T = Y^TT\Lambda_t^{-1}T^T, \tag{13.187}$$

and, similarly

$$T\backslash Y = T\Lambda_t^{-1}T^TY = T_oT_o^TY. \tag{13.188}$$

Using (13.185) the SPE can be written as

$$V_N = \mathrm{tr}(Y^TY) - \mathrm{tr}(Y^TT\Lambda_t^{-1}T^TY). \tag{13.189}$$

The second term on the right hand side is related to the SVD of $\Lambda_t^{-\frac{1}{2}} T^T Y$. Define

$$\Lambda_t^{-\frac{1}{2}} T^T Y = T_o^T Y = \tilde{\mathcal{U}}\tilde{\mathcal{S}}\tilde{\mathcal{V}}^T. \tag{13.190}$$

Then,

$$V_N = \mathrm{tr}(Y^T Y) - \sum_{i=1}^{\min(a,m)} \tilde{s}_i^2. \tag{13.191}$$

Note also the following alternative expression for the PLS regression matrix

$$B_{\mathrm{PLS}} = W(P^T W)^{-1}(T^T T)^{-1} T^T Y. \tag{13.192}$$

A proof is given as follows

$$T\backslash Y = X B_{\mathrm{PLS}} = T\Lambda_t^{-1} T^T Y = \underbrace{TP^T}_{X} \underbrace{W(P^T W)^{-1}(T^T T)^{-1} T^T Y}_{B_{\mathrm{PLS}}}. \tag{13.193}$$

13.12.2 The PLS algorithm in terms of the W weight matrix

The results in this section is of interest sinse it shows that the entire PLS algorithm is dependent on the weight matrix W, only. We have the following results.

The PLS prediction of Y can be expressed as

$$\hat{Y} = XW(W^T X^T X W)^{-1} W^T X^T Y. \tag{13.194}$$

This result can be proved by combining $\hat{Y} = T(T^T T)^{-1} T^T Y$ and $T = XW(T^T XW)^{-1}\Lambda_t$.

Comparing this with $\hat{Y} = X B_{\mathrm{PLS}}$ gives

$$B_{\mathrm{PLS}} = W(W^T X^T X W)^{-1} W^T X^T Y. \tag{13.195}$$

Similarly as (13.194), the PLS prediction of X can be expressed as

$$\hat{X} = XW(W^T X^T X W)^{-1} W^T X^T X. \tag{13.196}$$

The above result are important since it shows that the PLS estimate of B can be computed from X, Y and the weight matrix W, only. The only problem is how to compute the weight matrix W. The PLS techique is only one approach.

Note also that if W is an arbitrary but square non-singular matrix, then (13.195) reduces to the ordinary least squares (LS) solution. This shows that the PLS estimate is equivalent to the LS solution when the number of components is $a = r$.

An expression for the rank one reduction of X in PLS can be derived from the equation for the residual

$$G = X - XW(W^T X^T XW)^{-1} W^T X^T X. \tag{13.197}$$

Similarily, for the outputs

$$F = Y - XW(W^T X^T XW)^{-1} W^T X^T Y. \tag{13.198}$$

The consequense of this is that the SPE is

$$\begin{aligned} V_N(B) &= \parallel Y - \hat{Y} \parallel_F^2 = \parallel Y - XB_{\text{PLS}} \parallel_F^2 \\ &= \text{tr}(Y^T Y) - \text{tr}(\hat{Y}^T \hat{Y}) \\ &= \text{tr}(Y^T Y) - \text{tr}(Y^T XW(W^T X^T XW)^{-1} W^T X^T Y). \end{aligned} \tag{13.199}$$

13.13 Matlab code: The PLS1 algorithm

13.14 Matlab code: The PLS2 algorithm

13.15 Matlab code: The PLS3 algorithm

Appendix A

Proof of the ESSM

A.1 Proof of the ESSM

The common equation in subspace identification is derived from the linear state space model and given by

$$\tilde{y}_{k|L} = \mathcal{O}_L x_k + \mathcal{H}_L^d \tilde{u}_{k|L} \tag{A.1}$$

The scalar positive integer parameter L can be viewed as the horizon necessary for the observability matrix $\mathcal{O}_L$ to be of rank n. Hence $L \geq n - \text{rank}(D) + 1$ in order for the state space vector x_k to be identified from Equation (A.1). Thus it make sense to define L as the *identification horizon*.

