KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT DER TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee)

SUBSPACE METHODS FOR IDENTIFICATION AND CONTROL OF LINEAR AND BILINEAR SYSTEMS

Promotor:
Prof. B. De Moor

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Wouter FAVOREEL

November 1999

KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT DER TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee)

SUBSPACE METHODS FOR IDENTIFICATION AND CONTROL OF LINEAR AND BILINEAR SYSTEMS

Jury:
Prof. J. Delrue, voorzitter
Prof. B. De Moor, promotor
Prof. J. Vandewalle, vice-decaan
Prof. M. Moonen
Prof. A. Bultheel
Prof. M. Gevers (UC Louvain-la-Neuve)
Prof. M. Verhaegen (TU Twente)

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Wouter FAVOREEL

U.D.C. 621.397.3

November 1999

D/1999/7515/47

ISBN 90-5682-216-0

Voorwoord

Uiteindelijk is het dan zover: m'n doctoraat is af. Als ik bij deze gedachte blijf stilstaan dan moet ik toch wel even slikken. Hoewel ik een gevoel heb van "oef, 't is voorbij" moet ik zeggen dat het vier fantastische jaren waren. De toffe sfeer in de groep, de volleybal over de middag, de cake-time op vrijdag, de laatste pint op de Oude Markt, de Sista-weekends, ... Ik denk, en ik hoop, dat het me allemaal nog een hele tijd zal bijblijven. Het waren niet alleen toffe jaren, maar ook uiterst nuttige jaren. De vorm van kritisch denken, analyseren en de zelfkennis die men verkrijgt tijdens het maken van een doctoraat zijn van onschatbare waarde. Uiteraard sta je voor zo'n taak nooit echt alleen maar word je altijd wel, op directe of indirecte wijze, bijgestaan. Daarom zou ik graag een aantal mensen bedanken.

Eerst en vooral mijn promotor Prof. Bart De Moor. Zijn voortdurende steun en motivatie waren onontbeerlijk.

Dank ook aan het leescomite, Prof. Marc Moonen en Prof. Joos Vandewalle. Hun kritisch oog en de nauwkeurigheid waarmee ze het werk gelezen hebben waren waardevol.

Verder hartelijk dank aan de leden van de Jury voor de tijd die ze vrijmaakten, hoe druk hun agenda ook is, om dit werk na te lezen en te zetelen in de Jury. De Voorzitter Prof. Delrue, Prof. Adhemar Bultheel, Prof. Michel Verhaegen en Prof. Michel Gevers. In het bijzonder dank ik Prof. Michel Gevers en Prof. Michel Verhaegen voor de aangename manier waarop we hebben kunnen samenwerken in het kader van deze thesis.

Uiteraard wens ik het I.W.T. te danken voor de financiële steun.

Natuurlijk waren er ook de collega's. Door hen is Sista wat Sista is, namelijk een toffe bende burgies. Graag zou ik alle Sista-ianen van harte willen bedanken voor de fijne werksfeer waarin ik gedurende vier jaar heb mogen werken. Hierbij zijn er toch een aantal mensen aan wie ik speciaal denk. Eerst en vooral Piet en Jeroen ofte de West-Vlaamse harde kern waarmee een echte West-Vlaming toch nog het best kan klappen. Verder de mannen van het eerste uur Bart DS,

Bart M, Eric, Ida, Johan S, Koen, Leentje & Philippe, Lieven, Peter DG, Rita, Willem, Yves en die van het tweede uur Frank, Geert L, Geert R, Johan C, Katleen, Katrien, Simon en Tony.

Verder was er m'n familie: pa en ma, Trui en Frank, Geertrui en Stefaan. Zij waren van de weinige niet-ingewijden die begrepen wat het betekent om te doctoreren. Dit was reeds een hele steun op zich.

Tenslotte dank ik Nathalie. Zij stond me steeds bij in goede en kwade tijden.

Wouter Favoreel

15 november 1999

Abstract

Part I: Subspace identification

The last decade, subspace system identification of linear systems has become a mature field of research. The success of linear subspace methods can, among other reasons, be attributed to their robustness when applied in difficult real-life industrial situations. In this thesis we will, instead of restricting ourselves to linear systems, consider a wider class of systems namely bilinear systems. The main advantage of bilinear systems is that they make a good trade-off between the precision of general nonlinear systems and the mathematical tractability of linear systems.

In the first part of this thesis, a new class of subspace algorithms is introduced for the identification of multivariable bilinear time-invariant systems. They are a direct generalization of the existing linear subspace algorithms.

Part II: Subspace-based control

Part II of this thesis focuses on control instead of on identification. We examine how subspace system identification algorithms can be used for control purposes. It turns out that there are a lot of analogies between subspace system identification and linear optimal control. Where as in the case of system identification one is interested in finding a plant model given measured input/output data, control-design is more interested in finding a control input given a plant model.

Using techniques from the field of subspace system identification, we derive a new class of control algorithms that allow for the calculation of an optimal controller of an unknown plant, directly from measurements of the inputs and the outputs of that plant, and thus without an intermediate plant model. Further the link with optimal and predictive control is made and we show how, using specific weights in the algorithm, the obtained subspace-based controller can be directly in balanced form.

Contents

Voorwoord **i**

Abstract **iii**

Contents **iv**

Nederlandse Samenvatting **ix**

1 Introduction **1**

1.1 Theme I: Subspace system identification 2
1.2 Theme II: Subspace-based controller design 6
1.3 Theme III: Bilinear systems 10
1.3.1 Sense and non-sense of bilinear systems 10
1.3.2 Bilinear identification 15
1.3.3 Bilinear control 16
1.4 Outline and contributions 17

I Subspace identification **23**

2 Linear subspace identification: An overview **25**

2.1 Problem description 25
2.2 Stochastic properties of linear systems 26
2.2.1 The deterministic and the stochastic subsystem 26
2.2.2 Forward innovation representation 27
2.3 The subspace structure of linear systems 30
2.3.1 System related matrices 30
2.3.2 Data block-Hankel matrices 31

2.3.3 Matrix input/output equations 32
2.3.4 Subspace projections 32
2.4 The three main steps in subspace identification 34
2.4.1 First step: LQ-decomposition 34
2.4.2 Second step: SV-decomposition 38
2.4.3 Third step: recovering the state-space matrices 40
2.5 Implementation: LQ-decomposition 45
2.6 Unifying framework 48
2.7 Comparison of different subspace identification algorithms . . . 48

3 Subspace identification of bilinear systems with white inputs 55
3.1 Introduction – problem description 55
3.2 Stochastic properties of bilinear systems 57
3.2.1 Deterministic and stochastic subsystem 57
3.2.2 Realization theory of bilinear systems 58
3.2.3 Kalman filter 65
3.3 Subspace structure of bilinear systems 66
3.3.1 System related matrices 67
3.3.2 Generalized data block-Hankel matrices 68
3.3.3 Matrix input/output equations 70
3.4 A bilinear subspace identification algorithm 71
3.4.1 First step: LQ-decomposition 71
3.4.2 Second step: SV-decomposition 73
3.4.3 Third step: recovering the state-space matrices 74
3.4.4 N4SID algorithm 75

4 Subspace identification of bilinear systems with general inputs 77
4.1 Introduction 77
4.2 Bilinear subspace identification 78
4.3 Algorithm 82
4.4 Implementation of bilinear subspace identification 85

5 Some examples 89
5.1 White inputs 89
5.1.1 Second order system 89
5.1.2 Binary distillation column 90
5.2 General inputs 93

5.2.1 A bilinear MIMO system 93
5.2.2 Distillation column 94
5.2.3 Linear vs. bilinear N4SID 95

II Subspace-based control 99

6 Model-free subspace-based LQG-design 101
6.1 The model-free LQG-control problem 101
6.2 LQG - The classical approach 104
6.3 LQG - A subspace approach 105
6.4 Algorithm 109
6.5 Equivalence between the two approaches 113
6.5.1 Infinite backward horizon 114
6.5.2 White inputs 114
6.6 Example 115

7 Closed-loop model-free subspace-based LQG-design 119
7.1 Introduction 119
7.2 Closed-loop subspace system identification 121
7.2.1 Problem definition 121
7.2.2 Brief overview of the theory 121
7.2.3 Algorithm 124
7.3 Closed-loop subspace-based LQG-controller 124
7.4 Example 126

8 Subspace-based balanced controller reduction 131
8.1 Introduction 131
8.2 Classical balanced controller reduction 132
8.2.1 Open-loop balanced model reduction 135
8.2.2 Closed-loop balanced controller reduction 136
8.3 Balanced subspace system identification 137
8.4 Subspace-based balanced controller reduction 139
8.4.1 Controller reduction 139
8.4.2 Subspace-based open-loop balanced controller reduction 143
8.4.3 Subspace-based closed-loop balanced controller reduction 145

9 Model-free subspace-based bilinear predictive control 147

9.1 Introduction – problem description 147
9.2 Model-free subspace-based predictive control for bilinear systems 149
9.3 Algorithm 152
9.4 Example 152

10 Conclusions and open problems **155**
10.1 Conclusions 155
10.2 Open problems 156

A Appendix **171**
A.1 Proof of Theorem 3.2.3 171
A.2 Proof of Theorem 3.4.1 172
A.3 Proof of Theorem 3.4.2 177
A.4 Proof of Theorem 4.2.2 178
A.5 Proof of Theorem 6.5.1 179
A.6 Proof of Theorem 8.3.1 182
A.7 Proof of Theorem 8.4.1 184
A.8 Proof of Theorem 8.4.2 186

Deelruimte-methoden voor de identificatie en controle van lineaire en bilineaire systemen

Nederlandse samenvatting

Hoofdstuk 1: Inleiding

Zoals de titel laat vermoeden, kan deze thesis opgesplitst worden in twee delen. Enerzijds beschouwen we identificatie en controle, anderzijds worden lineaire en bilineaire systemen behandeld. Op het eerste zicht lijkt dit een veel te algemene problematiek om in één enkele thesis te kunnen bestuderen. Alleen al op het gebied van lineaire systeemidentificatie werden reeds verscheidene boeken geschreven. Om daarnaast ook bilineaire systemen te behandelen, en bovendien ook de controle van dergelijke systemen te beschouwen, kan hopeloos lijken. Desalniettemin bestaat er een verband tussen deze ogenschijnlijk zeer verscheiden problemen. De rode draad in deze thesis is namelijk dat *deelruimte-technieken* gebruikt worden om een nieuwe klasse van algoritmen te ontwikkelen voor de identificatie en controle van zowel lineaire als bilineaire systemen. Onder deelruimte-technieken verstaan we de klasse van methoden waar, door middel van singuliere waarden ontbindingen van gestructureerde matrices, nuttige informatie uit beschikbare datasets wordt berekend.

Deze thesis is opgesplitst in twee delen. Het eerste deel bestudeert het deelruimte-identificatie probleem van bilineaire systemen. Gedurende het laatste decennium werd duidelijk dat deelruimte-identificatiemethoden een volwaardig

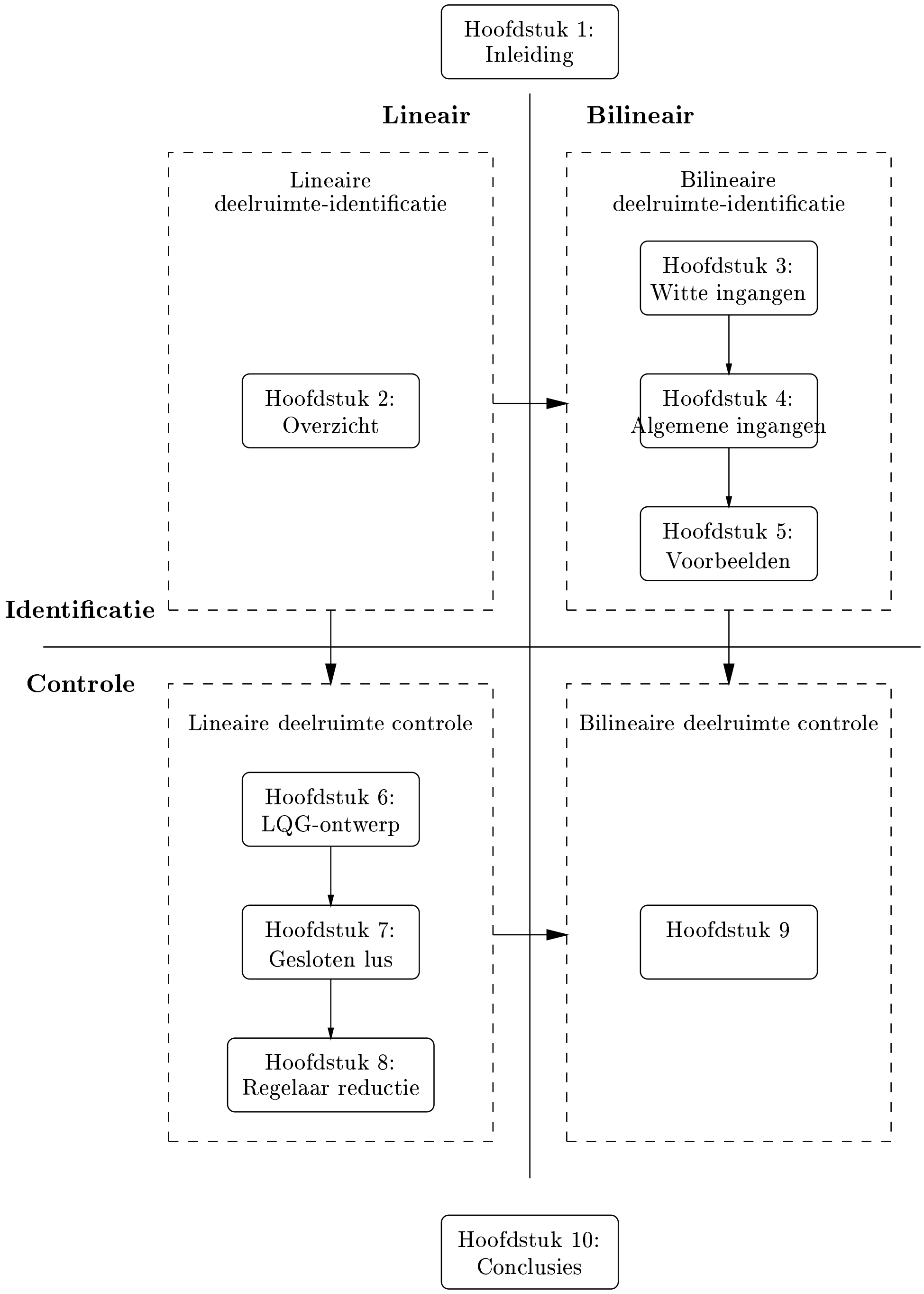
	Lineair	Bilineair
Deelruimte-identificatie	Hoofdstuk 2	Hoofdstuk 3, 4, 5
Deelruimte-gebaseerde regeling	Hoofdstuk 6, 7, 8	Hoofdstuk 9

Tabel 0.1: **Deze thesis is opgesplitst in twee delen: een identificatie deel en een controle deel. Elk van deze delen kan op zich ook opgesplitst worden in een lineair en een bilineair deel.**

alternatief zijn voor de meer klassieke predictiefout methoden, vooral voor de identificatie van hoge orde multivariabele lineaire systemen. Eén van de hoofddoelstellingen van deze thesis bestaat erin om de bestaande lineaire deelruimte-identificatietechnieken uit te breiden naar een ruimere klasse van systemen, namelijk bilineaire systemen. In Hoofdstuk 2 wordt eerst een korte herhaling gegeven van de theorie rond lineaire deelruimte-identificatie. In de Hoofdstukken 3 en 4 vervolgens wordt aangetoond dat, onder bepaalde voorwaarden, een gelijkaardige theorie kan ontwikkeld worden voor bilineaire systemen. In Hoofdstuk 5 tenslotte wordt aan de hand van verscheidene vergelijkende simulatievoorbeelden geïllustreerd dat bilineaire deelruimte-identificatie een duidelijk meerwaarde kan brengen ten op zichte van lineaire deelruimte-identificatie.

In het tweede deel van deze thesis concentreren we ons meer op de *controle* van lineaire en bilineaire systemen. Er wordt aangetoond dat er zeer sterke verbanden bestaan tussen deelruimte-identificatie enerzijds en modelgebaseerde predictieve controle (MPC) anderzijds. Gebaseerd op standaard technieken uit het gebied van de deelruimte-identificatie wordt een nieuwe klasse van algoritmen ontwikkeld om een optimale regelaar te berekenen, vertrekkend van metingen van de ingangen en uitgangen van een onbekend systeem en zonder expliciete systeemidentificatie. Dit gebeurt op een gelijkaardige manier als waarop een model van een onbekend systeem wordt bepaald met de ingangs- en uitgangsdata. In plaats van het vinden van een model van het onbekende systeem wordt nu direct een regelaar voor dit systeem gevonden. Eerst worden deze algoritmen ontwikkeld voor lineaire systemen (Hoofdstukken 6, 7, 8), waarna de uitbreiding naar bilineaire systemen wordt behandeld in Hoofdstuk 9.

Tabel 0.1 vat de opsplitsing van deze thesis in vier grote delen samen, namelijk identificatie en controle enerzijds, van lineaire en bilineaire systemen anderzijds. Figuur 0.1 stelt de structuur van de thesis voor en geeft het verband aan tussen de verschillende hoofdstukken.



Figuur 0.1: **Structuur van de thesis.**

Deel I: Deelruimte-identificatie

Hoofdstuk 2: Lineaire deelruimte-identificatie - Een overzicht

Wiskundige modellen van dynamische systemen worden gebruikt voor analyse, simulatie, predictie, foutdetectie, training, controle, etc... In de praktijk bestaan er verscheidene methoden om een model van een systeem te genereren. Men zou bijvoorbeeld kunnen vertrekken van de vergelijkingen die het fysische gedrag van het systeem beschrijven. Hoewel deze methode meestal tot vrij nauwkeurige modellen leidt heeft ze ook een aantal nadelen. Bijvoorbeeld een grote voorkennis van de fysica van het systeem moet beschikbaar zijn, wat niet altijd het geval is.

Een andere methode bestaat erin om eerst een parametrizatie van het systeem vast te leggen. Daarna worden de parameters van het model berekend zodat de metingen van de ingangen en uitgangen van het systeem zo goed mogelijk verklaard worden door het model. Deze techniek noemt men systeemidentificatie.

Het ontstaan van systeemidentificatie gaat terug tot het begin van de jaren 70. Gedurende de laatste 30 jaar is systeemidentificatie uitgegroeid tot een rijp en volwaardig domein van de systeemtheorie. Zeker de ontwikkeling van de zogenaamde predictiefout methoden heeft hiertoe bijgedragen [92]. Tot het begin van de jaren 90 was dit zeker de belangrijkste klasse van systeemidentificatiemethoden. Toen ontstond een nieuwe type algoritmen namelijk deelruimte-identificatie. Daar waar predictiefout methoden vertrekken van een transfertfunctie parametrizatie van het model, beschouwen deelruimte-identificatie methoden toestandsmodellen van de vorm:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k, && (0.1)\\ y_k &= Cx_k + Du_k + v_k. && (0.2) \end{aligned}$$

De ingang $u_k \in \mathbb{R}^m$, de uitgang $y_k \in \mathbb{R}^l$, de toestand $x_k \in \mathbb{R}^n$. De procesruis $w_k \in \mathbb{R}^n$ en de meetruis $v_k \in \mathbb{R}^l$ zijn zero-gemiddelde, witte, Gaussiaans verdeelde, stochastische variabelen met covariantie matrices:

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \quad \geq 0. \qquad (0.3)$$

Het probleem dat opgelost wordt door deelruimte-identificatie kan als volgt omschreven worden:

> **Gegeven** zijnde een groot aantal metingen van de ingangen u_k en de uitgangen y_k van het onbekende systeem (0.1)-(0.3), **bepaal** de orde n van het systeem en de toestandsmatrices A, B, C, D en de ruiscovariantie matrices Q, R, S.

Deelruimte-identificatie is hoofdzakelijk onstaan door het combineren van realizatietheorie en numeriek robuuste technieken uit de lineaire algebra zoals singuliere waarden ontbinding en LQ-ontbinding. Eén van de belangrijke nieuwe concepten van deelruimte-identificatie is het aantonen van het belang van de *toestand* binnen het kader van de systeemidentificatie. In tegenstelling tot klassieke identificatiemethoden wordt bij deelruimte-identificatie eerst de toestand van het systeem berekend, en pas later de toestandsmatrices van het systeem. Dit belangrijke onderscheid tussen deelruimte-identificatie en predictiefout methoden wordt geïllustreerd in Figuur 0.2

Een deelruimte-identificatiealgoritme bestaat meestal uit drie stappen:

Stap 1: LQ-ontbinding Tijdens de eerste stap worden uit de beschikbare ingangs/uitgangsdata bepaalde gestructureerde datamatrices Y_f, W_p en U_f gevormd. De rijruimtes van deze matrices worden dan op een welbepaalde manier op elkaar geprojecteerd:

$$Y_f /_{U_f} W_p \quad = \quad L_w W_p .$$

Deze projectie kan op een numeriek uiterst efficiënte manier berekend worden d.m.v. een LQ-ontbinding.