Equation (A.1) can be used to predict M steps into the future. Assume $M \geq 1$. We have

$$\tilde{y}_{k+M|L} = \mathcal{O}_L x_{k+M} + \mathcal{H}_L^d \tilde{u}_{k+M|L} \tag{A.2}$$

The state space vector in the future x_{k+M} is related to the present state x_k through the state equation. Hence we have

$$x_{k+M} = A^M x_k + \mathcal{C}_M^d \tilde{u}_{k|M} \tag{A.3}$$

Substituting Equation (A.3) into (A.2) gives

$$\tilde{y}_{k+M|L} = \mathcal{O}_L A^M x_k + \mathcal{O}_L \mathcal{C}_M^d \tilde{u}_{k|M} + \mathcal{H}_L^d \tilde{u}_{k+M|L} \tag{A.4}$$

The present state x_k is determined from Equation (A.1) and given by

$$x_k = \mathcal{O}^\dagger (\tilde{y}_{k|L} - \mathcal{H}_L^d \tilde{u}_{k|L}) \tag{A.5}$$

Finally, substituting x_k defined in equation (A.5) into (A.4) gives

$$\tilde{y}_{k+M|L} = \mathcal{O}_L A^M \mathcal{O}_L^\dagger (\tilde{y}_{k|L} - \mathcal{H}_L^d \tilde{u}_{k|L}) + \mathcal{O}_L \mathcal{C}_M^d \tilde{u}_{k|M} + \mathcal{H}_L^d \tilde{u}_{k+M|L} \tag{A.6}$$

which can be written as

$$\begin{aligned}
\tilde{y}_{k+M|L} &= \tilde{A}_L^M \tilde{y}_{k|L} + \tilde{B}_{L|L+M} \ \tilde{u}_{k|L+M} + \tilde{C}_{L|L+M} \ \tilde{e}_{k|L+M} && \text{(A.7)} \\
\tilde{B}_{L|L+M} &= \left[\begin{array}{cc} \mathcal{O}_L \mathcal{C}_M^d & \mathcal{H}_L^d \end{array} \right] - \left[\begin{array}{cc} \tilde{A}_L^M \mathcal{H}_L^d & 0_{Lm \times Mr} \end{array} \right] && \text{(A.8)} \\
\tilde{C}_{L|L+M} &= \left[\begin{array}{cc} \mathcal{O}_L \mathcal{C}_M^s & \mathcal{H}_L^s \end{array} \right] - \left[\begin{array}{cc} \tilde{A}_L^M \mathcal{H}_L^s & 0_{Lm \times Mm} \end{array} \right] && \text{(A.9)} \\
\tilde{A}_L^M &= \mathcal{O}_L A^M \mathcal{O}_L^\dagger && \text{(A.10)}
\end{aligned}$$

The ESSM model correspond to Equation (A.7) with $M = 1$ and $k \geq L$. The past inputs and outputs, $0 \leq k < L$, is used as instruments to remove future noise.

A.2 Discussion

An interesting result is obtained by choosing $M = L$. In this case the toepliz matrix $\mathcal{H}_L^d$ can be determined directly if the inputs are sufficiently exiting.

Consider the case in which $M = L$, then

$$\begin{aligned}
Y_{k+L|L} &= \tilde{A}_L^L Y_{k|L} + \tilde{B}_L U_{k|2L} + \tilde{C}_L E_{k|2L} && \text{(A.11)} \\
\tilde{B}_L &= \left[\begin{array}{cc} \mathcal{O}_L \mathcal{C}_L - \tilde{A}_L^L \mathcal{H}_L^d & \mathcal{H}_L^d \end{array} \right] && \text{(A.12)} \\
\tilde{C}_L &= \left[\begin{array}{cc} \mathcal{O}_L \mathcal{C}_L - \tilde{A}_L^L \mathcal{H}_L^s & \mathcal{H}_L^s \end{array} \right] && \text{(A.13)}
\end{aligned}$$

A.3 Selector matrices

Our experience is that the choice of the parameter L (number of block rows in the extended observability matrix) should be chosen as small as possibile. I.e. close to the minimal value in order to ensure observability.