Stap 2: SW-ontbinding Het interessante van de hierboven gedefiniëerde projectie is dat, als het aantal kolommen in de datamatrices Y_f, W_p en U_f oneindig is, ze rangdeficiënt is. Bovendien is ze gelijk aan het product van de observeerbaarheidsmatrix Γ_i en een Kalman filter schatting $\widehat{X}_f$ van de toestandssequentie X_f van het systeem:

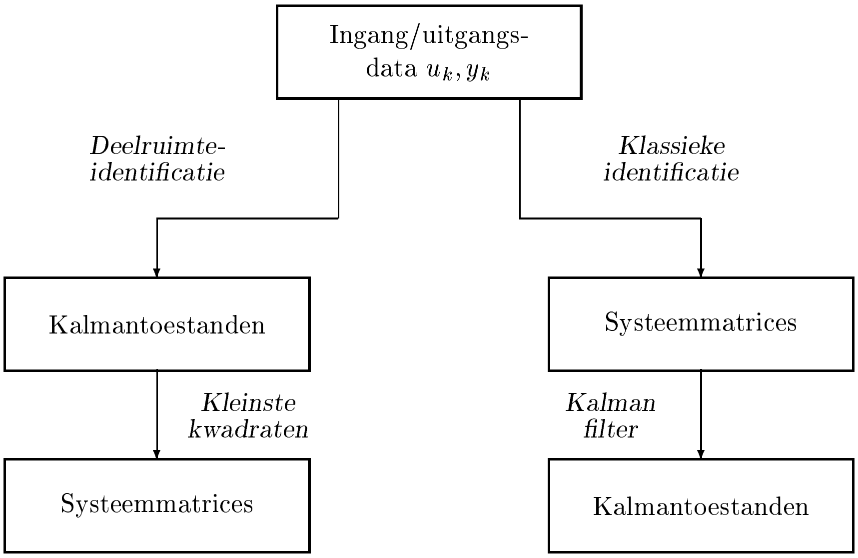
$$Y_f /_{U_f} W_p \quad = \quad \Gamma_i \widehat{X}_f .$$

Daaruit kunnen we besluiten dat a.h.v. de singuliere waarden (SW) ontbinding van L_w:

$$\begin{aligned} W_1 L_w W_2 &\stackrel{\text{def}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \\ &\approx U_1 S_1 V_1^T \end{aligned}$$

we onmiddellijk de observeerbaarheidsmatrix Γ_i en de Kalman filter schatting $\widehat{X}_f$ van de toestandssequentie X_f van het systeem kunnen berekenen:

$$\Gamma_i \quad = \quad W_1^{-1} U_1 S_1^{1/2} ,$$



Figuur 0.2: **Deze figuur vergelijkt de werking van een deelruimtemethode en die van een klassieke identificatiemethode. Bij de klassieke methode worden eerst de systeemmatrices berekend waarna, indien noodzakelijk, de Kalman filter toestand kan geschat worden. Bij deelruimte-identificatie echter worden de Kalman filter toestanden direct uit de ingangs/uitgangsdata berekend, waarna het uiterst eenvoudig is om de systeemmatrices A, B, C en D te berekenen.**

$$\widehat{X}_f = S_1^{1/2} V_1^T W_2^{-1} W_p .$$

Stap 3: A, B, C, D, Q, R, S De laatste stap van deelruimte-identificatie algoritmen bestaat erin om, vertrekkend van de geschatte observeerbaarheidsmatrix Γ_i en de Kalman filter toestandssequentie $\widehat{X}_f$, de systeemmatrices A, B, C, D en de ruiscovariantiematrices Q, R, S te berekenen. Dit is het punt waar de meeste deelruimte-identificatiealgoritmen verschillen. Sommigen gebruiken de toestandsschatting $\widehat{X}_f$ om de systeemmatrices te bepalen (N4SID [123], CVA [88]) terwijl anderen vooral gebruik maken van de observeerbaarheidsmatrix Γ_i (MOESP [127]).

Hoofdstuk 3: Deelruimte-identificatie van bilineaire systemen met witte ingangen

In heel wat praktische situaties worden processen geregeld op basis van lineaire modellen. In vele gevallen echter, en vooral in de chemische industrie, vertonen deze processen een duidelijk nietlineair gedrag, en het werkingspunt kan sterk veranderen in de tijd. Een typisch voorbeeld hiervan is de opstartprocedure van een distillatiekolom. Het gevolg is dat regelaars gebaseerd op lineaire modellen voor dergelijke processen een slecht regelgedrag kunnen vertonen, of zelfs tot instabiliteit leiden.

Een oplossing hiervoor bestaat erin om nietlineaire modellen, bvb. witte doos modellen, te gebruiken. Deze zijn immers geldig in een breder domein rond een werkingspunt dan lineaire modellen. Het probleem hierbij is dat, vanuit een wiskundig standpunt, dergelijke modellen veel ingewikkelder zijn dan lineaire modellen, met als gevolg dat de resulterende regelaars ook een hogere complexiteit zullen hebben. Een goed compromis tussen de eenvoud van lineaire modellen enerzijds en de precisie van nietlineaire modellen anderzijds, zijn bilineaire modellen. Bilineaire modellen lijken qua vorm zeer sterk op lineaire modellen en worden beschreven door de volgende toestandsvergelijkingen:

$$x_{k+1} = Ax_k + Nu_k \otimes x_k + Bu_k + w_k, \tag{0.4}$$
$$y_k = Cx_k + Du_k + v_k, \tag{0.5}$$

waar de ingang $u_k \in \mathbb{R}^m$, de uitgang $y_k \in \mathbb{R}^l$, de toestand $x_k \in \mathbb{R}^n$. De procesruis $w_k \in \mathbb{R}^n$ en de meetruis $v_k \in \mathbb{R}^l$ zijn zero-gemiddelde, witte, Gaussiaans verdeelde, stochastische variabelen met covariantie matrices:

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \quad \geq 0. \tag{0.6}$$

De matrix N is gedefiniëerd als:

$$N \stackrel{\text{def}}{=} \begin{pmatrix} N_1 & \dots & N_m \end{pmatrix}$$

en $\otimes$ het Kronecker product.

In dit hoofdstuk maken we ook de belangrijke veronderstelling dat de ingang u_k van het systeem wit is en onafhankelijk van de meet- en procesruis:

$$\mathbf{E}[u_p \begin{pmatrix} u_q \\ w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} I_m & 0_{m\times n} & 0_{m\times l} \end{pmatrix}. \tag{0.7}$$

In dit hoofdstuk worden de in Hoofdstuk 2 herhaalde lineaire deelruimte-identificatietechnieken uitgebreid tot analoge algoritmen voor bilineaire systemen. Dit probleem kan als volgt geformuleerd worden:

> **Gegeven** zijnde een groot aantal metingen van de ingangen u_k en de uitgangen y_k van het onbekende systeem (0.4)-(0.6), en onder de veronderstelling dat de ingangen u_k wit en onafhankelijk zijn van de meet- en procesruis, **bepaal** de orde n van het systeem en de toestandsmatrices A, B, C, D, N en de ruiscovariantie matrices Q, R, S.

Net als bij lineaire systemen speelt realizatietheorie ook een belangrijke rol bij bilineaire deelruimte-identificatie. Eerst wordt getoond hoe een bilineair systeem van de vorm (0.4)-(0.6) kan opgesplitst worden in de som van een stochastisch en een deterministisch deelsysteem en wordt het deterministisch en stochastisch realizatieprobleem van bilineaire systemen kort herhaald. Vervolgens worden de Kalman filter vergelijkingen van een bilineair systeem (0.4)-(0.6) afgeleid. Dit is nodig omdat, net als bij lineaire deelruimte-identificatie, de toestanden geschat met het bilineair deelruimte-algoritme specifieke Kalman filter toestanden zijn van het te identificeren systeem. Uiteindelijk worden de veralgemeende Hankel matrices, verkregen uit de beschikbare ingangs/uitgangsdata van het systeem, gepresenteerd. Deze matrices zijn een veralgemening van de Hankel matrices die gebruikt worden in lineaire deelruimte-identificatie in die zin dat ze, naast de rijen van de lineaire Hankel matrices, ook nog rijen bevatten die bestaan uit welbepaalde producten tussen ingangen en uitgangen. Zodoende hebben ze een veel rijkere structuur en wordt de *bilineaire* informatie, d.w.z. die informatie aanwezig in de data gegenereerd door de bilineaire term $u_k \otimes x_k$, direct meegenomen in de data matrices.

Het resulterende deelruimte-identificatiealgoritme is zeer gelijkaardig aan dat van lineaire deelruimte-identificatie. Men kan ook duidelijk drie stappen onderscheiden, namelijk het bepalen van bepaalde veralgemeende Hankel matrices uit de beschikbare ingangs/uitgangsdata waarop de volgende bewerkingen worden uitgevoerd:

- LQ-ontbinding of projectie
- SW-ontbinding
- Bepaling van A, B, C, D, N, Q, R, S.

Er werd aangetoond dat het algoritme consistente schattingen geeft van het deterministisch deelsysteem. Op de schatting van het ruismodel, d.w.z. het stochastisch deelsysteem, bevindt zich een lichte systematisch fout die kleiner wordt en asymptotisch verdwijnt naarmate het aantal kolommen in de veralgemeende Hankel matrices groter wordt.

Een verdere eigenschap van dit algoritme is dat het een directe veralgemening is van de bestaande lineaire deelruimte-identificatie algoritmen. Concreet betekent dit dat als het algoritme toegepast wordt op data afkomstig van een lineair systeem, dan vinden we ook een consistente schatting van dit lineair systeem terug.

Hoofstuk 4: Deelruimte-identificatie van bilineaire systemen met algemene ingangen

In Hoofdstuk 3 werd verondersteld dat de ingangen u_k van het systeem wit zijn en onafhankelijk van de meet- en systeemruis. Hoewel dit een vrij plausibele veronderstelling is zijn er toch heel wat praktische situaties waar, om veiligheids- of economische redenen, witte ruis een onaanvaardbaar ingangssignaal is. De energie- en productverliezen die zouden gepaard gaan tijdens een dergelijk experiment, in het geval van een industriële distillatiekolom bijvoorbeeld, zijn gigantisch. Een andere situatie waar witte ingangssignalen niet realiseerbaar zijn is het geval waar de data opgemeten wordt in gesloten lus omdat het systeem bijvoorbeeld onstabiel is in open lus.

In dit hoofdstuk wordt het deelruimte-identificatie algoritme van Hoofdstuk 3 veralgemeend tot een gelijkaardig algoritme waar de veronderstelling dat de ingang wit moet zijn niet meer noodzakelijk is. Evenals de vorige algoritmen vinden we ook hier de typische drie stappen terug, namelijk LQ-ontbinding, SW-ontbinding en extractie van de toestandsmatrices. Hoewel dit algoritme enkel consistente schattingen van zowel het deterministisch als het stochastisch deelsysteem geeft indien het aantal rijen in de veralgemeende data Hankelmatrices oneindig is, is ook dit algoritme een directe veralgemening van zowel het de lineaire deelruimte-algoritmen (Hoofdstuk 2) als het bilineaire deelruimte-algoritme waar de ingangen verondersteld werden wit te zijn (Hoofdstuk 3).

Eén van de, blijkbaar onvermijdelijke, problemen binnen het kader van bilineaire systeemtheorie is de vloek van de dimensionaliteit. Bijvoorbeeld, daar waar bij lineaire systemen het aantal Markov parameters die nodig is om de uitgang te beschrijven als een functie van de i laatste ingangen lineair stijgt als functie van i, stelt men vast dat bij bilineaire systemen het aantal Volterra kernen een exponentieel stijgende functie is van i. Deze eigenschap heeft uiteraard ook zijn invloed op de bilineaire deelruimte-identificatiealgoritmen. Zo zal, daar waar bij lineaire deelruimte-identificatie het aantal rijen van de data Hankel matrices lineair stijgt, het aantal rijen bij bilineaire deelruimte-identificatie exponentieel stijgend. In Hoofdstuk 4 worden enkele praktische tips gegeven om deze zogenaamde vloek van de dimensionaliteit tegen te gaan. Zo worden onder andere enkele methoden voorgesteld om de veralgemeende Hankel matrices op een zinvolle manier te reduceren.

Hoofstuk 5: Voorbeelden

In dit hoofdstuk worden verschillende voorbeelden gegeven waar lineaire en bilineaire deelruimte-identificatie met elkaar vergeleken worden. De vergelijking gebeurt door, op basis van één en dezelfde ingangs/uitgangsdataset, enerzijds een lineair systeem te identificeren en anderzijds een bilineair systeem. Hiervoor worden de bestaande lineaire deelruimte-identificatiealgoritmen en de bilineaire deelruimte-algoritmen van de Hoofdstukken 3 en 4 gebruikt.

Enerzijds worden eenvoudige academische voorbeelden behandeld en anderzijds een volledig nietlineair witte doos simulatie model van een twee-componenten distillatiekolom. Uit al deze voorbeelden blijkt dat bilineaire deelruimte-identificatie een significante verbetering kan betekenen ten opzichte lineaire deelruimte-identificatie.

Deel II: Deelruimte-gebaseerde regeling

Hoewel het probleem, dat in Deel II van deze thesis behandeld wordt, vrij gelijkaardig is aan dat van Deel I, ligt de nadruk meer op regeling i.p.v. op identificatie. Terwijl het in systeemidentificatie de bedoeling is om een model van een onbekend systeem te vinden, proberen we hier direct een optimale regelaar te vinden voor dit systeem, zonder eerst een model van het systeem zelf te zoeken.

Hoofstuk 6: Modelloos deelruimte-gebaseerd LQG-ontwerp

Een veel voorkomende reden om een model te maken van een systeem is om dit model te gebruiken voor regeldoeleinden. Inderdaad, de meeste "moderne"regelstrategieën gebruiken een model van het te regelen systeem. De filosofie hierachter bestaat erin om de regelingang zodanig te berekenen dat het verschil tussen de gewenste uitgang en de werkelijke uitgang van het systeem zo klein mogelijk is. Deze ingang wordt meestal berekend door het minimaliseren van een bepaald regelcriterium. Optimale regeling (LQG) en modelgebaseerd predictieve regeling (MPC) zijn hier voorbeelden van.

Een meer formele definitie van het probleem dat in dit hoofdstuk behandeld wordt is gegeven in Figuur 0.3.

Modelloos LQG-regelprobleem

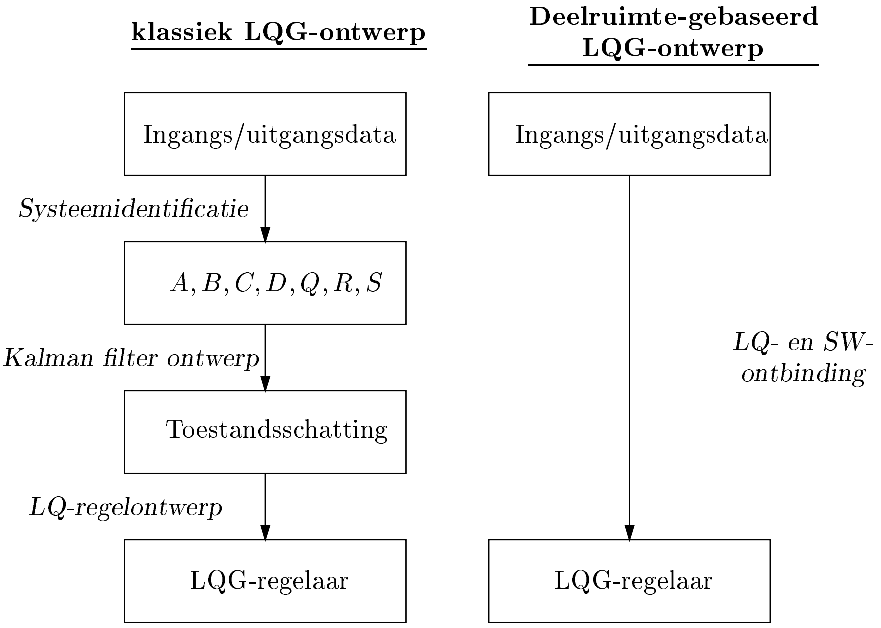
Gegeven *zijnde een groot aantal open lus metingen van de ingangen* u_k *en de uitgangen* y_k*,* $k < 0$ *van een onbekend lineair tijdsinvariant systeem (0.1)-(0.3),* **vind** *de ingangssequentie* $\widehat{u}_f = (\widehat{u}_1, \ldots, \widehat{u}_N)$ *die de volgende kwadratisch kostfunctie* J *minimaliseert over de horizon* N*:*

$$J = \sum_{k=1}^{N} \left\{ (\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k \right\} .$$

Figuur 0.3: **Modelloos LQG-regelprobleem.**

Vertrekkend van ingangs- en uitgangsmetingen van een onbekend systeem, bestaat de klassieke weg om dit probleem op te lossen meestal uit drie verschillende stappen. Eerst moet een model van het systeem berekend worden, wat typisch gebeurt door middel van een systeemidentificatie methode. In een tweede stap wordt dit model gebruikt om een schatting van de toestand van het systeem te berekenen. Dit gebeurt dikwijls met een zogenaamd Kalman filter. Uiteindelijk worden zowel de toestandsschatting en het model gebruikt om de optimale regelaar te ontwerpen. Hoewel voor het oplossen van deze drie problemen afzonderlijk tal van onderzoek werd gedaan en vrij elegante oplossingen bestaan, werden deze drie problemen tot nu toe nooit als één groot probleem beschouwd. De meeste resultaten beschikbaar in de literatuur beschouwen bijvoorbeeld enkel het systeemidentificatieprobleem, of het regelaarontwerp gegeven zijnde het systeemmodel.

In dit hoofdstuk wordt, gebaseerd op technieken gebruikt in de deelruimte-identificatie, een nieuw algoritme ontwikkeld dat toelaat om, vertrekkend van ingangs- en uitgangsmetingen van een onbekend lineair systeem, op een uiterst directe manier een optimale regelaar voor dit systeem te vinden. De methode is zeer sterk verwant met de lineaire deelruimte-identificatiealgoritmen van Hoofdstuk 2 en is dus ook enkel gebaseerd op numeriek robuste lineaire algebra technieken zoals LQ- en SW-ontbinding. In Figuur 0.4 geven we een



Figuur 0.4: **De hoofdidee van Deel II van de thesis. Gebaseerd op deelruimte-identificatie wordt een nieuw algoritme ontwikkeld om het modelloos LQG-regelprobleem van Figuur 0.3 op te lossen. De klassieke manier om dit te doen bestaat typisch uit drie stappen. Eerst wordt een systeemidentificatie gedaan om de toestandsmatrices** A, B, C, D, Q, R, S **te vinden. Een tweede stap worden deze matrices gebruikt in een Kalman filter ontwerp om een schatting van de toestand van het systeem te vinden. Uiteindelijk wordt deze toestandsschatting gebruikt in een LQ-regelaar. De hoofdidee van Deel II is dat, gebaseerd op technieken gebruikt in deelruimte-identificatie, een nieuwe methode wordt ontwikkeld waardoor de drie stappen in het klassieke LQG-regelprobleem kunnen vervangen worden door een LQ- en een SW-ontbinding van matrices die enkel metingen van de ingangen en de uitgangen van het onbekende systeem bevatten.**

directe vergelijking tussen de klassieke manier en de deelruimte-gebaseerde manier om het modelloos LQG-regelprobleem (Figuur 0.3) op te lossen.

Ook wordt aangetoond dat, indien het aantal rijen van de matrices in het deelruimte-gebaseerde LQG-algoritme oneindig is, de klassieke en de deelruimte-gebaseerde LQG-regelaars equivalent zijn. Dit wordt ook aan de hand van een simulatievoorbeeld gestaafd.

Hoofdstuk 7: Deelruimte-gebaseerd LQG-ontwerp in gesloten lus

In het deelruimte-gebaseerde LQG-algoritme van het vorige hoofdstuk werd verondersteld dat de beschikbare metingen van de ingangen en de uitgangen van het systeem opgemeten werden in open lus. Indien het algoritme zou toegepast worden op metingen opgenomen in gesloten lus zou er een systematische fout aanwezig zijn op de zo berekende LQG-regelaar. In dit hoofdstuk wordt de oorsprong van dit probleem geanalyseerd en een uitbreiding van het algoritme uit Hoofdstuk 6 voorgesteld zodat ook metingen uit een gesloten lus kunnen gebruikt worden.

We veronderstellen dat de metingen opgenomen werden onder lineaire tijdsinvariante terugkoppeling, waar de regelaar beschreven wordt door de volgende toestandsvergelijkingen:

$$x^c_{k+1} = A_c x^c_k + B^y_c y_k + B^r_c r_k, \tag{0.8}$$

$$u_k = C_c x^c_k + D^y_c y_k + D^r_c r_k \tag{0.9}$$

met $r_k \in \mathbb{R}^l$ de referentieuitgang, $x^c_k \in \mathbb{R}^{n_c}$ de toestand van de regelaar en n_c de orde. Verder wordt aangenomen dat het probleem niet singulier is in die zin dat $I_l + DD^y_c$ van volle rang is.

Het probleem dat we in dit hoofdstuk behandelen wordt formeel beschreven in Figuur 0.5.

De reden waarom het open lus deelruimte-gebaseerd LQG-algoritme van Hoofdstuk 6 foute schattingen geeft van de regelaar, indien metingen uit een gesloten worden gebruikt, is te wijten aan de ruis op het systeem. In het open lus deelruimte-gebaseerd algoritme wordt de fundamentele veronderstelling gemaakt dat de ingangen van het systeem onafhankelijk zijn van de systeem- en meetruis. Wanneer de ingangen en uitgangen gemeten zijn in open lus is dit inderdaad een juiste veronderstelling. Door de terugkoppeling echter komt deze veronderstelling in het gedrang. De uitgang, en dus ook de meet -en systeemruis, worden namelijk door de feedback terug naar de ingang gevoerd. Gebaseerd op resultaten van de deelruimte-identificatieliteratuur wordt in dit hoofdstuk een oplossing geboden voor dit probleem. De idee bestaat erin om de ingang, die gecorreleerd is met de ruis, te vervangen door een nieuwe variabele die niet gecorreleerd is met de ruis. Deze variabele kan berekend worden op basis van de kennis van de N eerste impulsresponsiecoefficiënten van de voorlopige regelaar en de beschikbare gesloten lus ingangs- en uitgangsdata.

Het resulterende gesloten lus modelloos deelruimte-gebaseerd LQG-algoritme is

Gesloten lus modelloos LQG-regelprobleem

Gegeven *zijnde een groot aantal metingen van de ingangen* u_k *en de uitgangen* y_k, $k < 0$ *van een onbekend lineair tijdsinvariant systeem (0.1)-(0.3) en de* N *eerste impulsresponsie coefficiënten van de voorlopige regelaar (0.8)-(0.9),* **vind** *de ingangssequentie* $\widehat{u}_f = (\widehat{u}_1, \dots, \widehat{u}_N)$ *die de volgende kwadratisch kostfunctie* J *minimaliseert over de horizon* N*:*

$$J = \sum_{k=1}^{N} \left\{ (\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k \right\}.$$

Figuur 0.5: **Gesloten lus modelloos LQG-regelprobleem.**

zeer gelijkaardig aan het open lus algoritme van Hoofdstuk 6. Aan de hand van een simulatievoorbeeld wordt aangetoond dat het open lus algoritme inderdaad een foute regelaar geeft indien gesloten lus data wordt gebruikt. Het gesloten lus algoritme van dit hoofdstuk echter geeft een consistente regelaar.

Hoofdstuk 8: Deelruimte-gebaseerde regelaar reductie

Wanneer we de modelloze deelruimte-gebaseerde LQG-algoritmen uit de Hoofdstukken 6 en 7 wat verder bestuderen, dan stellen we vast dat de orde van de resulterende regelaars gelijk is aan het aantal blokrijen M in de data Hankel matrices. Zoals werd aangetoond in Hoofdstuk 6 zal de deelruimte-gebaseerde LQG-regelaar enkel gelijk zijn aan de klassieke LQG-regelaar indien het aantal blokrijen M oneindig groot is. Dit heeft als gevolg dat de deelruimte-gebaseerde regelaars van een vrij hoge orde kunnen zijn. Uiteraard is het in de praktijk belangrijk om de orde van een regelaar zo laag mogelijk te houden.

In de literatuur bestaan er verscheidene methoden om een regelaar met een lage orde te berekenen. In de meeste gevallen kunnen ze opgesplitst worden in twee grote klassen:

Open lus: Hierbij wordt eerst het model van systeem gereduceerd. Met dit lage orde model wordt dan een regelaar berekend die ook van lage orde is. Deze werkwijze wordt over het algemeen als gevaarlijk beschouwd omdat tijdens de reductie van het systeemmodel op geen enkele manier rekening gehouden wordt met het feit dat het systeem in een terugkoppeling zal functioneren. Door het reduceren van het systeemmodel in open lus kunnen namelijk bepaalde dynamische effecten van het model verwijderd worden die in open lus slechts een kleine rol spelen, maar van primordiaal belang zijn in de gesloten lus. Dit fenomeen, dat kan leiden tot een slecht regelgedrag of zelfs tot instabiliteit, noemt men dikwijls *spill-over*. Door zijn relatieve eenvoud is open lus *gebalanceerde* modelreductie [98] een vrij populaire reductiemethode.

Gesloten lus: Een veiliger manier van werken bestaat erin om eerst een regelaar te ontwerpen op basis van een hoog orde systeemmodel. Daarna kan de orde van de resulterende hoge order regelaar gereduceerd worden naar een lagere orde. Het grote verschil met open lus reductie is dat tijdens deze reductie kan rekening gehouden worden met de terugkoppeling. Gesloten lus gebalanceerde modelreductie [66] is een vrij recente uitbreiding van de open lus versie [98].

De resultaten van dit hoofdstuk zijn tweeërlei:

- Eerst wordt uiteengezet hoe het deelruimte-gebaseerd LQG-algoritme van Hoofdstuk 6 aangepast kan worden zodanig dat, in plaats van een hoge orde regelaar te berekenen, direct een regelaar met een lage orde berekend wordt.
- Verder wordt de invloed van wegingsmatrices in de SW-ontbinding op de regelaarreductie bestudeerd. Er wordt aangetoond dat, naargelang de welbepaalde keuze van deze wegingsmatrices, de regelaar die berekend wordt met het algoritme zowel in open of gesloten gebalanceerde vorm kan zijn.

Hoofdstuk 9: Modelloze deelruimte-gebaseerde predictieve regeling van bilineaire systemen

In het voorlaatste hoofdstuk van deze thesis combineren we de resultaten van Hoofdstuk 3 en die van Hoofdstuk 6, namelijk de deelruimte-identificatie van bilineaire systemen enerzijds en de deelruimte-gebaseerde LQG-algoritmen (Hoofdstuk 6) anderzijds. Het resultaat is een nieuw deelruimte-gebaseerd algoritme voor het berekenen van predictieve regelaars voor MIMO bilineaire systemen van de vorm (0.4)-(0.6).

Hoofdstuk 10: Conclusies en open problemen

Deel I: Deelruimte-identificatie

Gedurende de laatste 10 jaar is deelruimte-identificatie uitgegroeid tot een rijp en veelbelovend onderzoeksdomein. Het succes van lineaire deelruimte-identificatie kan o.a. toegeschreven worden aan de succesvolle toepassingen op moeilijke industriële processen. In het eerste deel van deze thesis hebben we de bestaande lineaire deelruimte-algoritmen uitgebreid tot een nieuwe klasse van algoritmen voor de identificatie van bilineaire systemen. Deze algoritmen zijn een directe veralgemening van de bestaande lineaire deelruimte-methoden. Het nut van bilineaire systemen bestaat erin dat ze een goede balans zoeken tussen de precisie van algemene nietlineaire systemen en de wiskundige eenvoud van lineaire systemen.

Deel II: Deelruimte-gebaseerde predictieve regeling

In Deel II werd bestudeerd hoe de deelruimte-methoden van Deel I kunnen gebruikt worden voor regeldoeleinden. Het werd duidelijk dat er een verregaande analogie bestaat tussen deelruimte-identificatie enerzijds en optimale regeling anderzijds. Daar waar identificatie probeert om een model te genereren uit ingangs- en uitgangsdata, probeert regeling een optimale regelingang te vinden op basis van een model.

Deze dualiteit werd toegepast op deelruimte-identificatiealgoritmen. Dit leidde tot een nieuwe klasse algoritmen die, gebaseerd op technieken uit deelruimte-systeemidentificatie, toelaat om een regelaar van een onbekend systeem te vinden vertrekkend van metingen van de ingangen en de uitgangen van dit onbekende systeem. Het verband tussen deze algoritmen en lineaire optimale regeling werd aangetoond.

Chapter 1

Introduction

As the title suggests, this thesis can be divided into several parts. On the one hand it considers identification as well as control, on the other hand linear as well as bilinear systems are studied. At first this might seem a much too general subject to be considered in depth in one single thesis. Already in the field of linear system identification, numerous books have been written. Considering bilinear systems, i.e. a larger class than linear systems, and moreover developing new controller design methods for such systems might seem hopeless. However, there is a glue that connects these apparently quite different problems. The red wire throughout this thesis is that *subspace techniques* are used to design a new class of algorithms for the identification and control of linear as well as bilinear systems. By subspace techniques we mean the class of methods that use projections and/or singular value decompositions of matrices.

This thesis is divided in two main parts. The first part deals with subspace identification of bilinear systems. The last decade, subspace identification methods have proven to be an excellent alternative to the more classical prediction error methods, for the identification of high order, multivariable, linear systems. It is our purpose to extend linear subspace identification to a wider class of systems, namely bilinear systems. In Chapter 2 we first give a brief overview of the main results of linear subspace system identification. In Chapters 3 and 4 it is shown that, under certain conditions, a similar theory can be derived for bilinear systems. In Chapter 5, finally, several examples are given where bilinear subspace identification is applied and compared with linear subspace identification. The second part of this thesis focuses on the control of linear and bilinear systems. It became clear that there exist strong analogies between subspace system identification and optimal model-based predictive control methods. Using the standard subspace identification techniques, a new class of algorithms is derived to calculate an optimal controller directly from measured input and

	Linear	Bilinear
Subspace identification	Chapter 2	Chapter 3, 4, 5
Subspace-based control	Chapter 6, 7, 8	Chapter 9

Table 1.1: **The thesis is divided in two main parts: an identification part and a control part. In turn, both of them can be subdivided into a linear and a bilinear part.**

output data of an unknown system. This is done in a similar way as a model is calculated from data in the subspace identification problem, but instead of finding a model of the unknown plant, the controller is found. First the case of linear controllers is treated (Chapters 6, 7, 8) and further the extension to predictive controllers for bilinear systems is made in Chapter 9.

Table 1.1 summarizes the four main pillars of this thesis, namely linear and bilinear subspace identification on the one hand, and linear and bilinear subspace-based predictive controller design on the other.

Although the thesis is divided into an identification and a control part, there are clearly three main themes that are discussed:

Theme I: Subspace system identification

Theme II: Subspace-based controller design

Theme III: Bilinear systems.

In what follows we give a short historical overview of these three topics. After that we present a detailed chapter by chapter overview where the main contributions of the thesis are pointed out, together with some references to the literature where these results have appeared.

1.1 Theme I: Subspace system identification

In this section we give a brief historical overview of subspace system identification. This overview was partially borrowed from various papers [29], [31], [32].

Mathematical models of dynamical systems are used for analysis, simulation, prediction, optimization, monitoring, fault detection, training and control. There are several approaches to generate a model of a system. One could for instance start from first principles, such as writing down the basic physical or chemical laws that generate the behavior of the system. This so-called "white-box" approach works for simple examples, but its complexity increases rapidly for real-world systems. In some cases, the system's equations are known up to within some unknown parameters, which are estimated using some parameter estimation method ("grey-box" modeling).

Another approach is provided by system identification, in which first measurements or observations are collected from the system, which are then "modeled" using a so-called "black-box" identification approach. Such an approach basically consists of first defining a parameterization of the model, and then determining the model parameters in such a way that the measurements are "explained" as accurately as possible by the model. Typically, this is done by formulating the identification problem as an optimization problem, in which the variables are the unknown parameters of the model, the constraints are the model equations and the objective function a measure of the deviation between the observations and the predictions or simulations obtained from the model.

Models and or systems can be roughly divided into classes, such as linear and nonlinear, time-invariant or time-varying, discrete-time or continuous-time, with lumped or with distributed parameters, etc... While at first sight, the class of linear time-invariant models with lumped parameters seems to be rather restricted, it turns out in practice that many real-life input/output behaviors of practical, industrial processes can be approximated very well by such a model.

The field of (linear) system identification is certainly not new, although we can safely say that it only started to blossom in the 1970's. Yet, 30 years of research have generated a lot of results and practical hands-on experience[1]. If we would be forced to mention some key references for the field, they would certainly include [8](state-of-the-art survey of 1971), [18](a classic in time-series analysis), [37], [111]. The field certainly matured with the advent of so-called *prediction error methods* (PEM) by Ljung and co-workers [92].

The beginning of the 1990's witnessed the birth of a new type of linear system identification algorithms, called *subspace methods*. Linear subspace identification methods consider systems and models of the form[2]:

$$x_{k+1} = Ax_k + Bu_k + w_k \, , \tag{1.1}$$

$$y_k = Cx_k + Du_k + v_k \, , \tag{1.2}$$

with

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \quad \geq 0. \tag{1.3}$$

[1]The number of system identification book titles that can be found e.g. at http://www.esat.kuleuven.ac.be/sista/daisy runs in the hundreds, not to mention the literally thousands of papers that have appeared in conference proceedings and journals.

[2]$\mathbf{E}$ denotes the expected value operator and δ_{pq} the Kronecker delta.

The vectors $u_k \in \mathbb{R}^{m\times 1}$ and $y_k \in \mathbb{R}^{l\times 1}$ are the measurements at time instant k of respectively the m inputs and l outputs of the process. The vector x_k is the state vector of the process at time instant k, $v_k \in \mathbb{R}^{l\times 1}$ and $w_k \in \mathbb{R}^{n\times 1}$ are unobserved vector signals, v_k is called the measurement noise and w_k is called the process noise. It is assumed that they are zero mean, stationary white noise vector sequences and uncorrelated with the inputs u_k. $A \in \mathbb{R}^{n\times n}$ is the system matrix, $B \in \mathbb{R}^{n\times m}$ is the input matrix, $C \in \mathbb{R}^{l\times n}$ is the output matrix while $D \in \mathbb{R}^{l\times m}$ is the direct feed-through matrix. The matrices $Q \in \mathbb{R}^{n\times n}$, $S \in \mathbb{R}^{n\times l}$ and $R \in \mathbb{R}^{l\times l}$ are the covariance matrices of the noise sequences w_k and v_k. In subspace identification it is typically assumed that the number of available data points goes to infinity, and that the data is ergodic.

The main problem treated in subspace identification is:

> **Given** a large number of measurements of the inputs u_k and the outputs y_k generated by the unknown system (1.1)-(1.3), **determine** the order n of the unknown system, the system matrices A, B, C, D up to within a similarity transformation and the noise covariance matrices Q, R, S.

Subspace methods basically originate in a happy *marriage-à-trois* between system theory, geometry and numerical linear algebra. Some historical basis references in which the subspace system identification methodology was developed are [70], [136], [85], [98] (realization theory), [5], [38], [118] (stochastic realization), [33], [28], [96] (deterministic identification), [88], [89], [127], [119], [126], [123] (combined deterministic/stochastic identification). Previous survey papers and books emphasizing different aspects of subspace system identification and signal processing, and in which one can find large sets of references to the literature are [29], [117], [130], [31], [123]. We should also mention some special issues of the journals Automatica ("Special Issue on Statistical Signal Processing and Control", Jan. 94; "Special Issue on System Identification", Dec. 95;) and Signal Processing ("Special Issue on Subspace Methods for System Identification", Jul. 96), which contained contributions on subspace identification, as well as the Proceedings of the 11th IFAC Symposium on System Identification (Kitakyushu, Japan, Jul. 97) and the book [123].

One of the important ideas at the origin of the development of subspace algorithms was the re-introduction of the concept of *the state x_k of a dynamical system* within the system identification context. In contrast to "classical" identification algorithms, a lot of the subspace algorithms first estimate/calculate the states (implicitly or explicitly), while next the (state-space) model is determined. This main difference between PEM methods and subspace methods is illustrated in Figure 1.1.

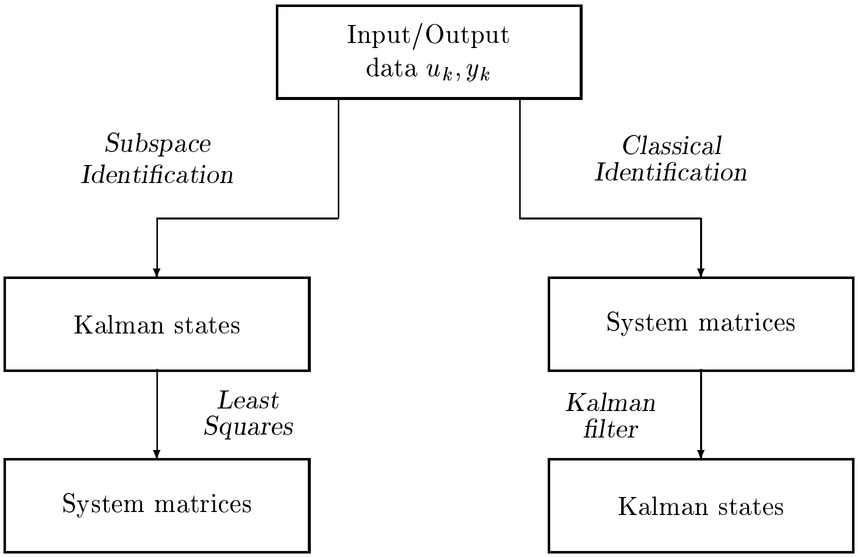


Figure 1.1: **System identification aims at constructing models from input/output data. The left hand side shows a subspace method approach: the (Kalman filter) states can be estimated directly from input/output data, after which it is easy to obtain the system matrices A, B, C and D. The right hand side is the classical approach: first the system matrices are obtained after which the states can be estimated.**

Why would one bother to first obtain the states, directly from input-output data, and only after that the state-space model? The answer is twofold:

- When the states x_k are available, it can be seen from the state-space model (1.1)-(1.3) that the matrices A, B, C and D can be obtained by solving a least squares problem, as these elements appear linearly as unknowns in the state-space equations. Further, the covariance matrices Q, R and S can then be found from the least squares residuals (see Section 2.4.3). In doing so, the identification problem is *linearized* in the sense that it is reduced to a simple least squares problem, contrary to PEM where nonlinear optimization problems have to be tackled!

- Secondly, in most of the modern control methods the knowledge of the state is needed to feed it back to the control input. As will become clear in Chapter 6, the fact that the states are found directly from the data, i.e. even before the plant model is calculated, can be heavily exploited for control purposes. It turns out that subspace methods are in fact very well suited for optimal control.

1.2 Theme II: Subspace-based controller design

A typical reason in practise to make a model of a system is to use it for control purposes. Indeed, most of the "modern" control strategies use a model of the system to be controlled. The philosophy of these methods is to calculate the control input as a function of the difference between the desired output of the process and the output predicted with the model. This input is often calculated as the solution of some quadratic control criterion. Optimal control (LQG) and Model Predictive Control (MPC) are examples of such control methods.

Starting from input/output data of an unknown system, the classical way a model-based controller is designed consists of several steps. First a system identification is performed to obtain a model. This model is then used in a model-based control algorithm to design the controller. It is beyond doubt that the historical evolution of model-based control and system identification are closely related, simply because one strongly depends on the other. These two problems however are hardly ever considered together. Most authors either concentrate on the problem of finding a model from the data, i.e. the system identification problem, or on finding a controller given the plant model, i.e. the control design. In the second part of this thesis we will try to solve this quite general and important problem which can be formulated as in Figure 1.2.

There are of course a lot of topics in the literature that are, in one way or another, related to this problem. On the one hand there is the field of optimal control and system identification, while on the other hand there is also the adaptive control literature which searches to combine system identification and control design. Let us now focus some more on these topics:

System identification: For an overview of the most important references in the field of system identification we refer to Section 1.1.

Optimal control: The origin of optimal control can be traced back to the work of Kalman [78] which was later resumed very nicely in the books of Anderson and Moore [7] and Kwakernaak and Sivan [87]. Kučera [86] studied the optimal control problem using a transfer function description of the system instead of the classical state-space framework that was used up till then. Another class of methods that is strongly related to

Model-free LQG-control problem

Given *an infinite number of measurements of the inputs* u_k *and the outputs* y_k *of an unknown system (1.1)-(1.3),* **find** *the linear time-invariant output feedback controller that minimizes the following cost function:*

$$J = \sum_{k=1}^{N} \{(\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k\} \qquad (1.4)$$

where $\widehat{y}_k$ *is the optimal k-step ahead predictor,* r_k *is the reference output,* $\widehat{u}_k$ *the corresponding control inputs and* Q_k, R_k *are user-dependent weighting matrices.*

Figure 1.2: **Model-free LQG-control problem.**

the work of this thesis, and that should certainly be kept in mind, are the so-called Model Predictive Control (MPC) methods. Whereas LQG was mainly developed in academic circles, MPC arose from a need in industry for powerful control methods. Throughout the eighties they have evolved towards each other, until one realized that unconstrained MPC and finite-horizon LQG are in fact the same [17]. By MPC we mean the whole set of methods that appeared in the literature at the end of the seventies and throughout the eighties, and that use a model to predict the outputs and a quadratic error criterion to calculate the control inputs over a certain receding horizon. As it would only lead to confusion if we would try to explain them all, we just mention some of the more important ones: Model Predictive Heuristic Control (IDCOM) [105], Dynamic Matrix Control (DMC) [27], Internal Model Control (IMC) [58], Generalized Predictive Control (GPC) [26], etc... The main contributions of MPC consist in the introduction of the receding horizon principle and the study of the influence of constraints on the controller stability.

Joint identification and optimal control: The first attempt to combine identification and control resulted in the well-known adaptive control methods. One of the early results here is Minimum Variance Control theory of [9] and its self-tuning version [10] where the control objective is to minimize the square of the error between the one-step-ahead predicted output $\widehat{y}_{k+1}$ and the corresponding desired output r_{k+1}:

$$J = (\widehat{y}_{k+1} - r_{k+1})^2.$$

The first adaptive versions of LQG-controllers were described in [102], [63]. As mentioned in [107], the "Achilles heel" of self-tuning however, was that the identification algorithm was originally introduced more as an afterthought than as an integral part of the design. In most of the papers from that period, the model was assumed to be more or less readily available and obtained from "some" identification method, mostly recursive least squares. The reason for this was that, for models that are linear in the parameters such as ARX and AR models, linear RLS methods are non-iterative, fast and easy to implement, i.e. the perfect method to be used in an adaptive framework. However, for more general classes of models such as ARMAX models, iterative system identification methods are necessary which were, at that moment, less attractive for adaptive control. In [17], the interplay of recursive least squares identification, LQG-design and robust control was studied and presented in a very elegant way. It served as a basis for a lot of subsequent research.

Several researchers started to realize that the identification step, when designing a controller, is far too important to be neglected, which was more or less what happened up till then. Notions such as "Closed-loop identification" and "Identification for control" became common in the systems and control literature (See e.g. [59], [60], [116], [53] and the references therein). The main idea there is that, in terms of achieved control performance, it is better to design a controller with a model identified from closed-loop data than from open-loop data. Several new iterative identification/control algorithms resulted from this [65], [91], [106], [66]. The philosophy there is that by alternating closed-loop system identification and subsequent controller design, the quality of the model, and therefore the performance of the controller, should increase.

All previously mentioned adaptive LQG-methods could be labeled as "explicit" self-tuning methods because a parameterized model of the plant to be controlled, either a state-space model or a transfer function model, is explicitly calculated. This model, and therefore the whole system identification step, is in fact nothing else than a vehicle for the eventual controller design. Thus, if one could find a method that allows for the calculation of an optimal controller directly from the data, i.e. without the use of a model, the system identification step would become superfluous. This is exactly what we are looking for in the second part of this thesis.

The first LQG-algorithms only using a non-parametric model of the system were based on the Markov parameters of the plant [110], [54], [55]. The idea there is that given the impulse response, i.e. the Markov parameters of the plant, an LQG-controller is calculated. Of course, the problem of measuring or calculating the Markov parameters from input/output data in an accurate way remains. A similar type of result, but tackled from another point of view, is that of [80], [81] where, given free and impulse responses of the unknown process, an LQR-controller is calculated. The drawbacks of this approach are that it is iterative and only holds for deterministic systems.

> The idea of computing an optimal LQG-controller directly from data, and without use of a parametric model, was first developed by Hjalmarsson and collaborators (see [69], [68]) using a technique called Iterative Feedback Tuning (IFT). It is very closely related to prediction error system identification [92] (see also Section 1.1) and the previously mentioned iterative identification/control techniques [65], [91], [106], [66]. Instead of first identifying the model parameters from the input/output data by minimizing the identification criterion, which is what is done in PE identification methods, the IFT-method directly calculates the controller parameters from the input/output data by minimizing the control criterion (1.4). Like PE-methods, IFT requires the solution of a nonlinear optimization problem which is typically solved in an iterative way. It was shown to converge to at least a local minimum, under weak conditions, and has been successfully applied to a range of mechanical and industrial control problems.

Instead of making the link between prediction error system identification and LQG-control design, which is one of the ideas behind the IFT-method, in this thesis we analyze how subspace system identification methods are related to LQG-design. In some ways, the fundamental difference between IFT-controller design and the subspace-based controller design developed in this thesis parallels the fundamental difference between prediction error identification based on criterion minimization and subspace-based identification based on projections and singular value decomposition. The advantages and drawbacks of the two identification methods in terms of optimality, parametric versus non-parametric, local minima versus uniqueness of the solution, etc... are transported from the realm of identification to the realm of control design.