The reason for this is twofold. First, the computational expence will increase when L is increased. Second, for porely exiting input signals, we avoid (pseudo) inversion of badly conditioned Hankel matrices when L is small. This will usually influence on the variance of the parameter estimates.

$$\begin{aligned}
S_y \tilde{y}_{k+M|L} &= \tilde{A}_p^M S_y \tilde{y}_{k|L} + \tilde{B}_p S_u \tilde{u}_{k|L+M} + \tilde{C}_p S_e \tilde{e}_{k|L+M} && \text{(A.14)} \\
\tilde{A}_p^M &= S_y \tilde{A}_L^M S_y^\dagger && \text{(A.15)} \\
\tilde{B}_p &= S_y \tilde{B}_L S_u^\dagger && \text{(A.16)} \\
\tilde{C}_p &= S_y \tilde{C}_L S_e^\dagger && \text{(A.17)}
\end{aligned}$$

A.4 A study of the predictor $Z^d_{J|L+1}$ for $Y_{J|L+1}$.

A possible subspace based predictor for the matrix $Y_{J|L+1}$ of future outputs is given by the matrix $Z^d_{J|L+1}$ as defined in Di Ruscio (1997). I.e.,

$$\begin{aligned} Z^d_{J|L+1} &= Y_{J|L+1}/\begin{bmatrix} U_{J|L+g} \\ W_1 \end{bmatrix} \\ &= \overbrace{Y_{J|L+1}\begin{bmatrix} U^T_{J|L+g} & W_1^T \end{bmatrix}(\begin{bmatrix} U_{J|L+g} \\ W_1 \end{bmatrix}\begin{bmatrix} U^T_{J|L+g} & W_1^T \end{bmatrix})^{-1}}^{B_z}\begin{bmatrix} U_{J|L+g} \\ W_1 \end{bmatrix}, \end{aligned} \tag{A.18}$$

where

$$W_1 = \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}, \tag{A.19}$$

and B_z is the matrix of regression coefficients from the corresponding LS problem. For simplicity of notation we will skip the subscript indexes in the following. First let us compute the regression coefficient matrix

$$B_z = \begin{bmatrix} YU^T & YW^T \end{bmatrix}\begin{bmatrix} UU^T & UW^T \\ WU^T & WW^T \end{bmatrix}^{-1}. \tag{A.20}$$

The inverse of a partitioned matrix is (Kailath (1980))

$$\begin{bmatrix} UU^T & UW^T \\ WU^T & WW^T \end{bmatrix}^{-1} = \begin{bmatrix} (UU^T)^{-1} + (UU^T)^{-1}UW^T\Delta^{-1}WU^T(UU^T)^{-1} & -(UU^T)^{-1}UW^T\Delta^{-1} \\ -\Delta^{-1}WU^T(UU^T)^{-1} & \Delta^{-1} \end{bmatrix} \tag{A.21}$$

where Δ is the Schur complement of UU^T given by

$$\Delta = WW^T - WU^T(UU^T)^{-1}UW^T = WU^\perp W^T \tag{A.22}$$

where we have used index f for future. This gives

$$B_z = \begin{bmatrix} B_1 & B_2 \end{bmatrix} \tag{A.23}$$

where

$$\begin{aligned} B_1 &= YU^T((UU^T)^{-1} + (UU^T)^{-1}UW^T\Delta^{-1}WU^T(UU^T)^{-1}) - YW^T\Delta^{-1}WU^T(UU^T)^{-1} \\ &= (Y - YU^\perp W^T(WU^\perp W^T)^{-1}W)U^T(UU^T)^{-1}, \qquad (A.24) \\ B_2 &= -YU^T(UU^T)^{-1}UW^T\Delta^{-1} + YW^T\Delta^{-1}, \\ &= YU^\perp W^T(WU^\perp W^T)^{-1}. \qquad (A.25) \end{aligned}$$

Hence, $Z^d_{J|L+1}$ is the sum of two parts, i.e.,

$$\begin{aligned} Z^d_{J|L+1} &= Y_{J|L+1}/_{W_1}U_{J|L+g} + Y_{J|L+1}/_{U_{J|L+g}}W_1 = B_1U + B_2W \\ &= (Y - YU^\perp W^T(WU^\perp W^T)^{-1}W)/U + YU^\perp W^T(WU^\perp W^T)^{-1}W. \end{aligned} \tag{A.26}$$