One of the largely underestimated features that make subspace identification so different from other system identification methods is the fact that *first* a Kalman filter estimate of the states can be found even *before* the system parameters are determined [123] (See also Figure 1.1). Since the state plays a crucial role in any state feedback control method, it is clear that this interesting feature can be heavily exploited for control purposes. Up to now no research has been done in this area.

The classical way to solve the LQG-problem in Figure 1.2 is to split it up into different subproblems. The three typical subproblems one will encounter when solving the model-free LQG-control problem are:

System identification: Given measurements of the inputs u_k and the outputs y_k of the unknown system (6.1)-(6.3), one has to find an estimate of the system matrices A, B, C, D and the noise related matrices Q, R and S.

Kalman filter design: Since LQG is a state-space based control method, the knowledge of the state is mandatory. The second step of LQG-design then consists in finding an estimate of the state of the system, given the system parameters and measurements of the past inputs and outputs.

LQR-controller design: Once the Kalman filter estimate of the state is known, it is used to design a linear quadratic controller.

In this thesis, we introduce a new method, based on techniques of linear subspace identification (Chapter 2), to design an LQG-controller of an unknown system directly from the measured input and output data. The method does not follow the typical three-step method described above. No explicit system identification is done since the state-space matrices are never calculated. Moreover, no Kalman filter equations nor LQR-equations have to be solved. The method has some very clear parallels with the subspace algorithms of Chapter 2 in the sense that first an LQ-decomposition of matrices constructed out of the available input/output data is calculated, followed by a singular value decomposition. This is the main idea of Part II of this thesis (See also Figure 1.3).

1.3 Theme III: Bilinear systems

By pointing out the advantages and drawbacks of bilinear systems, the present section gives some guidelines when and where bilinear system theory can be useful. Several references to practical applications of control and/or identification of processes by means of bilinear systems are given. Further, a brief overview of the theoretical results in the bilinear control and identification literature is presented.

1.3.1 Sense and non-sense of bilinear systems

In many practical situations, processes are very often controlled on the basis of linear models (if a model is used at all). When this process is stable and operating around a small region of a fixed working point one can indeed assume that, in most of the cases, it can be approximated by a linear system. In many industrial applications however, especially chemical industry, processes are strongly nonlinear and the variations around the operating point may be significant. Typical examples are the starting up procedure of a chemical distillation column, or the situation where one has to change the set point of a process. Controlling such a system with a linear model might lead to a bad control behavior or even instability.

A solution could then consist in using nonlinear models that can be applied in

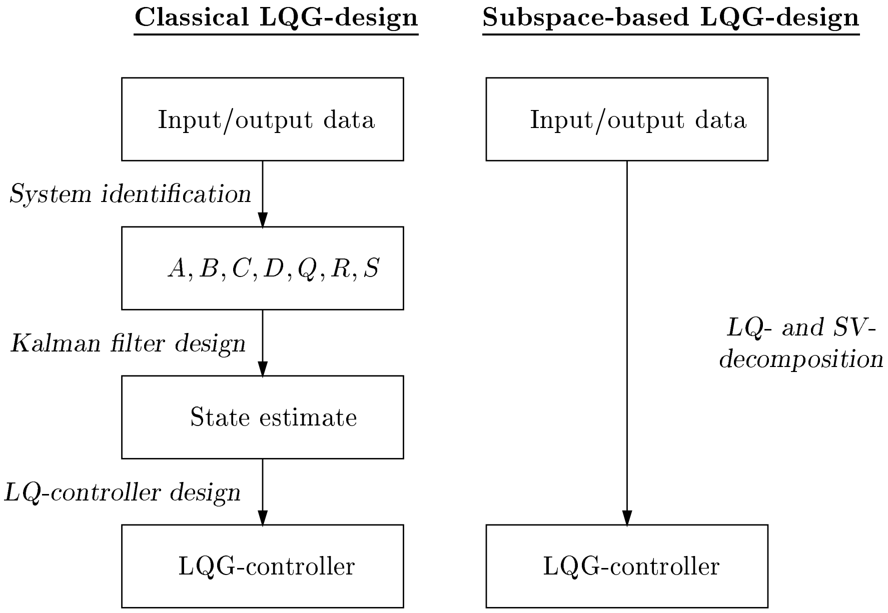


Figure 1.3: **Main idea of Part II of the thesis. Based on subspace system identification, a new algorithm is proposed to solve the model-free LQG-problem of Figure 1.2. In the classical LQG-framework, first a system identification step is performed to find the state-space matrices A, B, C, D, Q, R and S. In a second step, these matrices are used to design a Kalman filter which gives an estimate of the state. Finally, this state estimate is used in a linear quadratic controller. The main novelty of Part II is that, based on techniques from subspace identification, a new method is proposed where these three steps can be short-circuited and replaced by an LQ- and an SV-decomposition of matrices constructed with measurements of the inputs and the outputs of the unknown system (1.1)-(1.3).**

a larger region around a working point. In that case one can either use white box models where, starting from the physics of the system, a dynamic model is built based on differential equations. Although such models can be quite accurate and lead to some insight into the dynamics of the system, they also have some disadvantages:

- They are often very complicated from a mathematical point of view, for instance due to nonlinearities, high order, etc... As a result, when such a model will be used for control purposes, the controller will be complicated also.

- They demand a lot of prior knowledge on the system, which is not always available. In some cases, it is even impossible to describe the physical behavior of the system, simply because no laws have been found.
- In comparison to black box system identification, the design of white box models is quite time consuming.

The above mentioned points are among the reasons why bilinear systems are interesting, since on the one hand they are much easier to analyze than general nonlinear systems such as neural networks, while on the other hand they are more general than linear systems, i.e. they have more degrees of freedom to describe the systems behavior. A nice illustration of this is that bilinear systems are "universal" in the sense that they can approximate, with arbitrary accuracy, any nonlinear system that is linear in the inputs [20], [112]:

$$\begin{aligned} x_{k+1} &= f(x_k) + g(x_k)u_k, \\ y_k &= h(x_k, u_k). \end{aligned}$$

One could conclude by saying that bilinear systems are a good compromise between the accuracy of general nonlinear systems and the mathematical tractability of linear systems.

Before continuing we should first define what we mean by a bilinear system. In the literature, one can find different ways to represent the dynamics of discrete-time bilinear systems:

State-space model: The original, and by far most common representation, is the following set of state-space equations:

$$x_{k+1} = Ax_k + Nu_k \otimes x_k + Bu_k + w_k, \tag{1.5}$$

$$y_k = Cx_k + Du_k + v_k, \tag{1.6}$$

where the measurement noise $v_k \in \mathbb{R}^l$ and the process noise $w_k \in \mathbb{R}^n$ are zero mean, white, stationary sequences with covariance matrix:

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \geq 0. \tag{1.7}$$

This representation is very similar to that of linear systems (1.1)-(1.3) since the only difference is an additional term in the dynamical equation. The matrix $N \in \mathbb{R}^{n \times nm}$ is defined as:

$$N \stackrel{\text{def}}{=} \begin{pmatrix} N_1 & \dots & N_m \end{pmatrix},$$

with $N_i \in \mathbb{R}^{n \times n}$ and the Kronecker product $a \otimes b$ of two vectors $a \in \mathbb{R}^p$

and $b \in \mathbb{R}^q$ defined as:

$$a \otimes b \stackrel{\text{def}}{=} \begin{pmatrix} a_1 b \\ a_2 b \\ \vdots \\ a_p b \end{pmatrix} \in \mathbb{R}^{pq}.$$

The state-space equations (1.5)-(1.6) also explain where the name "bilinear" system comes from. Indeed, they are linear in the control u_k and in the states x_k, but not in the two at the same time. Although very unusual in the literature, a more general form than (1.5)-(1.7) could consist in adding an input/state product-term $u_k \otimes x_k$ in the output equation (1.6) or adding a product-term between the noise and the states. Throughout this thesis we will use the common state-space representation (1.5)-(1.7) given above.

Input/output model: Another form of bilinear systems that is sometimes used in the literature (See e.g. [57], [52], ...) is the following SISO input/output representation:

$$\begin{aligned} y_k &= \sum_{i=1}^{m} a_i y_{k-i} + \sum_{i=1}^{n} b_i u_{k-i} + \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} c_{ij} u_{k-i} y_{k-j} + \\ & \quad + \ldots + \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} d_{ij} e_{k-i} y_{k-j} \end{aligned} \tag{1.8}$$

where e_k is a white, Gaussian, zero-mean, random variable with variance σ_e. This description is not equivalent with the state-space description, which can be understood from the following reasoning. In order to transform the state-space representation (1.5)-(1.7) in the input/output representation (1.8) it would be necessary to express the state x_k as a linear combination of polynomial expansions of past inputs u_k and outputs y_k. As we will discuss in Chapter 4, this is not possible with a finite number of terms in (1.8).

Volterra kernel model: The Volterra kernel representation, which is a generalization of the convolution description of linear time-invariant systems:

$$\begin{aligned} y_k &= y_0 + \sum_{i_1=0}^{k} \mathcal{K}^1_{i_1} u_{k-i_1} + \sum_{i_1=0}^{k} \sum_{i_2=0}^{k} \mathcal{K}^2_{i_1,i_2} u_{k-i_1} u_{k-i_2} + \ldots \\ & \quad + \sum_{i_1=0}^{k} \cdots \sum_{i_k=0}^{k} \mathcal{K}^k_{i_1,\ldots,i_k} u_{k-i_1} \ldots u_{k-i_k} \end{aligned} \tag{1.9}$$

where $\mathcal{K}^q_{i_1,\ldots,i_q}$ are the qth order Volterra Kernels. Note that the kernels of *all* orders are necessary to describe the bilinear dynamics, and not just the second order kernels as one might intuitively think!

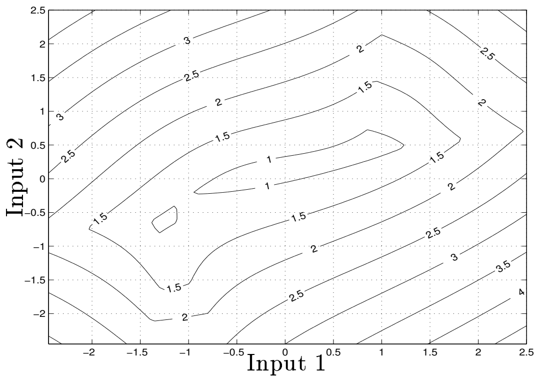


Figure 1.4: **The steady-state stability of a two input 5th order bilinear system is studied. The lines represent the iso-lines in the input space with the same maximum absolute values of the eigenvalues of $A + N_1 u_1 + N_2 u_2$ as a function of the two inputs u_1 and u_2. One can see that the set of inputs u_1 and u_2 that stabilizes the system (i.e. $max||abs(eig(A + N1u_1 + N_2u_2))|| < 1$) at steady state is neither convex nor connected.**

In this thesis we will mainly use the state-space representation of bilinear systems.

Although it is clear from the different descriptions above that bilinear systems are close to linear systems, one should be careful when stating that bilinear systems are "simple". It is true that bilinear systems are probably the most simple nonlinear extension of linear systems but, especially from a control point of view, their analysis can be much tougher than for linear systems. To illustrate this let us consider the, apparently easy, problem of the steady-state stability of bilinear systems which consists in finding the set of constant inputs that stabilize a two input, one output, 5th order bilinear system at steady-state. Therefore, the maximum eigenvalue of the matrix $A + N_1 u_1 + N_2 u_2$ has to be calculated as a function of the constant inputs u_1 and u_2. As illustrated in Figure 1.4, the set of inputs for which the maximum eigenvalue of $A + N_1 u_1 + N_2 u_2$ is smaller than 1 in absolute value, is neither convex nor connected.

The practical and theoretical interest of bilinear systems has been pointed out by many authors (See [21], [95] and the references therein). Practical applications where bilinear system theory has been used for identification and/or control purposes are numerous and have been reported on in several papers. Among others we can find the modeling of a distillation column [36], [50], the nuclear fission in a reactor [11], an experimental winding process [14] and the predictive control of a distillation column [94]. An exhaustive list can be found in the overview book on bilinear systems [95].

A rather interesting question for the practician is when to recur to bilinear systems or not. Although it is very difficult to give an unambiguous answer to this question, the following general guidelines could help deciding:

- If the system dynamics are clearly nonlinear, and this nonlinearity can not by modeled in a satisfying way by linear systems, then bilinear systems may be used. For instance the start up procedure of a distillation column is clearly nonlinear and can be modeled quite well by a bilinear system.
- If the order of the, presumed, bilinear system is low ($n < 4$) and the number of inputs is limited ($m < 3$) bilinear system theory may be considered. If the order or the number of inputs is too high, one will always run into a dimensionality problem that seems to be inherent to bilinear system theory. For instance, the number of Volterra kernels needed to describe the dynamical behavior of a bilinear system is proportional to $\sum_{i=0}^{n-1}(m+1)^i lm$ and is therefore an exponential function of the system order n.

One could conclude by saying that, for low order systems with a restricted number of inputs and a clearly nonlinear behavior, bilinear system theory is justifiable.

1.3.2 Bilinear identification

When comparing the system identification literature of bilinear systems with that of linear systems, one can clearly observe a similar development. The first real bilinear system identification techniques were, like for linear systems, to be found among the prediction error methods [57], [56], [52] and instrumental variable methods [2]. They all consider bilinear systems in an input/output form (1.8). The reasons for this are twofold:

- A model of the form (1.8) is linear in the parameters.
- Most of the linear prediction error system identification techniques were developed for input/output models.

As we mentioned earlier, the problem with bilinear models of the form (1.8) is that they are not equivalent with the classical state-space form (1.5)-(1.7).

Besides the PEM, a completely different class of methods, based on orthogonal basis functions, appeared. These methods study the identification of bilinear systems in a continuous-time state-space form:

$$\dot{x} = Ax + Nu \otimes x + Bu.$$

The idea there is basically to *approximate* the input u and the state derivative $\dot{x}$ by a linear combination of orthogonal basis functions. The advantage of doing so is that x and $u \otimes x$ are still linear in the unknown parameters. However, this way of studying the bilinear identification problem is very restrictive because no noise is considered and it only gives an approximate solution. Many types of orthogonal functions have been tested to identify bilinear systems in this way. Among others, one can find in the literature Walsh functions [79], block pulse functions [24], Shifted Legendre polynomials [71], Laguerre polynomials [104], Shifted Chebyshev series [25], Generalized orthogonal polynomials [133], etc....

Up till now no attempts have been made to study the bilinear system identification problem from a subspace identification point of view i.e. how to find a state-space model (1.5)-(1.7) directly from the available input/output data. As we mentioned in Section 1.1, realization theory played an important role in the development of linear subspace identification. Realization theory of deterministic bilinear systems appeared in [73], [74] and solved the problem of finding the state-space matrices of a system given the Volterra-kernels of that system. Realization of stochastic bilinear systems only appeared more than ten years later [35]. One of the early realization-based bilinear system identification algorithms is described in [11]. The aim there was to estimate the first and second order Volterra kernels of the system by calculating the cross correlations between certain generalized inputs and outputs. A standard bilinear realization algorithm can then be used to find the state-space matrices A, N, B, C, D. A result following the same line of thought, but extended to bilinear systems with noise and estimating the Volterra kernels of all orders, can be found in [72]. One of the main goals in this thesis is to extend linear subspace identification to bilinear systems.

1.3.3 Bilinear control

Control of bilinear systems has been, just like linear control, a long-standing research topic. It would be far beyond the scope of this thesis to give a complete overview of the literature. We just give a flavor of the different existing techniques in a more or less chronological order, together with some references:

- The reachability from the origin of bilinear systems was studied in [73] and the references therein. Although several papers consider this problem for special classes of bilinear systems [115], [99], [84], [82], the general controllability problem of bilinear systems of the form (1.5)-(1.6) is still open however.
- Although optimal control of bilinear systems was studied in several papers, e.g. [114], [34], [62], [16], it is basically an unsolved problem. Most of these results only consider special classes of bilinear systems or rely on

quadratic cost functionals modified by inclusions of additional nonnegative state-dependent penalizing functions. An *iterative* approximation procedure for the general finite-time bilinear optimal control problem with a strictly quadratic cost function was described in [1].

- Model predictive control of, once again, only special classes of bilinear systems was studied in [100], [135], [134], [51]. MPC for general bilinear systems (1.5)-(1.7) has not been considered yet.

In the second part this thesis, one of our goals will be to combine the linear subspace-based LQG-design methods of Chapter 6 on the one hand and the bilinear subspace identification techniques of Chapter 3, 4 on the other, in order to develop a new model-free data-based bilinear predictive control algorithm.

1.4 Outline and contributions

Figure 1.5 illustrates the structure of the thesis and shows the links between the different chapters.

Chapter 1: Introduction

A general introduction to the problems tackled in the thesis is presented and split up into different subproblems (See Table 1.1 and Figure 1.5).

Part I: Subspace identification

Chapter 2: Linear subspace identification - An overview

The first part of this chapter merely gives an overview of the most important results in the field of linear subspace system identification. Parts of this chapter can also be found in the overview papers [29], [123], [32], [48] and [49]. First the subspace system identification problem and some properties of linear systems (1.1)-(1.3) are recalled. The data block-Hankel matrices that are typically used in subspace identification are presented. Further, the three main steps in every subspace system identification algorithm are outlined and the importance of the Kalman filter within the subspace identification framework is stressed. Finally, three of the most popular subspace identification algorithms in the literature are compared with each other on the basis of accuracy (prediction/simulation error). The results of this comparison have also been presented in [44].

Chapter 3: Subspace identification of bilinear systems with white inputs

In this chapter the theory of linear subspace identification is extended to

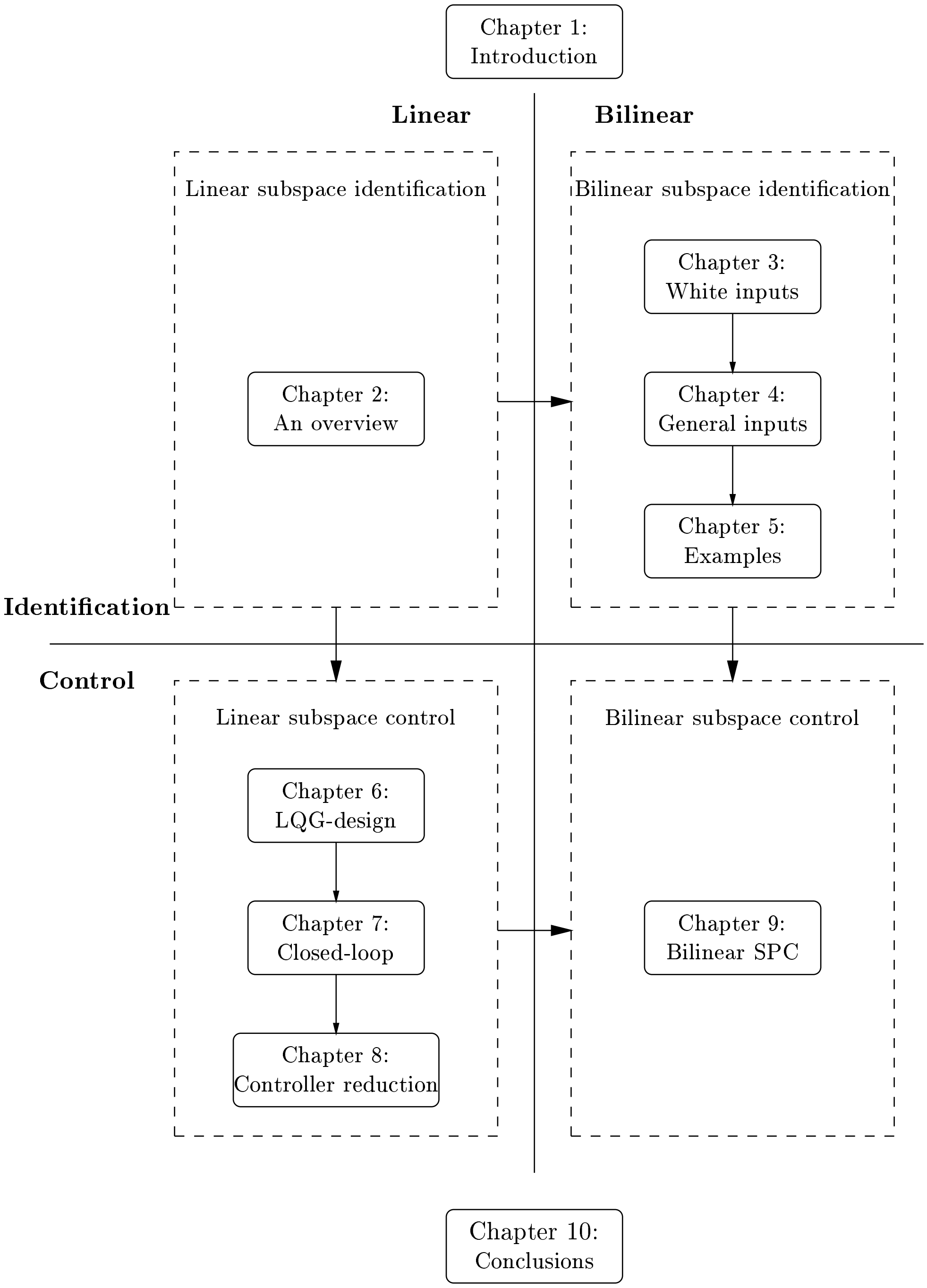


Figure 1.5: **Structure of the thesis.**

bilinear systems. It is shown that, under the assumption that the inputs of the available data set are white and independent of the measurement and process noise, most of the results of linear subspace identification still hold for bilinear systems. First, some structural properties of bilinear systems are presented, after which the generalized data Hankel matrices used in bilinear subspace identification are introduced. Further it is also shown that, just like for linear subspace identification, there exists a direct link between the subspace projection and the Kalman filter states of the system. Finally, the resulting bilinear subspace identification algorithm is presented. It has the same structure and ingredients (LQ- and SV-decomposition) as the typical linear subspace identification algorithms, and is in fact an extension of the latter. Most of the results presented in this chapter have appeared or will appear in [45], [46], [50].

Chapter 4: Subspace identification of bilinear systems with general inputs

The main assumption of the previous chapter, i.e. that the inputs should be white, can sometimes be quite restrictive. In the present chapter we try to get rid of this assumption by extending the algorithm of Chapter 3 to a more general framework. The resulting algorithm is also very similar to the existing linear subspace identification algorithms. Although the algorithm gives a biased estimate of the system, it is shown that it is a direct extension of the linear subspace algorithms of Chapter 2 on the one hand and of the bilinear algorithms with white inputs of Chapter 3 on the other. The paper [47] also reports on these results.