Appendix B

Linear Algebra and Matrix Calculus

B.1 Trace of a matrix

The trace of a $n \times m$ matrix A is defined as the sum of the diagonal elements of the matrix, i.e.

$$tr(A) = \sum_{i=1}^{n} a_{ii} \tag{B.1}$$

We have the following trace operations on two matrices A and B of apropriate dimensions

$$tr(A^T) = tr(A) \tag{B.2}$$

$$tr(AB^T) = tr(A^T B) = tr(B^T A) = tr(BA^T) \tag{B.3}$$

$$tr(AB) = tr(BA) = tr(B^T A^T) = tr(A^T B^T) \tag{B.4}$$

$$tr(A \pm B) = tr(A) \pm tr(B) \tag{B.5}$$

B.2 Gradient matrices

$$\frac{\partial}{\partial X} \quad tr[X] = I \tag{B.6}$$

$$\frac{\partial}{\partial X} \quad tr[AX] = A^T \tag{B.7}$$

$$\frac{\partial}{\partial X} \quad tr[AX^T] = A \tag{B.8}$$

$$\frac{\partial}{\partial X} \quad tr[AXB] = A^T B^T \tag{B.9}$$

$$\frac{\partial}{\partial X} \quad tr[AX^T B] = BA \tag{B.10}$$

$$\frac{\partial}{\partial X} \quad tr[XX] = 2X^T \tag{B.11}$$

$$\frac{\partial}{\partial X} \quad tr[XX^T] = 2X \tag{B.12}$$

$$\frac{\partial}{\partial X} \quad tr[X^n] = n(X^{n-1})^T \tag{B.13}$$

$$\frac{\partial}{\partial X} \quad tr[AXBX] = A^T X^T B^T + B^T X^T A^T \tag{B.14}$$

$$\frac{\partial}{\partial X} \quad tr[e^{AXB}] = (Be^{AXB}A)^T \tag{B.15}$$

$$\frac{\partial}{\partial X} tr[XAX^T] = 2XA, \text{ if } A = A^T \tag{B.16}$$

$$\frac{\partial}{\partial X^T} \quad tr[AX] = A \tag{B.17}$$

$$\frac{\partial}{\partial X^T} \quad tr[AX^T] = A^T \tag{B.18}$$

$$\frac{\partial}{\partial X^T} \quad tr[AXB] = BA \tag{B.19}$$

$$\frac{\partial}{\partial X^T} \quad tr[AX^TB] = A^T B^T \tag{B.20}$$

$$\frac{\partial}{\partial X^T} \quad tr[e^{AXB}] = Be^{AXB}A \tag{B.21}$$

B.3 Derivatives of vector and quadratic form

The derivative of a vector with respect to a vector is a matrix. We have the following identities:

$$\frac{\partial x}{\partial x^T} \quad = I \tag{B.22}$$

$$\frac{\partial}{\partial x} \quad (x^T Q) = Q \tag{B.23}$$

$$\frac{\partial}{\partial x} \quad (Qx) = Q^T \tag{B.24}$$

$$\tag{B.25}$$

The derivative of a scalar with respect to a vector is a vector. We have the following identities:

$$\frac{\partial}{\partial x} \quad (y^T x) = y \tag{B.26}$$

$$\frac{\partial}{\partial x} \quad (x^T x) = 2x \tag{B.27}$$

$$\frac{\partial}{\partial x} \quad (x^T Q x) = Qx + Q^T x \tag{B.28}$$

$$\frac{\partial}{\partial x} \quad (y^T Q x) = Q^T y \tag{B.29}$$

Note that if Q is symmetric then

$$\frac{\partial}{\partial x} \quad (x^T Q x) = Qx + Q^T x = 2Qx. \tag{B.30}$$

B.4 Matrix norms

The most frequently used matrix norm in numerical analysis and linear algebra is the Frobenius norm (the F-norm).

The trace of the matrix product $A^T A$ is related to the Frobenius norm of A as follows

$$\|A\|_F^2 = \mathrm{tr}(A^T A), \tag{B.31}$$

where $A \in \mathbb{R}^{N\times m}$.

A frequently used notation and expression for the matrix Frobenius norm is also

$$\|A\|_F^2 = (\sum_{i=1}^{N}\sum_{j=1}^{m} a_{ij})^{\frac{1}{2}}, \tag{B.32}$$

i.e. equal the square root of the sum of all elements.