Chapter 5: Some examples

In this chapter we provide several examples where a bilinear model is identified using the bilinear subspace identification algorithms of the previous chapters. An interesting example is the identification of a distillation column. The available data is obtained from a nonlinear white box model of a two-component distillation column. When identifying a linear and a bilinear model from the same available input/output data set, it turns out that the bilinear model is more suited to represent the dynamics than the linear model [45], [46], [50], [47].

Part II: Subspace predictive control

Chapter 6: Model-free subspace-based LQG-design

The importance of the Kalman filter and the principle of splitting up the data matrices into "past" and "future" are important conceptual novelties in the framework of subspace system identification. These are in fact ingredients that can also be found in LQG-controllers. In the present chapter these links between subspace system identification and LQG-control are elaborated on. It turns out that, in a similar way as the process model is identified from the data in subspace *system* identification, the *controller* can be "identified" from the available data. The resulting algorithm uses the same numerical techniques and has a structure that is very similar to subspace system identification algorithms. However, instead of finding a model of the unknown process, one directly finds an LQG-controller for the unknown process. Since no explicit process model is identified we refer to the method as "model-free" LQG-design. The results of this chapter have been or will be published in [40], [39] and [42].

Chapter 7: Closed-loop subspace-based LQG-design

The method as presented in chapter 6 will work fine under the assumption that the available data set, used to calculate the controller with, was measured under open-loop conditions. However, it is sometimes impossible to do open-loop experiments. This can for instance occur if the system is unstable in open-loop, or if opening the loop would be unacceptable from an economical point of view. In these cases one only disposes of closed-loop measurements of the inputs and the outputs of the system. The present chapter tackles this problem. Based on existing results of closed-loop subspace system identification, an extension of the subspace-based LQG-algorithm of the previous chapter is presented that can handle closed-loop data. This result was also presented in [41].

Chapter 8: Subspace-based balanced controller reduction

A disadvantage of the subspace-based LQG-algorithms presented in the two previous chapters is that the resulting controllers are of a high order. The order equals the number of block rows in the data block-Hankel matrices used in the subspace projection, which can be quite high (typically larger than 20). In this chapter an extension of the previous algorithms is presented to obtain a reduced order controller. By adding specific weights to the subspace algorithm, it is possible to obtain a controller that is directly in balanced form. According to the choice of the weights it is even possible to obtain an open-loop or a closed-loop balanced controller.

Chapter 9: Model-free subspace-based predictive control of bilinear systems

The last chapter of Part II combines the bilinear subspace identification algorithm of Chapter 4 and the subspace-based LQG-controller algorithm of Chapter 6 into a new algorithm for the design of predictive controllers for bilinear systems. By the hand of a simple example it is illustrated that this algorithm can bring substantial improvement of the control performance, when compared to linear subspace-based LQG-control.

Chapter 10: Conclusions and open problems

Part I

Subspace identification

Chapter 2

Linear subspace identification: An overview

In this chapter we will first describe the general concepts of subspace identification (For a more thorough analysis of subspace system identification we refer the reader to the book [123]). The problem is defined and some basic definitions, notations and geometric tools needed to understand the theory are recalled. Further, the three basic steps all subspace methods consist of are presented. The first step is an LQ-decomposition, the second step is an SV-decomposition and the final step consists in extracting the state-space matrices from the first two steps. The importance of the link between the projection step in subspace system identification and Kalman filter theory is recalled. We also make a distinction between subspace algorithms using the states and those using the observability matrix to extract the state-space matrices. Finally, the main subspace algorithms existing in the literature are analyzed in a unifying framework and are compared with each other in a simulation study when applied to 15 different real-life data sets.

2.1 Problem description

The main problem treated by linear subspace system identification is represented in Figure 2.1.

2.2 Stochastic properties of linear systems

2.2.1 The deterministic and the stochastic subsystem

A linear system of the general form (2.1)-(2.2) can be split up into the sum of a deterministic system (superscript d) and a stochastic system (superscript s). The states and the outputs of the global system are simply the sum of the states of the stochastic and the deterministic parts:

Open-loop linear subspace identification problem

Given *a large number of measurements of the input u_k and the output y_k generated by the unknown system:*

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{2.1}$$
$$y_k = Cx_k + Du_k + v_k \tag{2.2}$$

with

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \geq 0, \tag{2.3}$$

determine *an estimate of the system order n, the system matrices A, B, C, D up to within a similarity transformation and the noise covariance matrices Q, R, S.*

Figure 2.1: **The linear subspace system identification problem formulation**

$$x_k = x_k^d + x_k^s,$$
$$y_k = y_k^d + y_k^s.$$

Deterministic subsystem:

$$x_{k+1}^d = Ax_k^d + Bu_k, \tag{2.4}$$
$$y_k^d = Cx_k^d + Du_k. \tag{2.5}$$

This system describes the influence of the deterministic input u_k on the deterministic output y_k^d.

Stochastic subsystem:

$$x_{k+1}^s = Ax_k^s + w_k, \tag{2.6}$$

$$y_k^s = Cx_k^s + v_k \tag{2.7}$$

with

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix}\begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}\delta_{pq} \geq 0. \tag{2.8}$$

This subsystem describes the influence of the noise sequences w_k and v_k on the stochastic output y_k^s.

2.2.2 Forward innovation representation

Deterministic realization theory

From realization theory [70], [136] it is well-known that, in the case of the deterministic subsystem (2.4)-(2.5), the system matrices A, B, C, D can be found directly from the impulse response h_i of the system:

$$\begin{aligned} h_i &= CA^{i-1}B, \;\; i > 0 \\ h_0 &= D \end{aligned}$$

by taking the singular value decomposition of the block-Hankel matrix $\mathcal{M}_i$ with $i > n$ containing the Markov parameters of the system:

$$\begin{aligned} \mathcal{M}_i &\overset{\text{def}}{=} \begin{pmatrix} h_1 & h_2 & h_3 & \dots & h_i \\ h_2 & h_3 & h_4 & \dots & h_{i+1} \\ h_3 & h_4 & h_5 & \dots & h_{i+2} \\ \dots & \dots & \dots & \dots & \dots \\ h_i & h_{i+1} & h_{i+2} & \dots & h_{2i-1} \end{pmatrix} \\ &= \begin{pmatrix} C \\ CA \\ CA^2 \\ \dots \\ CA^{i-1} \end{pmatrix} \begin{pmatrix} B & AB & A^2B & \dots & A^{i-1}B \end{pmatrix} \\ &\overset{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}. \end{aligned}$$

One possible realization of the system matrices is then:

$$\begin{aligned} A &= S_1^{-1/2} U_1(1 : end - l, :)^{\dagger} U_1(l+1 : end, :) S_1^{1/2}, \\ B &= V_1^T(:, 1 : m), \\ C &= U_1(1 : l, :). \end{aligned}$$

D is determined as the first element of the impulse response: $D = h_0$. It should be noted that this problem is under determined in the sense that A, B and C can only be found up to a similarity transformation, but clearly this does not matter.

Stochastic realization theory

Although the stochastic realization problem is a bit more complicated than the deterministic one, it is solved in a similar way [4]. The problem here is to find the system matrices of the stochastic subsystem (2.6)-(2.8), i.e. A, C, Q, R, S, from the output covariance sequence Λ_i:

$$\begin{aligned}\Lambda_i &\stackrel{\text{def}}{=} \mathbf{E}[y_k^s (y_{k-i}^s)^T] \\ &= CA^{i-1}G, \text{ for } i > 0,\end{aligned}$$

where

$$G \stackrel{\text{def}}{=} \mathbf{E}[x_{k+1}^s (y_k^s)^T].$$

Intuitively one can see that, when comparing the stochastic with the deterministic realization problem, more parameters have to be determined from the same amount of data (A, C, Q, R, S instead of A, B, C, D). Therefore the similarity transformation will not be the only indeterminacy.

In the same way as A, B and C were calculated in the deterministic realization problem, A, G and C are found by taking the SVD of a block-Hankel matrix $\mathcal{C}_i$ ($i > n$) containing the output covariance sequence Λ_i:

$$\begin{aligned}\mathcal{C}_i &\stackrel{\text{def}}{=} \begin{pmatrix} \Lambda_1 & \Lambda_2 & \Lambda_3 & \ldots & \Lambda_i \\ \Lambda_2 & \Lambda_3 & \Lambda_4 & \ldots & \Lambda_{i+1} \\ \Lambda_3 & \Lambda_4 & \Lambda_5 & \ldots & \Lambda_{i+2} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \Lambda_i & \Lambda_{i+1} & \Lambda_{i+2} & \ldots & \Lambda_{2i-1} \end{pmatrix} \\ &= \begin{pmatrix} C \\ CA \\ CA^2 \\ \ldots \\ CA^{i-1} \end{pmatrix} \begin{pmatrix} G & AG & A^2G & \ldots & A^{i-1}G \end{pmatrix} \\ &\stackrel{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}.\end{aligned}$$

One possible realization of the system matrices is then:

$$\begin{aligned} A &= S_1^{-1/2} U_1(1 : end - l, :)^{\dagger} U_1(l + 1 : end, :) S_1^{1/2}, \\ G &= V_1^T(:, 1 : m), \\ C &= U_1(1 : l, :). \end{aligned}$$

The problem that remains to be solved now is that of finding Q, R, S from A, G, C and Λ_0. It can be shown that these matrices are related through the following set of equations:

$$\Lambda_0 = C\Sigma^s C^T + R,$$

$$\begin{aligned} G &= A\Sigma^s C^T + S, \\ \Sigma^s &= A\Sigma^s A^T + Q \end{aligned}$$

where Σ^s is the state covariance matrix: $\Sigma^s \stackrel{\text{def}}{=} \mathbf{E}[x_k^s (x_k^s)^T]$. It can be seen that, if A, G, C, Λ_0 are given, the above equations have a unique solution for Q, R, S only if Σ^s is known. Therefore the origin of the indeterminacy, besides the similarity transformation, is also the state covariance matrix Σ^s. As it was shown in [38], Σ^s can be chosen in an arbitrary way as long as it is an element of the following closed convex and bounded set:

$$\mathcal{F} \leq \Sigma^s \leq \mathcal{B}^{-1}$$

where $\mathcal{F}$ and $\mathcal{B}$ are the solutions of the following algebraic Riccati equations:

$$\begin{aligned} \mathcal{F} &= A\mathcal{F}A^T + (G - A\mathcal{F}C^T)(\Lambda_0 - C\mathcal{F}C^T)^{-1}(G - A\mathcal{F}C^T)^T, \\ \mathcal{B} &= A^T\mathcal{B}A + (C^T - A^T\mathcal{B}G)(\Lambda_0 - G^T\mathcal{B}G)^{-1}(C^T - A^T\mathcal{B}G)^T. \end{aligned}$$

Forward innovation form

If the state covariance matrix Σ^s is chosen equal to P, the system (2.1)-(2.3) is in the so-called *forward innovation* form (Figure 2.2).

Forward innovation form:

$$x_{k+1} = Ax_k + Bu_k + Ke_k, \quad (2.9)$$

$$y_k = Cx_k + Du_k + e_k \quad (2.10)$$

with

$$\mathbf{E}[e_k e_k^T] = \Lambda_0 - C\Sigma^s C^T, \quad (2.11)$$

$$\Sigma^s = A\Sigma^s A^T + (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}(G - A\Sigma^s C^T)^T, \quad (2.12)$$

$$K = (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}. \quad (2.13)$$

Figure 2.2: **Forward innovation representation of the linear system (2.1)-(2.3).**

Remark 2.2.1 From Figure 2.2 it should be clear that the forward innovation form of a system is in fact nothing else than the steady-state Kalman filter of that system where K is the steady-state Kalman filter gain and e_k the innovations. The importance of this observation will become clear in Section 2.4.1 where the link between the subspace identification algorithms and the Kalman filter states is made.

Remark 2.2.2 For the theoretical derivations in this thesis we will use the forward innovation representation (2.9)-(2.11) of the system instead of the equations (2.1)-(2.3). Since the output covariances Λ_i of both representations are the same, and are therefore input/output equivalent, and because in system identification one is mainly interested in the input/output behavior of a system, it does not really matter whether (2.9)-(2.11) or (2.1)-(2.3) is used. But why would we want to use the innovation representation (2.9)-(2.11) instead of (2.1)-(2.3)? The main reason for this is that the model, obtained with subspace system identification, will be typically in forward innovation form. As a result, the theoretical derivations and interpretations of the results turn out to be more simple when the forward innovation form is used.

2.3 The subspace structure of linear systems

In this section the notations and concepts used in this chapter are explained. First we define how the data used in the algorithms is structured into block-Hankel matrices. Finally the matrix input/output equations relating these different data matrices are presented.

2.3.1 System related matrices

We define the following matrices:

- The extended observability matrix Γ_i:

$$\Gamma_i \stackrel{\text{def}}{=} \begin{pmatrix} C \\ CA \\ CA^2 \\ \dots \\ CA^{i-1} \end{pmatrix} . \tag{2.14}$$

- The reversed extended controllability matrices Δ_i^d and Δ_i^s of the deterministic subsystem (2.4)-(2.5) and the stochastic subsystem (2.6)-(2.8) in innovation form:

$$\Delta_i^d \stackrel{\text{def}}{=} \begin{pmatrix} A^{i-1}B & A^{i-2}B & \dots & AB & B \end{pmatrix} . \tag{2.15}$$

$$\Delta_i^s \stackrel{\text{def}}{=} \begin{pmatrix} A^{i-1}K & A^{i-2}K & \dots & AK & K \end{pmatrix} . \tag{2.16}$$

- The deterministic lower triangular block-Toeplitz matrix H_i^d:

$$H_i^d \stackrel{\text{def}}{=} \begin{pmatrix} D & 0 & 0 & \ldots & 0 \\ CB & D & 0 & \ldots & 0 \\ CAB & CB & D & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ CA^{i-2}B & CA^{i-3}B & CA^{i-4}B & \ldots & D \end{pmatrix} . \quad (2.17)$$

- The stochastic lower triangular block-Toeplitz matrix H_i^s:

$$H_i^s \stackrel{\text{def}}{=} \begin{pmatrix} I & 0 & 0 & \ldots & 0 \\ CK & I & 0 & \ldots & 0 \\ CAK & CK & I & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ CA^{i-2}K & CA^{i-3}K & CA^{i-4}K & \ldots & I \end{pmatrix} . \quad (2.18)$$

2.3.2 Data block-Hankel matrices

The input, output and innovation block-Hankel matrices are defined as:

$$U_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} u_0 & u_1 & \ldots & u_{j-1} \\ u_1 & u_2 & \ldots & u_j \\ \ldots & \ldots & \ldots & \ldots \\ u_{i-1} & u_i & \ldots & u_{i+j-2} \end{pmatrix} , \quad (2.19)$$

$$Y_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} y_0 & y_1 & \ldots & y_{j-1} \\ y_1 & y_2 & \ldots & y_j \\ \ldots & \ldots & \ldots & \ldots \\ y_{i-1} & y_i & \ldots & y_{i+j-2} \end{pmatrix} , \quad (2.20)$$

$$E_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} e_0 & e_1 & \ldots & e_{j-1} \\ e_1 & e_2 & \ldots & e_j \\ \ldots & \ldots & \ldots & \ldots \\ e_{i-1} & e_i & \ldots & e_{i+j-2} \end{pmatrix} , \quad (2.21)$$

where we assume that the number of columns in these matrices is $j \to \infty$.

For convenience and short hand notation, we define:

$$U_p \stackrel{\text{def}}{=} U_{0|i-1}, \qquad U_f \stackrel{\text{def}}{=} U_{i|2i-1}, \quad (2.22)$$

$$Y_p \stackrel{\text{def}}{=} Y_{0|i-1}, \qquad Y_f \stackrel{\text{def}}{=} Y_{i|2i-1}, \quad (2.23)$$

$$E_p \stackrel{\text{def}}{=} E_{0|i-1}, \qquad E_f \stackrel{\text{def}}{=} E_{i|2i-1}, \quad (2.24)$$

where the subscript p and f denote the past and the future respectively. The matrix containing the past inputs U_p and outputs Y_p will be denoted by W_p:

$$W_p \stackrel{\text{def}}{=} \begin{pmatrix} Y_p \\ U_p \end{pmatrix} .$$

Finally, we define the past and future state sequence X_p and X_f:

$$X_p = X_0 \stackrel{\text{def}}{=} \begin{pmatrix} x_0 & x_1 & x_2 & \dots & x_{j-1} \end{pmatrix}, \quad (2.25)$$
$$X_f = X_i \stackrel{\text{def}}{=} \begin{pmatrix} x_i & x_{i+1} & x_{i+2} & \dots & x_{i+j-1} \end{pmatrix}. \quad (2.26)$$

2.3.3 Matrix input/output equations

The following matrix input/output equations [28] played, and still play, a very important role in the development of subspace identification. They directly derive from the system state-space equations (2.9)-(2.10) and express how the previously defined data block-Hankel matrices and state sequences are related to each other:

$$Y_p = \Gamma_i X_p + H_i^d U_p + H_i^s E_p, \quad (2.27)$$
$$Y_f = \Gamma_i X_f + H_i^d U_f + H_i^s E_f, \quad (2.28)$$
$$X_f = A^i X_p + \Delta_i^d U_p + \Delta_i^s E_p. \quad (2.29)$$

2.3.4 Subspace projections

Subspace system identification uses a specific class of geometrical tools, called orthogonal and oblique projections. We now recall some of the most important properties of these projections. For a more thorough analysis we refer the reader to [12], [3].

In what follows, we will use the matrices $A \in \mathbb{R}^{p\times j}, B \in \mathbb{R}^{q\times j}$ and $C \in \mathbb{R}^{r\times j}$ as dummy variables, meaning that they are only "local" variables in this subsection, not to be confused with the system matrices A, B and C we have defined before.

Orthogonal projection

The rows of A can be decomposed as linear combination of the rows of B and a matrix orthogonal to B. This can be written as:

$$A = L_B B + L_{B^\perp} B^\perp$$

where the rows of $B^\perp$ span the orthogonal complement of the row space of B. The orthogonal projection of the row space of A onto the row space of B is $L_B B$.

Definition 2.3.1 (Orthogonal projection)
The orthogonal projection of the row space of A into the row space of B is denoted by A/B and defined as:

$$A/B = AB^\dagger B.$$

$A/B^\perp$ is the projection of the row space of A onto $B^\perp$. We have that $A/B^\perp = A - A/B$.

Oblique projection

The rows of A can also be decomposed as linear combination of the rows of B, C and a third matrix orthogonal to B and C:

$$A = L_B B + L_C C + L_{B^\perp, C^\perp} \begin{pmatrix} B \\ C \end{pmatrix}^\perp .$$

The matrix $L_C C$ is defined as the oblique projection of the row space of A along the row space of B on the row space of C:

$$A/_B\, C \stackrel{\text{def}}{=} L_C C.$$

Figure 2.3 illustrates the oblique projection in the j-dimensional space. The name *oblique* refers to the non-orthogonal projection direction. The oblique projection can also be interpreted through the following recipe: Project the row space of A orthogonally on the joint row space of B and C, and decompose the result along the row space of C. Mathematically, the orthogonal projection of the row space of A on the joint row space of B and C can be stated as:

$$A/\begin{pmatrix} C \\ B \end{pmatrix} = A \begin{pmatrix} C \\ B \end{pmatrix}^\dagger \begin{pmatrix} C \\ B \end{pmatrix}.$$

Decomposing this expression along the row spaces of B and C leads to the following definition for the oblique projection:

Definition 2.3.2 (Oblique projection)
The oblique projection of the row space of $A \in \mathbb{R}^{p \times j}$ along the row space of $B \in \mathbb{R}^{q \times j}$ on the row space of $C \in \mathbb{R}^{r \times j}$ is defined as:

$$A/_B\, C \stackrel{\text{def}}{=} A \left[\begin{pmatrix} C \\ B \end{pmatrix}^\dagger \right] (:, 1:r) C.$$

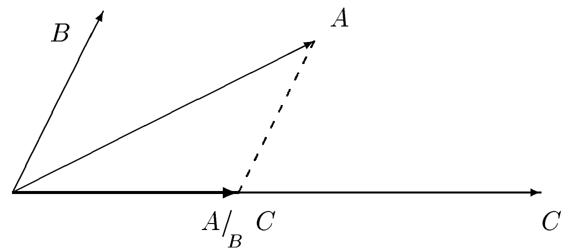


Figure 2.3: **Interpretation of the oblique projection in the j-dimensional space ($j = 2$ in this case). The oblique projection is formed by projecting the row space of A along the row space of B on the row space of C.**

Corollary 2.3.3 (Oblique projection)

The oblique projection of the row space of $A \in \mathbb{R}^{p\times j}$ along the row space of $B \in \mathbb{R}^{q\times j}$ on the row space of $C \in \mathbb{R}^{r\times j}$ can also be written as:

$$A/_B\, C \;=\; \left[A/B^{\perp}\right]\left[C/B^{\perp}\right]^{\dagger} C.$$

2.4 The three main steps in subspace identification

In this section we will explore the three main steps that all subspace algorithms consist of (see Figure 2.4). The first step consists in calculating the projection of the row space of specific data Hankel matrices. This is typically done with an LQ-decomposition. In the second step, the singular value decomposition of this projection is calculated. This directly leads to the observability matrix Γ_i and a Kalman filter estimate $\widehat{X}_f$ of the state sequence X_f. In the third and last step, the system matrices A, B, C, D and noise covariance matrices Q, S, R are determined from the observability matrix Γ_i and/or the estimated state sequence $\widehat{X}_f$. As shown in Figure 2.4, a clear distinction can be made between the algorithms that use the extended observability matrix Γ_i to obtain the state-space matrices, and those using the estimated state sequence $\widehat{X}_f$.