An important property of the Frobenius norm is that it is invariant with respect to orthogonal transformation. Assume given two orthogonal matrices Q and Z with appropriate dimensions we have that

$$\|A\|_F = \|QAZ\|_F. \tag{B.33}$$

B.5 Linearization

Given a vector function $f(x) \in \mathbb{R}^m$ where $x \in \mathbb{R}^n$. The derivative of the vector f with respect to the row vector x^T is defined as

$$\frac{\partial f}{\partial x^T} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m\times n} \tag{B.34}$$

Given a non-linear differentiable state space model

$$\begin{aligned} \dot{x} &= f(x,u), && \text{(B.35)} \\ y &= g(x). && \text{(B.36)} \end{aligned}$$

A linearized model around the stationary points x_0 and u_0 is

$$\begin{aligned} \dot{\delta x} &= Ax + Bu, && \text{(B.37)} \\ \delta y &= Dx, && \text{(B.38)} \end{aligned}$$

where

$$\begin{aligned} A &= \frac{\partial f}{\partial x^T} \mid_{x_0,u_0}, && \text{(B.39)} \\ B &= \frac{\partial f}{\partial u^T} \mid_{x_0,u_0}, && \text{(B.40)} \\ D &= \frac{\partial g}{\partial x^T} \mid_{x_0,u_0}, && \text{(B.41)} \end{aligned}$$

and where

$$\begin{aligned} x &= x - x_0, && \text{(B.42)} \\ u &= u - u_0. && \text{(B.43)} \end{aligned}$$

B.6 Kronecer product matrices

Given a matrix $X \in \mathbb{R}^{N \times r}$. Let I_m be the $(m \times m)$ identity matrix. Then

$$(X \otimes I_m)^T = X^T \otimes I_m, \tag{B.44}$$

$$(I_m \otimes X)^T = I_m \otimes X^T. \tag{B.45}$$

Appendix C

Optimization of the DSR algorithm

C.1 Existence of an optimal solution

For a given pair (D, A) there exists a canonical form (D_c, A_c) where D_c consist of only ones and zeroes and with a minimal number of free parameters in A_c. Denote the free parameters as

$$\theta_A = \text{css}(A) \tag{C.1}$$

where $\text{css}(\cdot)$ is defined as the column structure string operator. See Di Ruscio (1993) p. 121.

The problem of identifying the poles as well as the system matrices D and A is reduced to solving the following optimization problem.

$$\min_{\theta_A} \|R_{42} - O_L A (O_L^T O_L)^{-1} O^T R_{32}\|_F \tag{C.2}$$

where $\|\cdot\|_F$ is the matrix Frobenius norm.

The size of this optimization problem can be further reduced. Define the SVD

$$R_{32} = USV^T \tag{C.3}$$

The Frobenius norm is invariant under orthogonal transformations. See Golub and Loan (1883) p. 15. Then we have

$$\min_{\theta_A} \|R_{42}V - O_L A (O_L^T O_L)^{-1} O^T US\|_F \tag{C.4}$$

Monte Carlo simulations shows that this solution is optimal and different from the original DSR algorithm.

Note also that A can be computed from the following least squares problem

$$cs(R_{42}) = \left[\; ((O_L^T O_L)^{-1} O^T R_{32})^T \otimes O_L \; \right] cs(A) \tag{C.5}$$

C.2 Optimized algorithm for computing the B and E matrices

In the DSR algorithm the system order and the matrices D and A are computed in a first step. There are many alternatives for computing the system matrices B and E from known D and A. One alternative to the one presented in Di Ruscio (1995) is given in the following. A similar strategy for computing the B and E matrices for SISO systems is presented in the covariance based method in Di Ruscio (1993) p. 50. Monte Carlo simulations of combined deterministic and stochastic systems shows that the algorithm improves the accuracy of the B E estimates compared to the original DSR algorithm. The method is optimal in a least squares sense, i.e., the parameters in B and E are computed from a least squares optimal solution. Note that the algorithms gives the same result in the purely deterministic case.

From the DSR algorithm we have the matrix equation

$$M = \tilde{B}_L R_{11}, \tag{C.6}$$

where

$$M \overset{\text{def}}{=} R_{41} - \tilde{A}_L R_{31} \ \in \ \mathbb{R}^{Lm\times(L+g)r} \tag{C.7}$$

and

$$\tilde{A}_L = O_L A (O_L^T O_L)^{-1} O_L^T \ \in \ \mathbb{R}^{Lm\times Lm}. \tag{C.8}$$

The matrices $R_{41} \in \mathbb{R}^{Lm\times(L+g)r}$, $R_{31} \in \mathbb{R}^{Lm\times(L+g)r}$ and $R_{11} \in \mathbb{R}^{(L+g)r\times(L+g)r}$ in Equations (C.6) and (C.7) are submatrices from the lower Left, Q-orthogonal (LQ) decomposition.

The matrix equation (C.6) can be solved for $\tilde{B}_L$ for known matrices M and R_{11} provided R_{11} is non-singular. The matrices B and E can be computed from $\tilde{B}_L$ when A and D are known. In Di Ruscio (1995) a variant of this strategy which only have to invert a submatrix of R_{11} of size $(n - \text{rank}(D) + 1 + g)r$ is presented.

In the following an alternative and optimal method for computing the B and E matrices are presented. For given system matrices D and A the matrix $\tilde{B}_L$ is a function of the unknown matrices B and E. We will show that Equation (C.6) can be written as a least squares problem for the unknown parameters in B and E.

In the following a proper state space model with $E \neq 0_{m\times r}$ is assumed. From the definition we recall that $g = 1$ for this case. Define

$$N \overset{\text{def}}{=} \sum_{i=1}^{L+g} R_i^T \otimes (E_{i-1} - \tilde{A}_L E_i) \ \in \ \mathbb{R}^{Lm(L+g)r\times(n+gm)r} \tag{C.9}$$

where $\otimes$ denotes the Kronecker tensor product. The matrices R_i and E_i are defined below. We then have the following least squares problem for computing the B and E matrices.

$$cs(M) = Ncs(\begin{bmatrix} B \\ E \end{bmatrix}) \tag{C.10}$$

where $cs(\cdot)$ is the column string operator, i.e. a column vector constructed by stacking the columns of a matrix on each other. The matrix N is determined from the term $\tilde{B}_L R_{32}$ by extensive use of the identity

$$cs(AXB) = (B^T \otimes A)cs(X)$$

for the column string operation of the product of a triple matrices A, X and B of appriopriate dimensions (Vetter (1970)).

Then we have the least squares solution

$$cs(\begin{bmatrix} B \\ E \end{bmatrix}) = N^\dagger cs(M) \tag{C.11}$$

where $N^\dagger = (N^T N)^{-1} N^T$ is the Moore-Penrose pseudo-inverse of N.

The matrices

$$R_i \in \mathbb{R}^{r \times (L+g)r} \;\forall\; i = 1, \cdots, L+g$$

are block rows in the $R_{11} \in \mathbb{R}^{(L+g)r \times (L+g)r}$ matrix. The matrices

$$E_{i-1} \in \mathbb{R}^{Lm \times (n+mg)} \;\forall\; i = 1, \cdots, L+g$$

are defined as follows

$$E_0 = \begin{bmatrix} O_L & 0_{Lm \times m} \end{bmatrix}, \quad E_{L+1} = 0_{Lm \times (n+m)}, \tag{C.12}$$

$$E_1 = \begin{bmatrix} 0_{m \times n} & I_{m \times m} \\ D & 0_{m \times m} \\ DA & 0_{m \times m} \\ \vdots & \vdots \\ DA^{L-2} & 0_{m \times m} \end{bmatrix}, E_2 = \begin{bmatrix} 0_{m \times n} & 0_{m \times m} \\ 0_{m \times n} & I_{m \times m} \\ D & 0_{m \times m} \\ \vdots & \vdots \\ DA^{L-3} & 0_{m \times m} \end{bmatrix}, E_L = \begin{bmatrix} 0_{m \times n} & 0_{m \times m} \\ 0_{m \times n} & 0_{m \times m} \\ 0_{m \times n} & 0_{m \times m} \\ \vdots & \vdots \\ 0_{m \times n} & I_{m \times m} \end{bmatrix} \tag{C.13}$$

Some examples are presented in the following in order to illustrate the above method for computing the B and E matrices.