2.4.1 First step: LQ-decomposition

In one way or another, all subspace system identification methods start from the previously presented matrix input/output equation (2.28). When examining this equation we see that the $\Gamma_i X_f$-term is a particularly interesting term since

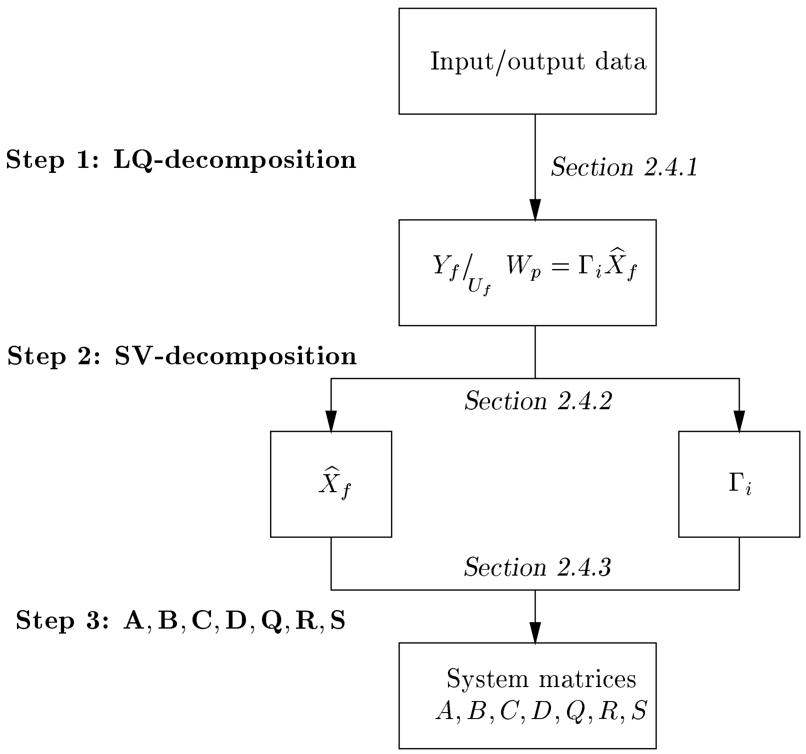


Figure 2.4: **Subspace identification algorithms always consist of three main steps. In Section 2.4.1 it is explained how, using an LQ-decomposition, the term $\Gamma_i \widehat{X}_f$ is found from the available input/output data while Section 2.4.2 shows how, using a singular value decomposition, this term can be split up into the extended observability matrix Γ_i and an estimate $\widehat{X}_f$ of the state sequence X_f. It is shown that $\widehat{X}_f$ is in fact a nonsteady-state Kalman filter estimate of X_f. Section 2.4.3 finally gives two possible algorithms for the third step of subspace algorithms, where the system matrices A, B, C, D and the noise covariance matrices Q, R, S are calculated. The first one uses the extended observability matrix Γ_i while the other one uses the Kalman filter state sequence $\widehat{X}_f$.**

its rank is equal to the system order n, and moreover the knowledge of either Γ_i or X_f directly leads to the state-space matrices A, B, C, D, Q, R, S (See Section 2.4.3). In this section we will elaborate on the first step in subspace system identification which consists in extracting the $\Gamma_i X_f$-term from (2.28).

This first step of the subspace identification problem can also be interpreted as follows: *given* the past inputs and outputs W_p and the future inputs U_f, *find* the optimal prediction $\widehat{Y}_f$ of the future outputs Y_f:

$$\widehat{Y}_f = L_w W_p + L_u U_f. \tag{2.30}$$

The least squares prediction $\widehat{Y}_f$ of Y_f can be found as the solution of the following least squares problem:

$$\min_{L_w, L_u} ||Y_f - \begin{pmatrix} L_w & L_u \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix}||_F^2$$

which can be easily solved by calculating the orthogonal projection of the row space of Y_f onto the row space spanned by W_p and U_f:

$$\widehat{Y}_f = Y_f / \begin{pmatrix} W_p \\ U_f \end{pmatrix}. \tag{2.31}$$

$$\tag{2.32}$$

With an oblique projection, $L_w W_p$ and $L_u U_f$ can be found separately:

$$\begin{aligned} Y_f /_{U_f} W_p &= L_w W_p, \\ Y_f /_{W_p} U_f &= L_u U_f. \end{aligned} \tag{2.33}$$

As we will analyze into more details in Section 2.5, a numerically efficient and robust way to implement this projection is with an LQ-decomposition.

The main contribution of the next two theorems is to show that the above projection (2.33) equals $\Gamma_i \widehat{X}_f$ which is the product between the extended observability matrix Γ_i and a nonsteady-state Kalman filter estimate $\widehat{X}_f$ of the state sequence X_f. Theorem 2.4.1 is a slightly modified version of the one presented in [123] (p. 101). It explains how the classical recursive nonsteady-state Kalman filter equations can be expressed in a closed form. The Theorem in [123] gives a similar expression. However, there the Kalman filter state is nonsteady, not only because the initial Kalman filter estimate $\widehat{x}_0$ is different from the steady-state Kalman filter estimate, but also because the Kalman gain is different from the steady-state Kalman gain K. This implies that an additional initial conditions Σ_0^s is necessary which is used as initial solution of the Riccati equation (2.12).

Theorem 2.4.1 (Nonsteady-state Kalman filter)
Given:

- *the initial state estimate* $\widehat{x}_0$
- *the input and output measurements* $u_0, y_0, \ldots, u_{i-1}, y_{i-1}$

then the nonsteady-state Kalman filter estimate $\widehat{x}_i$, defined by the recursive Kalman filter equations (2.9)-(2.13), can be written in closed form as:

$$\widehat{x}_i = \left(A^i - \Delta_i^s H_i^{-s}\Gamma_i \mid \Delta_i^s H_i^{-s} \mid \Delta_i^d - \Delta_i^s H_i^{-s} H_i^d \right) \begin{pmatrix} \widehat{x}_0 \\ \hline y_0 \\ \vdots \\ y_{i-1} \\ \hline u_0 \\ \vdots \\ u_{i-1} \end{pmatrix} . \quad (2.34)$$

Proof: The proof directly follows from the matrix input/output equations (2.27)-(2.29). It suffices to replace E_p of (2.27) in (2.29), leading to

$$\widehat{X}_f = \left(A^i - \Delta_i^s H_i^{-s}\Gamma_i \mid \Delta_i^s H_i^{-s} \mid \Delta_i^d - \Delta_i^s H_i^{-s} H_i^d \right) \begin{pmatrix} \widehat{X}_p \\ Y_p \\ U_p \end{pmatrix} .$$

(2.34) corresponds to the first column of this matrix equation. ■

Before we proceed, we need the following technical definition:

Definition 2.4.2 (Persistency of excitation [93])
The input sequence $u_k \in \mathbb{R}^m$ is persistently exciting of order $2i$ if the matrix $U_{0|2i-1}$ is of full row rank i.e. $2mi$.

Theorem 2.4.3 (Main linear subspace identification theorem [123])
Under the assumptions that:

- *the deterministic input u_k is uncorrelated with the noise e_k*
- *the input u_k is persistently exciting of order $2i$ (Definition 2.4.2)*
- *the number of available data points is large ($j \to \infty$)*
- *the noise e_k is not identically zero*

we have that the weighed projection $\mathcal{O}_i$ (2.33) defined as:

$$\mathcal{O}_i \stackrel{\text{def}}{=} Y_f /_{U_f} W_p = \Gamma_i \widehat{X}_f , \quad (2.35)$$

equals the product between the extended observability matrix Γ_i and the nonsteady-state Kalman filter state estimate $\widehat{X}_f$ of the state sequence X_f, as defined in Theorem 2.4.1, with $\widehat{X}_p = X_p /_{U_f} W_p$ as the initial state sequence (See also Figure 2.5).

Proof: The proof of this theorem can be found in [119], [123]. A less involved proof is given hereafter. With (2.28), the oblique projection (2.35) can be rewritten as:

$$\begin{aligned} Y_f/_{U_f} W_p &= \Gamma_i X_f/_{U_f} W_p + H_i^d \underbrace{U_f/_{U_f} W_p}_{=0} + H_i^s \underbrace{E_f/_{U_f} W_p}_{=0} \\ &= \Gamma_i \widehat{X}_f . \end{aligned}$$

The last two projections of the left hand side are zero if the first three assumptions of the present theorem are satisfied.

It now remains to prove that $\widehat{X}_f$ is a nonsteady-state Kalman filter estimate of X_f with $X_p/_{U_f} W_p$ as the initial state and the columns of W_p as the inputs of the filter (See also Figure 2.5). If the noise e_k is not identically zero, we can substitute E_p into X_f using Equations (2.27) and (2.29). We then get the following expression for the steady state Kalman filter state sequence:

$$X_f = (A^i - \Delta_i^s H_i^{-s}) X_p + \left(\Delta_i^s H_i^{-s} \mid \Delta_i^d - \Delta_i^s H_i^{-s} H_i^d \right) \begin{pmatrix} Y_p \\ U_p \end{pmatrix} .$$

After projecting X_f along the row space of U_f into the row space of W_p we have:

$$\begin{aligned} X_f/_{U_f} W_p &= (A^i - \Delta_i^s H_i^{-s}) X_p/_{U_f} W_p + \left(\Delta_i^s H_i^{-s} \mid \Delta_i^d - \Delta_i^s H_i^{-s} H_i^d \right) W_p/_{U_f} W_p \\ &= (A^i - \Delta_i^s H_i^{-s}) X_p/_{U_f} W_p + \left(\Delta_i^s H_i^{-s} \mid \Delta_i^d - \Delta_i^s H_i^{-s} H_i^d \right) \begin{pmatrix} Y_p \\ U_p \end{pmatrix} . \end{aligned}$$

With Theorem 2.4.1 it is easy to see that every element of $\widehat{X}_f$ is the solution to a bank of Kalman filters. Indeed, every element of $\widehat{X}_f$ is the nonsteady-state solution of the Kalman filter equations (2.34) or (2.9)-(2.13) with the corresponding column of $X_p/_{U_f} W_p$ and W_p as the initial state estimate and the filter inputs respectively. ∎

As illustrated in Figure 2.5, the estimated state sequence $\widehat{X}_f$ is generated by a bank of Kalman filters working in parallel on each of the columns of the matrix W_p. Each element in the sequence $\widehat{X}_f$ is the nonsteady-state Kalman filter estimate of the state, where the initial state estimate is the corresponding element in $\widehat{X}_p$ and the available input/output measurements the elements of corresponding column of W_p [123].

2.4.2 Second step: SV-decomposition

In the previous step we have seen how the product between the extended observability matrix Γ_i and a nonsteady-state Kalman filter estimate $\widehat{X}_f$ of the

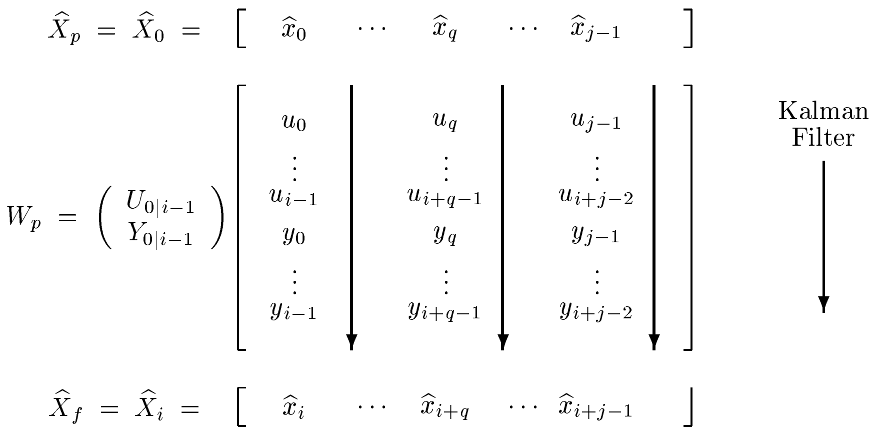


Figure 2.5: **The above figure illustrates how the state sequence $\widehat{X}_f$, estimated from the oblique projection (2.35), can be interpreted as a nonsteady-state Kalman filter estimate of the state sequence X_f. If the system matrices A, B, C, D, Q, R, S were known, the state $\widehat{x}_{i+q}$ could be determined from a Kalman filter as follows: Start the filter at time q, with an initial state estimate $\widehat{x}_q$. Then iterate the Kalman filter over i time steps (the vertical arrow down). The Kalman filter will then return a state estimate $\widehat{x}_{i+q}$. This procedure could be repeated for each of the j columns, and thus we speak about a *bank* of Kalman filters. The major observation in subspace algorithms is that the system matrices A, B, C, D, Q, R, S do not have to be known to determine the state sequence $\widehat{X}_i$. It can be determined directly from input/output data.**

state sequence X_f can be calculated with an oblique projection. From (2.33) it can be seen that this oblique projection can also be written as:

$$Y_f /_{U_f} W_p = L_w W_p .$$

In the second step of subspace system identification this term is split up into its components i.e. Γ_i and $\widehat{X}_f$. This is perfectly possible since, if the number of columns in the data block-Hankel matrices j is infinite ($j \to \infty$), L_w is a rank-n matrix where n is the order of the system. In practice however, due to the noise and the finite amount of data, L_w will not be exactly rank-deficient. A singular value decomposition is then used to approximate it by a rank-n matrix. For reasons that will become clear in Section 2.6, L_w is first weighed on the left and on the right by weighting matrices W_1 and W_2 before calculating the SVD:

$$\begin{aligned} W_1 L_w W_2 &= \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \\ &\approx U_1 S_1 V_1^T \end{aligned} \tag{2.36}$$

In order to preserve the rank of L_w, the weighting matrices W_1 and W_2 should be of full rank. The order n of the system can be estimated by inspecting the number of dominant singular values S_1. The observability matrix and the state sequence of the system can now be directly determined from the above SVD:

$$\Gamma_i = W_1^{-1} U_1 S_1^{1/2}, \tag{2.37}$$

$$\widehat{X}_f = S_1^{1/2} V_1^T W_2^{-1} W_p. \tag{2.38}$$

2.4.3 Third step: recovering the state-space matrices

Up till now we have seen how a specific nonsteady-state Kalman filter estimate $\widehat{X}_f$ of the state sequence X_f and the extended observability matrix Γ_i can be retrieved from the singular value decomposition (2.35). The only problem that remains to be solved now is that of finding the state-space matrices A, B, C, D, Q, R, S from Γ_i and/or $\widehat{X}_f$. This is the main point where the different subspace system identification algorithms, available in the literature, diverge. Due to the significant diversity of all these algorithms, both from a theoretical and the implementational point of view, it is difficult, if not impossible, to classify them all in a simple way.

In what follows, we discuss two main classes of subspace identification algorithms. The first class uses the state estimates $\widehat{X}_f$ to find the state-space model, whereas the second class uses the extended observability matrix Γ_i. Although in practice most algorithms use in fact both the states $\widehat{X}_f$ and the extended observability matrix Γ_i, one could say that N4SID [119] and CVA [90] are more inspired by the first class of algorithms, whereas MOESP [127], IV-4SID and *basic*-4SID [129], [64], [101] typically use the extended observability matrix.

Algorithms using the estimate $\widehat{X}_f$ of the state sequence

In this section we examine the class of algorithms that use the estimated state sequence $\widehat{X}_f$ to find the state-space matrices A, B, C, D, Q, R, S. The problem is split up in two parts:

System model A, B, C, D: Once the states are known, it is an easy problem to find the state-space matrices A, B, C, D by solving the following linear least squares problem:

$$\begin{pmatrix} \widehat{X}_{i+1} \\ Y_{i|i} \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} \widehat{X}_i \\ U_{i|i} \end{pmatrix} + \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}, \qquad (2.39)$$

where ρ_w and ρ_v are residual matrices. We have seen in the first two steps of subspace identification that the state sequences $\widehat{X}_i$ and $\widehat{X}_{i+1}$ can be obtained from the SV-decomposition of the following projections:

$$Y_{i|2i-1}/_{U_{i|2i-1}} W_{0|i-1} = \Gamma_i \widehat{X}_i, \qquad (2.40)$$

$$Y_{i+1|2i-1}/_{U_{i+1|2i-1}} W_{0|i} = \Gamma_{i-1} \widehat{X}_{i+1}. \qquad (2.41)$$

Care should be taken about the fact that both state sequences $\widehat{X}_i$ and $\widehat{X}_{i+1}$ should be calculated in the same state-space basis. By simply calculating $\widehat{X}_i$ and $\widehat{X}_{i+1}$ from the SVD of the above two projections, this is not necessarily the case. However, $\widehat{X}_i$ and $\widehat{X}_{i+1}$ can be forced into the same state-space basis with the following "trick". First calculate Γ_i and $\widehat{X}_i$ from the first projection (2.40) as in (2.37)-(2.38). Then calculate Γ_{i-1} from Γ_i as $\Gamma_{i-1} = \Gamma_i(1 : end - l, :)$. Finally calculate $\widehat{X}_{i+1}$ as:

$$\widehat{X}_{i+1} = \Gamma_{i-1}^{\dagger} Y_{i+1|2i-1}/_{U_{i+1|2i-1}} W_{0|i}.$$

Although $\widehat{X}_i$ and $\widehat{X}_{i+1}$ are now in the same state-space basis, it can be shown that, if $\widehat{X}_i$ and $\widehat{X}_{i+1}$ are calculated as above, the least squares problem (2.39) will lead to biased estimates of A, B, C, D. The reason for this is that (2.39) will only be consistent if $\widehat{X}_i$ and $\widehat{X}_{i+1}$ are *consecutive* estimates of the *same* Kalman filter. This means that the corresponding elements of $\widehat{X}_i$ and $\widehat{X}_{i+1}$ should be the consecutive solutions of the Kalman filter equations (2.9)-(2.10) i.e. with the same initial state estimate and the same inputs (See Figure 2.6). The problem with the sequences $\widehat{X}_i$ and $\widehat{X}_{i+1}$ is that, although they have the same input sequences, they have different initial state estimates. Indeed, their initial state sequences are respectively $\widehat{X}_0^1$ and $\widehat{X}_0^2$:

$$Y_{i|2i-1}/_{U_{i|2i-1}} W_{0|i-1} \longrightarrow \widehat{X}_0^1 = X_0/_{U_{i|2i-1}} W_{0|i-1},$$

$$Y_{i+1|2i-1}/_{U_{i+1|2i-1}} W_{0|i} \longrightarrow \widehat{X}_0^2 = X_0/_{U_{i+1|2i-1}} W_{0|i}$$

from which follows that $\widehat{X}_0^1$ and $\widehat{X}_0^2$ are different. As a consequence, the set of equations (2.39) is not consistent which means that the estimates of the system matrices A, B, C, D are slightly biased. We refer to [123] where more involved algorithms are presented that provide consistent estimates of A, B, C, D. These algorithms tackle the origin of the bias, i.e. the difference in initial state for the Kalman filter sequences $\widehat{X}_i$ and $\widehat{X}_{i+1}$, to find an unbiased version of the algorithm presented in this chapter.

$$\begin{array}{ccc}
\text{Initial state estimate} & \widehat{x}_0^1 & \widehat{x}_0^2 \\
\downarrow & \downarrow & \downarrow \\
\begin{array}{c}\text{Kalman} \\ \text{filter}\end{array} & \begin{bmatrix} u_0 \\ y_0 \\ \vdots \\ u_{i-1} \\ y_{i-1} \end{bmatrix} & \begin{bmatrix} u_0 \\ y_0 \\ \vdots \\ u_{i-1} \\ y_{i-1} \\ u_i \\ y_i \end{bmatrix} \\
\downarrow & \downarrow & \downarrow \\
\text{Kalman filter estimate} & \widehat{x}_{i-1} & \widehat{x}_i
\end{array}$$

Figure 2.6: **In a similar way as in Figure 2.5, it is shown how the elements of $\widehat{X}_i$ (left column) and of $\widehat{X}_{i+1}$ (right column) can be interpreted as consecutive solutions of different nonsteady-state Kalman filter i.e. with different initial state estimates $\widehat{x}_0^1$ and $\widehat{x}_0^2$.**

Noise model Q, R, S: The noise covariances Q, R and S can be estimated from the residuals ρ_w and ρ_v of the least squares problem (2.39):

$$\begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}_i = \frac{1}{j}[\begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}\begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}^T] \geq 0$$

where the index i denotes a bias induced for finite i, which disappears as $i \to \infty$. In that case, the nonsteady-state Kalman filter sequences $\widehat{X}_i$ and $\widehat{X}_{i+1}$ will have turned into steady-state Kalman sequences. For more details we refer to [123].

Figure 2.7 presents the basic steps of a subspace identification algorithm, that calculates the system parameters A, B, C, D, Q, R, S based on the estimated state sequence $\widehat{X}_f$. Since the aim of this chapter is only to give an overview of existing methods, we restricted ourselves to this simple, but slightly biased version of more sophisticated algorithms [123].

Algorithms using the extended observability matrix Γ_i

Now we will focus on the class of subspace system identification algorithms that use the extended observability matrix Γ_i to find the state-space matrices A, B, C, D, Q, R and S.

System model A, B, C, D: The matrices A, B, C, D are determined in two

Subspace algorithm using the states

1. Calculate the oblique projections with an **LQ-decomposition**:

$$\mathcal{O}_i \overset{\mathrm{LQ}}{=} Y_{i|2i-1}/_{U_{i|2i-1}} W_{0|i-1} = L_w W_{0|i-1},$$

$$\mathcal{O}_{i+1} \overset{\mathrm{LQ}}{=} Y_{i+1|2i-1}/_{U_{i+1|2i-1}} W_{0|i}$$

and determine L_w.

2. Calculate the weighed **SVD**:

$$W_1 L_w W_2 \overset{\mathrm{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

and

- determine an estimate of the order n of the system by inspecting the number of dominant singular values S_1.
- determine Γ_i and Γ_{i-1} as:

$$\Gamma_i = W_1^{-1} U_1 S_1^{1/2} \quad , \quad \Gamma_{i-1} = \Gamma_i(1 : end - l, :).$$

- determine the state sequences:

$$\widehat{X}_i = S_1^{1/2} V_1^T W_2^{-1} W_p \quad , \quad \widehat{X}_{i+1} = \Gamma_{i-1}^{\dagger} \mathcal{O}_{i+1}.$$

3. Calculate the estimates of the system matrices A, B, C, D and the noise covariance matrices Q, R, S:

System model:

$$\begin{pmatrix} \widehat{X}_{i+1} \\ Y_{i|i} \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} \widehat{X}_i \\ U_{i|i} \end{pmatrix} + \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}.$$

Noise model:

$$\begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}_i = \frac{1}{j}[\begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix} \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}^T].$$

Figure 2.7: **A schematic overview of a simple state-based subspace system identification algorithm. Note the "symmetry" in this algorithm: the matrices A, B, C, D as well as the matrices Q, R, S are all determined at the same time, which gives a certain elegance to this algorithm. However, it computes asymptotically slightly biased solutions. More sophisticated versions of this algorithm exist [123] that give unbiased estimates of the system matrices. Due to space limitations and for simplicity's sake, we restricted ourselves here to this algorithm.**

separate steps: first A and C are calculated from Γ_i after which B and D can be found from a linear least squares problem.