Example C.1
Considder the equation

$$Z = H_3^d U_{0|3} \tag{C.14}$$

and the problem of computing B and E from known matrices Z, $U_{0|3}$, D and A. We have

$$Z = \overbrace{\begin{bmatrix} E & 0_{m \times r} & 0_{m \times r} \\ DB & E & 0_{m \times r} \\ DAB & DB & E \end{bmatrix}}^{H_3^d} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} \tag{C.15}$$

and

$$Z = \begin{bmatrix} 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \\ DA & 0_{m\times m} \end{bmatrix} \begin{bmatrix} B \\ E \end{bmatrix} U_1 + \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \end{bmatrix} \begin{bmatrix} B \\ E \end{bmatrix} U_2 + \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & I_{m\times m} \end{bmatrix} \begin{bmatrix} B \\ E \end{bmatrix} U_3 \tag{C.16}$$

Then we have

$$cs(Z) = Ncs(\begin{bmatrix} B \\ E \end{bmatrix}) \tag{C.17}$$

where

$$N = U_1^T \otimes \begin{bmatrix} 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \\ DA & 0_{m\times m} \end{bmatrix} + U_2^T \otimes \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \end{bmatrix} + U_3^T \otimes \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & I_{m\times m} \end{bmatrix} \tag{C.18}$$

Example C.2
Considder the ESSM with $L = 1$ and $g = 1$, i.e.

$$Y_{1|1} = \tilde{A}_1 Y_{0|1} + \tilde{B}_1 U_{0|2} \tag{C.19}$$

Assume that the input and output data as well as $\tilde{A}_1$ are known. Define

$$M = Y_{1|1} - \tilde{A}_1 Y_{0|1} \tag{C.20}$$

Then we have the matrix equation

$$M = \tilde{B}_1 U_{0|2} \tag{C.21}$$

Problem: *Compute B and E from known matrices M, $\tilde{A}_1$ and $U_{0|2}$.*
Solution: *We have*

$$M = \overbrace{(\begin{bmatrix} DB & E \end{bmatrix} - \tilde{A}_1 \begin{bmatrix} E & 0_{m\times r} \end{bmatrix})}^{\tilde{B}_1} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} \tag{C.22}$$

The matrix $\tilde{B}_1$ can be written as follows

$$\tilde{B}_1 = \begin{bmatrix} DB - \tilde{A}_1 E & E \end{bmatrix} \tag{C.23}$$

From this we have

$$M = (E_0 - \tilde{A}_1 E_1) \begin{bmatrix} B \\ E \end{bmatrix} U_1 + E_1 \begin{bmatrix} B \\ E \end{bmatrix} U_2 \tag{C.24}$$

where

$$E_0 = \begin{bmatrix} D & 0_{m\times m} \end{bmatrix}, \quad E_1 = \begin{bmatrix} 0_{m\times n} & I_{m\times m} \end{bmatrix}. \tag{C.25}$$

This gives the least squares problem

$$cs(M) = Ncs(\begin{bmatrix} B \\ E \end{bmatrix}) \tag{C.26}$$

where

$$N = U_1^T \otimes (E_0 - \tilde{A}_1 E_1) + U_2^T \otimes E_1 \tag{C.27}$$

Example C.3
Considder the ESSM

$$Y_{1|2} = \tilde{A}_2 Y_{0|2} + \tilde{B}_2 U_{0|3} \tag{C.28}$$

Assume that the input and output data as well as $\tilde{A}_2$ are known. Define

$$M = Y_{1|2} - \tilde{A}_2 Y_{0|2} \tag{C.29}$$

Then we have

$$M = \tilde{B}_2 U_{0|3} \tag{C.30}$$

Problem: *Compute B and E from known matrices M, $\tilde{A}_2$ and $U_{0|3}$.*
Solution: *We have*

$$M = \overbrace{(\begin{bmatrix} DB & E & 0_{m\times r} \\ DAB & DB & E \end{bmatrix} - \tilde{A}_2 \begin{bmatrix} E & 0_{m\times r} & 0_{m\times r} \\ DB & E & 0_{m\times r} \end{bmatrix})}^{\tilde{B}_2} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix} \tag{C.31}$$

The matrix $\tilde{B}_2$ can be written as follows

$$\tilde{B}_2 = \begin{bmatrix} \begin{bmatrix} DB \\ DAB \end{bmatrix} - \tilde{A}_2 \begin{bmatrix} E \\ DB \end{bmatrix} & \begin{bmatrix} E \\ DB \end{bmatrix} - \tilde{A}_2 \begin{bmatrix} 0_{m\times r} \\ E \end{bmatrix} & \begin{bmatrix} 0_{m\times r} \\ E \end{bmatrix} \end{bmatrix} \tag{C.32}$$

From this we have

$$M = (E_0 - \tilde{A}_2 E_1) \begin{bmatrix} B \\ E \end{bmatrix} U_1 + (E_1 - \tilde{A}_2 E_2) \begin{bmatrix} B \\ E \end{bmatrix} U_2 + E_2 \begin{bmatrix} B \\ E \end{bmatrix} U_3 \tag{C.33}$$

where

$$E_0 = \begin{bmatrix} D & 0_{m\times m} \\ DA & 0_{m\times m} \end{bmatrix}, \quad E_1 = \begin{bmatrix} 0_{m\times n} & I_{m\times m} \\ D & 0_{m\times m} \end{bmatrix}, \quad E_2 = \begin{bmatrix} 0_{m\times n} & 0_{m\times m} \\ 0_{m\times n} & I_{m\times m} \end{bmatrix} \tag{C.34}$$

This gives the least squares problem

$$cs(M) = Ncs(\begin{bmatrix} B \\ E \end{bmatrix}) \tag{C.35}$$

where

$$N = U_1^T \otimes (E_0 - \tilde{A}_2 E_1) + U_2^T \otimes (E_1 - \tilde{A}_2 E_2) + U_3^T \otimes E_2 \tag{C.36}$$

Remark C.1
We conclude this section by commenting that the $cs(\cdot)$ operation of $R_{42} = \tilde{A}_L R_{32}$ is given by

$$cs(R_{42}) = [((O_L O_L)^{-1} O_L^T R_{32})^T \otimes O_L] cs(A) \tag{C.37}$$

which define a least squares problem for computing the system matrix A for known observability matrix. Note that the column space of R_{32} coincide with the column space of the observability matrix, i.e. the observability matrix can be estimated as the column space of R_{32}.

Appendix D

D-SR Toolbox for MATLAB

D.1 The DSR function for use with MATLAB

THE ACRONYM:
Deterministic and Stochastic system identification and Realization (DSR).

PURPOSE:
Given the output data matrix Y and the input data matrix U. DSR estimate the system order n, the matrices A, B, D, E, CF, F and the initial state vector $x0$ in the discrete time, combined deterministic and stochastic dynamic model, on innovations form:

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + Ce_t, \quad x_{t=0} = x0, & \text{(D.1)} \\ y_t &= Dx_t + Eu_t + e_t, & \text{(D.2)} \end{aligned}$$

where t is discrete time, $C = CF \cdot F^{-1}$ is the Kalman filter gain matrix and $\mathrm{E}(e_t e_t^T) = FF^T$ is the innovations noise covariance matrix.

SYNOPSIS:

$$\begin{aligned} &[A, B, D, E, CF, F, x0] = \mathrm{dsr}(Y, U, L) \\ &[A, B, D, E, CF, F, x0] = \mathrm{dsr}(Y, U, L, g) \\ &[A, B, D, E, CF, F, x0] = \mathrm{dsr}(Y, U, L, g, J, M, n) \end{aligned}$$

PARAMETERS ON INPUT:

Y – An $(N \times m)$ matrix with output data/observations. N is the number of observations and m is the number of output variables.
U – An $(N \times r)$ matrix with input data/observations. r is the number of input variables.
L – Integer. Number specifying the future horizon used for predicting the system order. Choose $L > 0$ such that the assumed system order satisfy $n \leq Lm$. $L = 1$ is default.

OPTIONAL INPUT PARAMETERS:
DSR has four optional integer parameters. These parameters is for advanced use. A description is given below.

g – Integer. Optional model structure parameter. $g = 1$ is default. If $g = 0$ then a model with $E = 0_{m \times r}$ is estimated. If $g = 1$ then E is estimated.
J – Integer. Number defining the past horizon used to define the instrumental variables used to remove future noise. $J = L$ is default and recommended.
M – Integer. With $M = 1$ (default) a simple method for computing CF and F is used. A more computational expensive method is used when $M \neq 1$.
n – Integer. Optional specification of model order, $0 < n \leq Lm$.