Determination of A and C: The matrices A and C can be easily determined from the extended observability matrix using the shift-invariance property [85]:

$$\Gamma_i(1:end-l,:)A = \Gamma_i(l+1:end,:)$$

From this equation, A and C can be found as:

$$\begin{aligned} A &= \Gamma_i(1:end-l,:)^\dagger \Gamma_i(l+1:end,:), \\ C &= \Gamma_i(1:l,:). \end{aligned}$$

Determination of B and D: We will only sketch one possible solution to find B and D. Since we assumed that the innovations e_k are not correlated with the inputs u_k and since the input was assumed to be persistently exciting of order $2i$, we can see from the matrix input/output equation (2.28) that:

$$\Gamma_i^\perp Y_f U_f^\dagger = \Gamma_i^\perp H_i^d,$$

which can also be written as:

$$\begin{aligned} \begin{pmatrix} \mathcal{M}_1 & \mathcal{M}_2 & \dots & \mathcal{M}_i \end{pmatrix} &= \begin{pmatrix} \mathcal{L}_1 & \mathcal{L}_2 & \dots & \mathcal{L}_i \end{pmatrix} \\ & \begin{pmatrix} D & 0 & 0 & \dots & 0 \\ CB & D & 0 & \dots & 0 \\ CAB & CB & D & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ CA^{i-2}B & CA^{i-3}B & CA^{i-4}B & \dots & D \end{pmatrix}, \end{aligned}$$

where $\mathcal{M}_k \in \mathbb{R}^{(li-n)\times m}$ and $\mathcal{L}_k \in \mathbb{R}^{(li-n)\times l}$. This can be rewritten as:

$$\begin{pmatrix} \mathcal{M}_1 \\ \mathcal{M}_2 \\ \vdots \\ \mathcal{M}_i \end{pmatrix} = \begin{pmatrix} \mathcal{L}_1 & \mathcal{L}_2 & \dots & \mathcal{L}_{i-1} & \mathcal{L}_i \\ \mathcal{L}_2 & \mathcal{L}_3 & \dots & \mathcal{L}_i & 0 \\ \mathcal{L}_3 & \mathcal{L}_4 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \mathcal{L}_i & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} I_l & 0 \\ 0 & \Gamma_i(1:end-l,:) \end{pmatrix} \begin{pmatrix} D \\ B \end{pmatrix}, \quad (2.42)$$

which is a set of linear equations in the unknowns B and D and is typically over determined. It could for instance be solved in a least squares sense.

Noise model Q, R, S: Since the system model A, B, C, D is known now, the stochastic part of the state and the output can be easily found. The noise covariances Q, R and S can then be calculated as in step 3 of the algorithm in Figure 2.7.

Figure 2.8 presents an algorithm that is based on the estimate of the observability matrix to extract the state-space matrices A, B, C, D, Q, R, S.

2.5 Implementation: LQ-decomposition

In this section we analyze into some more detail how the orthogonal and oblique projections (2.31)-(2.33) of the previous sections can be calculated in a numerically robust and efficient way.

Orthogonal and oblique projections can in fact be interpreted as a linear least squares problem. The problem of projecting the row space of Y_f into the joint row space of W_p and U_f consists in solving the following linear least squares problem for L_w and L_u:

$$\min_{L_w, L_u} ||Y_f - \begin{pmatrix} L_w & L_u \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix} ||_F^2. \tag{2.43}$$

For the orthogonal and the oblique projections we then have:

$$\begin{aligned} Y_f / \begin{pmatrix} W_p \\ U_f \end{pmatrix} &= L_w W_p + L_u U_f, & (2.44) \\ Y_f /_{U_f} W_p &= L_w W_p, & (2.45) \\ Y_f /_{W_p} U_f &= L_u U_f \end{aligned}$$

which can be calculated in a very elegant way using the following LQ-decomposition:

$$\begin{pmatrix} U_f \\ W_p \\ Y_f \end{pmatrix} \stackrel{\text{LQ}}{=} \begin{pmatrix} L_{cc} & 0 & 0 \\ L_{bc} & L_{bb} & 0 \\ L_{ac} & L_{ab} & L_{aa} \end{pmatrix} \begin{pmatrix} Q_c^T \\ Q_b^T \\ Q_a^T \end{pmatrix}. \tag{2.46}$$

It follows that Y_f, W_p and U_f can be expressed as:

$$\begin{aligned} U_f &= L_{cc} Q_c^T, \\ W_p &= L_{bc} Q_c^T + L_{bb} Q_b^T, \\ Y_f &= L_{ac} Q_c^T + L_{ab} Q_b^T + L_{aa} Q_a^T. \end{aligned}$$

The orthogonal and oblique projections (2.44)-(2.45) can now be derived:

Subspace algorithm using the extended observability matrix:

1. Calculate the projection with an **LQ-decomposition** and determine L_w:

$$\mathcal{O}_i \overset{\mathrm{LQ}}{=} Y_f /_{U_f} W_p = L_w W_p .$$

2. Calculate the weighed **SVD**:

$$W_1 L_w W_2 \overset{\mathrm{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

   and

   - determine an estimate of the order n of the system by inspecting the number of dominant singular values S_1.
   - determine Γ_i:

$$\Gamma_i = W_1^{-1} U_1 S_1^{1/2} .$$

3. Calculate the estimates of the system matrices A, B, C, D and the noise covariance matrices Q, R, S:

   System model:

   - $$A = \Gamma_i(1 : end - l, :)^\dagger \Gamma_i(l + 1 : end, :), \quad C = \Gamma_i(1 : l, :).$$
   - Solve the linear set of equations (2.42) for B and D:

$$\begin{pmatrix} \mathcal{M}_1 \\ \mathcal{M}_2 \\ \vdots \\ \mathcal{M}_i \end{pmatrix} = \begin{pmatrix} \mathcal{L}_1 & \mathcal{L}_2 & \dots & \mathcal{L}_{i-1} & \mathcal{L}_i \\ \mathcal{L}_2 & \mathcal{L}_3 & \dots & \mathcal{L}_i & 0 \\ \mathcal{L}_3 & \mathcal{L}_4 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \mathcal{L}_i & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} I_l & 0 \\ 0 & \Gamma_i(1 : end - l, :) \end{pmatrix} \begin{pmatrix} D \\ B \end{pmatrix} ,$$

   Noise model: See Step 3 of the algorithm in Figure 2.7.

Figure 2.8: **Typical subspace identification algorithm using the extended observability matrix Γ_i. The first two steps are basically the same as for the state-based algorithm of Figure 2.7. The main difference lies in the way A, B, C, D are determined in the third step.**

Orthogonal projection $Y_f/\binom{W_p}{U_f}$:

$$\begin{aligned} Y_f/\begin{pmatrix} W_p \\ U_f \end{pmatrix} &= Y_f \begin{pmatrix} W_p \\ U_f \end{pmatrix}^{\dagger} \begin{pmatrix} W_p \\ U_f \end{pmatrix} \\ &= Y_f \begin{pmatrix} W_p \\ U_f \end{pmatrix}^{T} \left(\begin{pmatrix} W_p \\ U_f \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix}^{T} \right)^{-1} \begin{pmatrix} W_p \\ U_f \end{pmatrix} \\ &= \underbrace{\begin{pmatrix} L_{ac} & L_{ab} \end{pmatrix} \begin{pmatrix} L_{cc} & 0 \\ L_{bc} & L_{bb} \end{pmatrix}^{\dagger}}_{\stackrel{\text{def}}{=} L} \begin{pmatrix} W_p \\ U_f \end{pmatrix}. \end{aligned}$$

Oblique projection $Y_f/_{U_f}\, W_p$:

It follows from the Definition 2.3.2 that the oblique projection:

$$Y_f/_{U_f}\, W_p = L(:, 1 : (m+l)i) W_p.$$

Since $Y_f/\binom{W_p}{U_f} = L_w W_p + L_u U_f$ we have that L_w and L_u can be found as:

$$\begin{aligned} L_w &= L(:, 1 : i(m+l)), \\ L_u &= L(:, end - im + 1 : end). \end{aligned}$$

We finish this section with the following remark:

Remark 2.5.1

- It can be proven [123] that, if the number of block rows i in the past data block-Hankel matrix W_p is infinite, the state estimate $\widehat{X}_f$ obtained with the resulting projection is the steady-state Kalman filter estimate X_f:

  $$Y_f/_{U_f}\, W_p = \Gamma_i X_f.$$

  Under such circumstances we also have that the block-Toeplitz matrices (2.17)-(2.18), containing the impulse response coefficients of the system to the deterministic input u_k and the stochastic input e_k respectively, are directly linked to the above defined LQ-decomposition (2.46):

  $$\begin{aligned} H_i^d &= L_u, \\ H_i^s &= L_{aa}. \end{aligned}$$

- If the inputs u_k are zero-mean white it can be shown [123] that $H_i^d = L_u$.

2.6 Unifying framework

Until now we did not yet discuss the problem of finding the weighting matrices W_1 and W_2 in the subspace algorithms of Figures 2.7 and 2.8. W_1 and W_2 can in fact be considered as tuning parameters:

- When doing system identification, it is sometimes important to have a good model in a certain frequency region. For instance if the DC-gain of the model is important, one will typically use a low pass filter on the data in order to increase the importance of the low frequencies. In the case of subspace identification, W_1 and W_2 can be given a similar interpretation [120], [123]. When chosen appropriately they can increase the importance of certain frequency regions.
- Another nice feature of W_1 and W_2 is that they influence the state-space basis in which the final state-space model is calculated. It was shown [97], [123] that, when W_1 and W_2 are well chosen, it is possible to find a model that is balanced which is of course interesting for model reduction purposes.

In Section 2.4.3 we have already mentioned that there are different types of subspace system identification algorithms. We divided them into two main classes namely those that use the extended observability matrix (IV-4SID and MOESP) and those that use a Kalman filter state estimate of the system (N4SID and CVA) to find the system matrices. Besides this classification, subspace methods can also be classified according to the choice of W_1 and W_2. Indeed, by choosing the appropriate weighting matrices W_1 and W_2, most subspace algorithms for LTI systems can be interpreted in the above framework including N4SID [119], MOESP [126], CVA [90], IV-4SID [129]. We present in Table 2.1 the acronyms and references of some algorithms and the weights to be plugged into (2.36). We refer to [75], [121], [120], [123] for proofs and details.

2.7 Comparison of different subspace identification algorithms

The weights given in Table 2.1 are not the only implementational difference between the different subspace identification algorithms. The main point where the algorithms diverge is the third step in the algorithm (See Section 2.4.3), i.e. the point where the state-space matrices are effectively calculated. In the present section it is our intention to make a direct comparison between three popular subspace system identification algorithms (CVA, N4SID and MOESP).

Acronym	W_1	W_2
N4SID	I_{li}	I_j
CVA	$[(Y_f/U_f^{\perp})(Y_f/U_f^{\perp})^T]^{-1/2}$	$I - U_f^{\perp} U_f$
PO-MOESP	I_{li}	$I - U_f^{\perp} U_f$
IV-4SID	I_{li}	Φ

Table 2.1: **This table interprets four different subspace identification algorithms in a unifying framework. All these algorithms calculate a weighed singular value decomposition (2.36). The left and right weighting matrices used in these algorithms are W_1 and W_2 respectively. The first two algorithms mainly use the state estimates $\widehat{X}_f$ to find the system matrices while the last two algorithms are based on the extended observability matrix Γ_i. The matrix Φ of the IV-4SID method is a matrix containing the instrumental variables.**

The comparison is made by applying the different subspace algorithms on 15 publicly available practical data sets. As we simply used the Matlab-code as provided by the authors of the algorithms, it is **not** our purpose to analyze the implementational differences between those algorithms but to test and compare the performance of the algorithms, according to certain criteria that will be explained further down. The three subspace identification algorithms we used are:

CVA: **C**anonical **V**ariate **A**nalysis [103], [90][1].

N4SID: **N**umerical algorithm for **S**ubspace **S**tate **S**pace **S**ystem **ID**entification. The algorithm we used is the `subid.m` Matlab function that comes with [123].

MOESP: **M**ultivariable **O**utput **E**rror **S**tate s**P**ace. The Matlab functions that have been used here are `dordpo.m`, `dmodpo.m` and `dac2bd.m` of the SMI-toolbox [67].

The different algorithms have been implemented under Matlab 5.1 on a Sun Ultra Sparc 5 workstation (266 MHz) as an m-file of the following format:

$$[A, B, C, D, K] \quad = \quad \text{subspace}(u_{id}, y_{id}, i, n)$$

where

- "subspace" is the name of the subspace identification algorithm (`CVA`, `N4SID` or `MOESP`).

[1] The Matlab-code for this algorithm was provided by Dr. Dietmar Bauer from the Technische Universität Wien.

- u_{id} and y_{id} are the inputs and the outputs of the identification data set.
- i is the number of block rows in the past and the future data block-Hankel matrices (2.19)-(2.20). For simplicity, the number of block rows in the past and future matrices is assumed to be the same.
- n is the estimated order of the system and should be such that $n < il$.
- the outputs of the function are the state-space matrices A, B, C, D of the system and the Kalman gain K.

In order to increase the accessibility and the reproducibility of the results presented here, all the data and the Matlab-code that have been used are made publicly available on the DAISY[2] internet site [30]:

`http://www.esat.kuleuven.ac.be/sista/daisy`

Every data set on this site is labeled by a 5 digit number. The list of data sets we used is:

1. **ball & beam -** Data from the ball and beam practicum at the ESAT/SISTA laboratory of the Katholieke Universiteit Leuven [96-004].
2. **cd player arm -** Data from the mechanical construction of a CD player arm [96-007].
3. **distillation -** Simulation data from an ethan-ethylene distillation column [96-001].
4. **dryer 1 -** Laboratory setup acting like a hair dryer [96-006].
5. **dryer 2 -** Data from an industrial dryer (by Cambridge Control Ltd) [96-016].
6. **evaporator -** A four-stage evaporator to reduce the water content of a product, for example milk [96-010].
7. **heat exchanger -** A liquid-saturated steam heat exchanger [97-002].
8. **flexible structure -** Experiment on a steel subframe flexible structure performed at LMS-International, Leuven-Belgium [96-013].
9. **glass furnace -** Data from a glass furnace (Philips) [96-002].
10. **pH data -** Simulation data of a pH neutralization process in a constant volume stirring tank [98-002].

[2]**DAISY** stands for **DA**tabase for the **I**dentifiction of **SY**stems

11. **powerplant -** Data from a power plant (Pont-sur-Sambre - France) of 120 MW [96-003].
12. **robot arm -** Data from a flexible robot arm [96-009].
13. **steam generator -** Steam generator at Abbott Power Plant in Champaign IL [98-003].
14. **thermic wall -** Heat flow density through a two layer wall [96-011].
15. **winding process -** Test setup of an industrial winding process [97-003].

For more details on the different data sets we refer to the DAISY web site. A data set is always split up into an identification part and a validation part. With the identification set the model parameters are calculated while the validation set is used to find the prediction/simulation error. The parameters of the identification of every data set are given in Table 2.2. m and l are the number of inputs and outputs respectively. id is the number of data points used for the identification whereas val is the length of the validation data set.

We considered the following criteria to be important for the quality of a system identification:

Prediction error:

$$\epsilon^p \stackrel{\text{def}}{=} \frac{1}{l}\sum_{q=1}^{l}\sqrt{\frac{\sum_{k=1}^{val}(y_{k,q}-y_{k,q}^p)^2}{\sum_{k=1}^{val}y_{k,q}^2}}$$

with $y_{k,q}$ the q-th output at time step k and y_k^p the one-step-ahead predicted output.

Simulation error:

$$\epsilon^s \stackrel{\text{def}}{=} \frac{1}{l}\sum_{q=1}^{l}\sqrt{\frac{\sum_{k=1}^{val}(y_{k,q}-y_{k,q}^s)^2}{\sum_{k=1}^{val}y_{k,q}^2}}$$

with y_k^s the simulated output.

The above two criteria are calculated for a wide range of the identification parameters i and n. For each data set we calculated the simulation and the prediction error for different values of i and n. These errors have been represented as a function of i and n in a three dimensional plot which, due to space

# data set	m	l	id	val
1.	1	1	1000	0
2.	2	2	1500	548
3.	1	2	3000	979
4.	1	1	750	250
5.	3	3	600	267
6.	3	3	5000	1305
7.	1	1	3000	1000
8.	2	1	6000	2523
9.	3	6	1000	247
10.	2	1	1500	501
11.	5	3	200	0
12.	1	1	800	224
13.	4	4	7000	2600
14.	2	1	1000	680
15.	5	2	1500	1000

Table 2.2: **Parameters of the different data sets.** m **is the number of inputs,** l **the number of outputs,** id **the number of data points used for the identification and** val **the number of data points used for the validation of the obtained model. If** val **is zero, the data set was too short to be split up into an identification and a validation part. In that case, the identification data set was used for the validation of the model.**

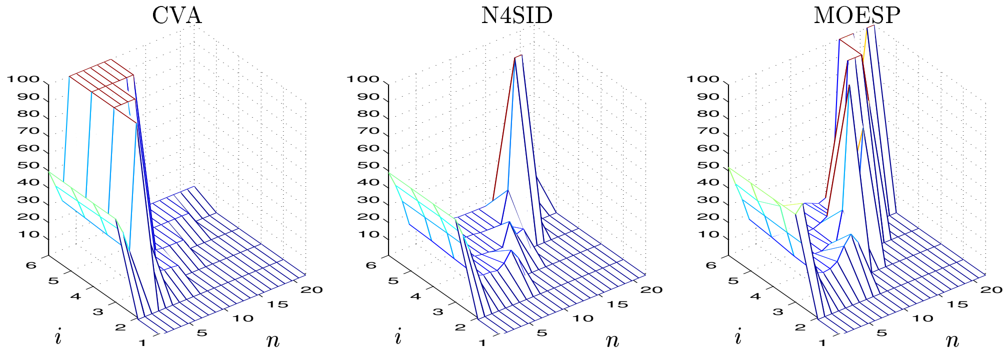


Figure 2.9: **Prediction error of data set 13 as a function of** i **and** n**. For** $il \geq n$ **no value has been calculated. The N4SID and MOESP algorithms perform worse than CVA as** i **is chosen close to the constraint** $il > n$**. This was also observed for other data sets.**

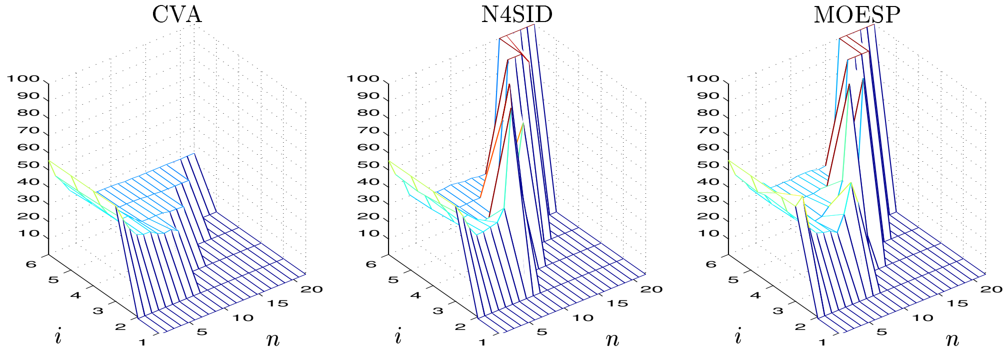


Figure 2.10: **Simulation error of data set 13 as a function of i and n. Same remark as in Figure 2.9.**

limitations, we only give for data set number 13 (Figures 2.9 and 2.10). The interested reader can find similar results for the other data sets in [43].

Tables 2.3 and 2.4 give the overall results for the 15 data sets. They present the minimum value of the prediction and the simulation errors over i and n. For each data set the value that gives the smallest error over the three algorithms is indicated in bold-face. The errors that are larger than 100 have been clipped at 100.

We remind the reader that the above results might depend strongly on the way the algorithm is implemented. Though very interesting, we did not try to link the results to the way the algorithm was implemented by the different authors. Although the performance of the methods is very comparable, CVA and N4SID tend to have smaller prediction errors whereas MOESP has smaller simulation errors. The CVA algorithm also seems to be less sensitive to the choice of the system order and the number of block rows in the data matrices.

In this chapter we have reviewed the most important results, for this thesis, of subspace identification. We have seen that subspace algorithms basically consist of an LQ-decomposition, an SV-decomposition and a third step where the state-space matrices are extracted. Further we have compared three different subspace algorithms, available in the literature, by applying them on 15 different real-life datasets.

Data set	CVA	N4SID	MOESP
1	2.4198	**2.3220**	2.4186
2	6.8151	**6.5376**	8.9553
3	**48.7454**	48.7784	49.1773
4	4.8265	4.7682	**4.7666**
5	11.5455	**10.7647**	12.6140
6	**20.0718**	20.1193	20.6335
7	**23.8987**	24.1221	24.2227
8	**11.3536**	12.2794	20.2570
9	17.0438	**13.6495**	14.4347
10	52.3222	**31.7654**	46.3633
11	6.8093	**4.6228**	5.0161
12	0.9642	0.1170	**0.0998**
13	**6.3111**	6.3273	9.2190
14	**14.6819**	14.8387	14.8210
15	**18.2243**	19.7667	19.4947

Table 2.3: **The minimum value of the prediction error over the different values of i and n for each data set. CVA and N4SID perform better than MOESP.**

Data set	CVA	N4SID	MOESP
1	100.0000	74.9221	**72.3515**
2	19.6774	23.5653	**18.0015**
3	55.6503	55.8385	**55.6116**
4	**7.5086**	7.8874	7.9014
5	**49.2288**	49.4671	50.3791
6	43.0268	42.9225	**37.4237**
7	32.5908	**32.5660**	32.8408
8	82.7149	77.6466	**75.9364**
9	43.2657	**38.0679**	41.6424
10	100.0000	100.0000	100.0000
11	16.3818	12.8543	**9.9283**
12	5.7773	**3.5258**	3.9485
13	24.8750	**24.4981**	28.6154
14	**14.8218**	14.9181	14.8399
15	**23.3648**	24.8119	24.3589

Table 2.4: **The minimum value of the simulation error over the different values of i and n for each data set. The errors larger than 100 have been clipped at 100. MOESP performs better than CVA and N4SID.**

Chapter 3

Subspace identification of bilinear systems with white inputs

In the present chapter, which is structured in a similar way as the previous chapter, we introduce subspace identification theory for bilinear systems, where we assume that the inputs are white and independent of the noise. We start by giving a formal definition of the problem. Further we analyze how, in a similar way as for linear systems, a bilinear system with white inputs can be split up into the sum of a stochastic and a deterministic part. Since subspace identification is closely related to realization theory, we continue by recalling stochastic and deterministic realization theory of bilinear systems. Also the link with Kalman filtering is made and the forward innovation form of a bilinear system is defined. The generalized data block-Hankel matrices used in bilinear subspace identification are defined and finally a subspace algorithm for the identification of bilinear systems with white inputs is presented.

3.1 Introduction – problem description

In this chapter we will consider time-invariant bilinear systems of the form:

$$\begin{aligned} x_{k+1} &= Ax_k + Nu_k \otimes x_k + Bu_k + w_k, \\ y_k &= Cx_k + Du_k + v_k, \end{aligned}$$

where the measurement noise $v_k \in \mathbb{R}^l$ and the process noise $w_k \in \mathbb{R}^n$ are zero mean, white, stationary sequences.

Throughout the whole chapter we make the following important assumption:

Assumption 3.1.1 *The input $u_k \in \mathbb{R}^m$ is assumed to be white, observed and independent of the measurement noise v_k and the process noise w_k.*

This assumption might at first seem somewhat restrictive. Nevertheless, the practical use of it is straightforward since in many cases, when one tries to identify a system, a well determined input signal is chosen. In order to guarantee persistency of excitation, this signal is often white. Pseudo Random Binary Sequences (PRBS) for instance are widely used for identification purposes.

The mutual covariance matrices of the inputs u_k, the measurement noise v_k and the process noise w_k are defined as:

$$\mathbf{E}[\begin{pmatrix} u_p \\ w_p \\ v_p \end{pmatrix}\begin{pmatrix} u_q \\ w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} I & 0 & 0 \\ 0 & Q & S \\ 0 & S^T & R \end{pmatrix}\delta_{pq} \geq 0.$$

The problem we will be concerned with in this chapter is defined in Figure 3.1.

Problem formulation

Given *a large number of measurements of the input u_k and the output y_k generated by the unknown bilinear system:*

$$\begin{aligned} x_{k+1} &= Ax_k + Nu_k \otimes x_k + Bu_k + w_k, && (3.1) \\ y_k &= Cx_k + Du_k + v_k && (3.2) \end{aligned}$$

with

$$\mathbf{E}[\begin{pmatrix} u_p \\ w_p \\ v_p \end{pmatrix}\begin{pmatrix} u_q \\ w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} I & 0 & 0 \\ 0 & Q & S \\ 0 & S^T & R \end{pmatrix}\delta_{pq} \geq 0, \qquad (3.3)$$

determine *an estimate of the system order n, the system matrices A, B, C, D, N up to within a similarity transformation and the noise covariance matrices Q, R, S.*

Figure 3.1: **The bilinear subspace system identification problem for bilinear systems, where the inputs are assumed to be white and independent from the noise.**

In the present and the next chapters we will sometimes use variables that are defined for both linear and bilinear systems e.g. the (generalized) observability matrix, the (generalized) data Hankel matrices, ... In order to avoid confusion in notation between the bilinear and the linear variables, the latter ones will always be indexed with the suffix l standing for "linear". For instance Γ_i^l will be the linear observability matrix and Y_f^l the future output data block-Hankel matrix used in linear subspace identification whereas Γ_i will be the generalized observability matrix and Y_f the generalized future output data Hankel matrix used in bilinear subspace identification.

3.2 Stochastic properties of bilinear systems

In a similar way as for linear systems we define a way to split up the system (3.1)-(3.3) into the sum of a stochastic and deterministic system. Here however, it is no longer possible to decouple both subsystems because the stochastic subsystem also contains the input u_k, which is a deterministic variable. Further, deterministic and stochastic realization theory of bilinear systems is reviewed. Finally, the innovation representation of a bilinear system is defined and the Kalman filter equations for bilinear systems (3.1)-(3.3) are derived.

3.2.1 Deterministic and stochastic subsystem

Every bilinear system of the form (3.1)-(3.3) can be considered as the sum of two subsystems, one only containing deterministic variables (index d) and the other containing all stochastic variables (index s). The state and the output are then simply the sum of the states and the outputs of both subsystems respectively:

$$\begin{aligned} x_k &= x_k^d + x_k^s, \\ y_k &= y_k^d + y_k^s. \end{aligned}$$

Deterministic subsystem

The deterministic subsystem is described by the following state-space equations:

$$x_{k+1}^d = Ax_k^d + Nu_k \otimes x_k^d + Bu_k, \tag{3.4}$$

$$y_k^d = Cx_k^d + Du_k. \tag{3.5}$$

Stochastic subsystem

The following equations describe the stochastic subsystem:

$$x^s_{k+1} = Ax^s_k + Nu_k \otimes x^s_k + w_k, \tag{3.6}$$
$$y^s_k = Cx^s_k + v_k \tag{3.7}$$

with

$$\mathbf{E}[\begin{pmatrix} u_p \\ w_p \\ v_p \end{pmatrix} \begin{pmatrix} u_q \\ w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} I & 0 & 0 \\ 0 & Q & S \\ 0 & S^T & R \end{pmatrix} \delta_{pq} \geq 0. \tag{3.8}$$

Note that, contrary to linear systems, the stochastic subsystem also contains the input u_k, which is a deterministic signal. As a consequence the stochastic system is non-stationary. However, we will suppose that it is *quasi-stationary* in the sense that the first order statistics are time-invariant while the second order statistics stay within certain bounds:

$$\mathbf{E}[x^s_k] = 0,$$
$$\mathbf{E}[x^s_k(x^s_k)^T] \leq C_b.$$

This implies that the input u_k is such that the eigenvalues of $A + Nu_k \otimes I_n$ are strictly inside the unit circle $\forall\ k$. This can be seen by observing that for linear systems, a necessary condition for the state of the stochastic subsystem (2.6)-(2.8) to be a stationary signal is that A is a stable matrix. In analogy with linear systems one can say that for the bilinear stochastic subsystem (3.6)-(3.8) the state will be a stationary stochastic process when $A + Nu_k \otimes I_n$ is a stable matrix $\forall k$:

Assumption 3.2.1 (Stability condition) *The bilinear stochastic subsystem (3.6)-(3.8) is assumed to be quasi-stationary, which implies that the input u_k is such that:*

$$\max \| \text{ eig } (A + Nu_k \otimes I_n)\| < 1, \quad \forall\ k.$$

3.2.2 Realization theory of bilinear systems

Deterministic bilinear realization [73], [74]

Before going into the matter of bilinear realization theory, we recall some structural properties of bilinear systems.

Definition 3.2.1 (Reachability)
The bilinear system (3.4)-(3.5) is said to be reachable from the origin if, for any state $x_k \in \mathbb{R}^n$, there exists an input sequence that brings the system from the origin to x_k in a finite number of steps.

Definition 3.2.2 (Unobservability [74])
The state x_k of the system (3.4)-(3.5) is said to be unobservable if at the output it is indistinguishable from the origin of the state-space.

Note that the notion of unobservability should not be confused with the observability of a system. For instance, it would be wrong to state that the system is observable if it is not unobservable.

A remarkable property of bilinear systems is that the notions of reachability and unobservability can be described in an analogous way as for linear systems. In [73], [74] it was shown that the reachability and the unobservability of a bilinear system (3.4)-(3.5) can be directly linked to specially structured matrices, in the same way as the controllability and the observability matrix are linked to the controllability and the observability of linear systems:

- A bilinear system (3.4)-(3.5) is reachable from the origin if and only if the extended reversed bilinear reachability matrix Δ_i^d has rank n for $i \geq n$:

$$\Delta_k^d \stackrel{\text{def}}{=} \begin{pmatrix} P_k^d & P_{k-1}^d & \dots & P_1^d \end{pmatrix}$$

  with

$$P_k^d \stackrel{\text{def}}{=} \begin{pmatrix} AP_{k-1}^d & N_1 P_{k-1}^d & \dots & N_m P_{k-1}^d \end{pmatrix}, \quad P_1^d \stackrel{\text{def}}{=} B.$$

- A bilinear system (3.4)-(3.5) has no unobservable states if and only if the unobservability matrix Γ_i^u has rank n for $i \geq n$:

$$\Gamma_i^u \stackrel{\text{def}}{=} \begin{pmatrix} O_1 \\ \vdots \\ O_i \end{pmatrix} \tag{3.9}$$

  with

$$O_k \stackrel{\text{def}}{=} \begin{pmatrix} O_{k-1}A \\ O_{k-1}N_1 \\ \vdots \\ O_{k-1}N_m \end{pmatrix}, \quad O_1 \stackrel{\text{def}}{=} C.$$

Before proceeding to the realization theory of deterministic bilinear systems we recall that the Volterra-kernel description of a general nonlinear discrete-time SISO system, starting from the zero-initial state, is given by:

$$\begin{aligned} y_k &= \sum_{i_1=0}^{k} \mathcal{K}^1_{i_1} u_{k-i_1} + \sum_{i_1=0}^{k}\sum_{i_2=0}^{k} \mathcal{K}^2_{i_1,i_2} u_{k-i_1} u_{k-i_2} + \dots \\ &\quad + \sum_{i_1=0}^{k} \dots \sum_{i_k=0}^{k} \mathcal{K}^k_{i_1,\dots i_k} u_{k-i_1} \dots u_{k-i_k} \end{aligned}$$

where $\mathcal{K}^q_{i_1,\dots,i_q}$ are the qth order Volterra Kernels. It can be seen from (3.4)-(3.5) that, in the case of deterministic bilinear systems, this expression can be rewritten as:

$$y_k = C \begin{pmatrix} P^d_k & \dots & P^d_1 \end{pmatrix} \begin{pmatrix} u^k_1 \\ \vdots \\ u^1_k \end{pmatrix} + D u_k$$

where the generalized inputs u^q_p are defined $\forall q \in \mathbb{N}_0$ as:

$$u^q_p \stackrel{\text{def}}{=} \begin{pmatrix} u^{q-1}_p \\ u_{p+q-1} \otimes u^{q-1}_p \end{pmatrix}, \quad u^1_p \stackrel{\text{def}}{=} u_p. \tag{3.10}$$

Just like the Markov parameters in the case of a linear system, the Volterra-kernels govern the input/output behavior of a bilinear system. These Volterra-kernels are in fact the elements of matrices $\mathcal{K}^d_{pq} \in \mathbb{R}^{l(m+1)^p \times m(m+1)^q}$ that are defined as:

$$\mathcal{K}_{pq} \stackrel{\text{def}}{=} O_p P^d_q, \;\; \forall p, q \in \mathbb{N}_0$$

and $\mathcal{K}^d_{00} \stackrel{\text{def}}{=} D$. The realization problem of the deterministic bilinear subsystem (3.4)-(3.5) can now be stated as:

Deterministic bilinear realization problem

Given *the matrices* $\mathcal{K}^d_{pq}$, $\forall p, q \in \{0, \dots, n+1\}$ *containing the Volterra kernels of the deterministic bilinear system (3.4)-(3.5),* **find** *the state-space matrices* A, B, C, D, N *up to a similarity transformation.*

As for linear systems, the idea of bilinear realization theory is to put the available Volterra kernels into a structured matrix $\mathcal{V}^d_i$ [74]:

$$\mathcal{V}^d_i \stackrel{\text{def}}{=} \begin{pmatrix} \mathcal{K}^d_{11} & \mathcal{K}^d_{12} & \dots & \mathcal{K}^d_{1i} \\ \mathcal{K}^d_{21} & \mathcal{K}^d_{22} & \dots & \mathcal{K}^d_{2i} \\ \dots & \dots & \dots & \dots \\ \mathcal{K}^d_{i1} & \mathcal{K}^d_{i2} & \dots & \mathcal{K}^d_{ii} \end{pmatrix} = \Gamma^u_i \Delta^d_i$$

and to extract the state-space matrices by exploiting its structure. The typical structure of matrices used in bilinear system theory is fundamentally different from the Hankel and Toeplitz matrices that are used in linear system theory. We will refer to them as *generalized* Hankel/Toeplitz matrices. In order to get some feeling with this "bilinear structure", we now give an example of what V_i^d looks like for a single input 2nd order deterministic bilinear system.

Example 3.2.1 *The Volterra kernels we need to realize the system are CB, CAB, CA^2B, CA^3B, CA^2NB, $CANB$, $CANAB$, CAN^2B. These kernels can be structured into a generalized block-Hankel matrix $\mathcal{V}_3^d$ in the following way:*

$$
\begin{aligned}
\mathcal{V}_3^d &= \begin{pmatrix} \mathcal{K}_{11}^d & \mathcal{K}_{12}^d & \mathcal{K}_{13}^d \\ \mathcal{K}_{21}^d & \mathcal{K}_{22}^d & \mathcal{K}_{23}^d \\ \mathcal{K}_{31}^d & \mathcal{K}_{32}^d & \mathcal{K}_{33}^d \end{pmatrix} \\
&= \left(\begin{array}{c|cc|cccc}
CB & CAB & CNB & CA^2B & CANB & CNAB & CN^2B \\ \hline
CAB & CA^2B & CANB & CA^3B & CA^2NB & CANAB & CAN^2B \\
CNB & CNAB & CN^2B & CNA^2B & CNANB & CN^2AB & CN^3B \\ \hline
CA^2B & CA^3B & CA^2NB & CA^4B & CA^3NB & CA^2NAB & CA^2N^2B \\
CANB & CANAB & CAN^2B & CANA^2B & CANANB & CAN^2AB & CAN^3B \\
CNAB & CNA^2B & CNANB & CNA^3B & CNA^2NB & CNANAB & CNAN^2B \\
CN^2B & CN^2AB & CN^3B & CN^2A^2B & CN^2ANB & CN^3AB & CN^4B
\end{array}\right) \\
&= \left(\begin{array}{c} C \\ \hline CA \\ CN \\ \hline CA^2 \\ CAN \\ CNA \\ CN^2 \end{array}\right) \left(\begin{array}{c|cc|cccc} B & AB & NB & A^2B & ANB & NAB & N^2B \end{array}\right) \\
&= \Gamma_3^u \Delta_3^d.
\end{aligned}
$$

In the case of general nth order MIMO bilinear systems, a possible realization algorithm is the following:

1. Construct the matrix $\mathcal{V}_i^d$, $i > n$ from the available Volterra kernels.
2. Calculate the singular value decomposition of $\mathcal{V}_i^d$:

$$
\mathcal{V}_i^d \overset{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}
$$

   and derive Γ_i^u and Δ_i^d:

$$
\Gamma_i^u = U_1 S_1^{1/2},
$$

$$\Delta_i^d = S_1^{1/2} V_1^T.$$

3. Determine the elements $O_i \in \mathbb{R}^{l(m+1)^i \times n}$, $\forall i \in \{1, \ldots, n\}$ using the definition of the unobservability matrix Γ_i^u (3.9).

4. Calculate the state-space matrices:

$$\begin{aligned}
A &= \begin{pmatrix} O_1 \\ \vdots \\ O_{i-1} \end{pmatrix}^\dagger \begin{pmatrix} O_2(1 : l(m+1), :) \\ \vdots \\ O_i(1 : l(m+1)^{i-1}, :) \end{pmatrix}, \\
N_p &= \begin{pmatrix} O_1 \\ \vdots \\ O_{i-1} \end{pmatrix}^\dagger \begin{pmatrix} O_2(pl(m+1)+1 : (p+1)l(m+1)+l, :) \\ \vdots \\ O_i(pl(m+1)^{i-1}+1 : (p+1)l(m+1)^{i-1}+l, :) \end{pmatrix}, \\
B &= \Delta_i^d(:, 1 : m), \\
C &= \Gamma_i^u(1 : l, :), \\
D &= \mathcal{K}_{00}^d.
\end{aligned}$$

Stochastic bilinear realization [35]

The realization of the stochastic bilinear system (3.6)-(3.8) was first considered in [35]. As for linear stochastic realization, the knowledge of certain specific covariances is assumed. These covariances are defined as:

$$\mathcal{K}_{pq}^s \stackrel{\text{def}}{=} \mathbf{E}[y_p^{i-p+1,s}(y_q^{i-q+1,s})^T]$$

where $y_{p,s}^q$ are generalized outputs of the stochastic subsystem, defined in a similar way as the generalized inputs (3.10):

$$y_{p,s}^q \stackrel{\text{def}}{=} \begin{pmatrix} y_p^{q-1,s} \\ u_{p+q-1} \otimes y_p^{q-1,s} \end{pmatrix}, \quad y_{p,s}^1 \stackrel{\text{def}}{=} y_p^s. \tag{3.11}$$

The stochastic realization problem can now be stated as follows:

<u>Stochastic bilinear realization problem</u>

Given *the generalized output covariances* $\mathcal{K}_{pq}^s$, $p, q \in \{0, \ldots, n+1\}$ *of the stochastic bilinear system (3.6)-(3.7),* **find** *the state-space matrices* A, C, N, Q, R, S *up to a similarity transformation.*

As in the linear case, the bilinear stochastic subsystem is characterized by the covariance matrices G, Λ_0 and Σ^s, that are defined in the same way as for linear systems. With the assumption that the inputs are white and independent of the noise we have:

$$
\begin{aligned}
\Sigma^s &\stackrel{\text{def}}{=} \mathbf{E}[x_k^s (x_k^s)^T] \\
&= \mathbf{E}[x_{k+1}^s (x_{k+1}^s)^T] \\
&= \mathbf{E}[(Ax_k^s + Nu_k \otimes x_k^s + w_k)(Ax_k^s + Nu_k \otimes x_k^s + w_k)^T] \\
&= A\Sigma^s A^T + \sum_{p=1}^{m} N_p \Sigma^s N_p^T + Q, \\
\Lambda_0 &\stackrel{\text{def}}{=} \mathbf{E}[y_k^s (y_k^s)^T] \\
&= \mathbf{E}[(Cx^s + v_k)(Cx^s + v_k)^T] \\
&= C\Sigma^s C^T + R, \\
G &\stackrel{\text{def}}{=} \mathbf{E}[x_{k+1}^s (y_k^s)^T] \\
&= \mathbf{E}[(Ax_k^s + Nu_k \otimes x_k^s + w_k)(Cx_k^s + v_k)^T] \\
&= A\Sigma^s C^T + S.
\end{aligned}
$$

As for the deterministic bilinear realization, the idea here is to put the available generalized output covariances $\mathcal{K}_{pq}^s$ into a structured matrix $\mathcal{V}_i^s$:

$$
\mathcal{V}_i^s \stackrel{\text{def}}{=} \begin{pmatrix} \mathcal{K}_{11}^s & \mathcal{K}_{12}^s & \dots & \mathcal{K}_{1i}^s \\ \mathcal{K}_{21}^s & \mathcal{K}_{22}^s & \dots & \mathcal{K}_{2i}^s \\ \dots & \dots & \dots & \dots \\ \mathcal{K}_{i1}^s & \mathcal{K}_{i2}^s & \dots & \mathcal{K}_{ii}^s \end{pmatrix}
$$

and exploit its structure to extract the state-space matrices. It was shown [35] that the generalized covariance matrix $\mathcal{V}_i^s$ can be written as:

$$
\mathcal{V}_i^s = \Gamma_i^u \Delta_i^c
$$

where Γ_i^u is the extended unobservability matrix (3.9) and Δ_i^c the reversed extended stochastic reachability matrix of the stochastic bilinear subsystem (3.6)-(3.7) defined as:

$$
\Delta_i^c \stackrel{\text{def}}{=} \begin{pmatrix} P_i^c & P_{i-1}^c & \dots & P_1^c \end{pmatrix}, \tag{3.12}
$$

where the superscript "c" stands for "covariance" and:

$$
P_k^c \stackrel{\text{def}}{=} \begin{pmatrix} AP_{k-1}^c & N_1 P_{k-1}^c & \dots & N_m P_{k-1}^c \end{pmatrix}, \quad P_1^c = G.
$$

The stochastic realization algorithm is very similar to the one of the deterministic subsystem:

1. Construct the matrix $\mathcal{V}_i^s$, $i > n$ from the available generalized output covariances $\mathcal{K}_{pq}^s$.
2. Calculate the singular value decomposition of $\mathcal{V}_i^s$:

$$\mathcal{V}_i^s \stackrel{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

   and derive Γ_i^u and Δ_i^c:

$$\begin{aligned} \Gamma_i^u &= U_1 S_1^{1/2}, \\ \Delta_i^c &= S_1^{1/2} V_1^T. \end{aligned}$$

3. Determine the elements $O_i \in \mathbb{R}^{l(m+1)^i \times n}$, $\forall i \in \{1, \ldots, n\}$ using the definition of the unobservability matrix Γ_i^u (3.9).
4. Calculate the state-space matrices:

$$\begin{aligned} A &= \begin{pmatrix} O_1 \\ \vdots \\ O_{i-1} \end{pmatrix}^{\dagger} \begin{pmatrix} O_2(1 : l(m+1), :) \\ \vdots \\ O_i(1 : l(m+1)^{i-1}, :) \end{pmatrix}, \\ N_p &= \begin{pmatrix} O_1 \\ \vdots \\ O_{i-1} \end{pmatrix}^{\dagger} \begin{pmatrix} O_2(pl(m+1)+1 : (p+1)l(m+1)+l, :) \\ \vdots \\ O_i(pl(m+1)^{i-1}+1 : (p+1)l(m+1)^{i-1}+l, :) \end{pmatrix}, \\ G &= \Delta_i^c(:, 1:m), \\ C &= \Gamma_i^u(1:l, :), \\ \Lambda_0 &= \mathcal{K}_{00}^s. \end{aligned}$$

5. Determine Q, R, S from:

$$\begin{aligned} Q &= \Sigma^s - A\Sigma^s A^T + \sum_{p=1}^{m} N_p \Sigma^s N_p^T, \\ R &= \Lambda_0 - C\Sigma^s C^T, \\ S &= G - A\Sigma^s C^T \end{aligned}$$

   where the stochastic state covariance matrix Σ^s can be chosen as any element of the convex set:

$$\mathcal{F} \le \Sigma^s \le \mathcal{B}^{-1}.$$

   $\mathcal{F}$ and $\mathcal{B}$ are the solution of the following bilinear Riccati equations:

$$\mathcal{F} = A\mathcal{F}A^T + \sum_{p=1}^{m} N_p \mathcal{F} N_p^T +$$

$$
\begin{aligned}
& (G - A\mathcal{F}C^T)(\Lambda_0 - C\mathcal{F}C^T)^{-1}(G - A\mathcal{F}C^T)^T, \\
\mathcal{B} &= A\mathcal{B}A^T + \sum_{p=1}^{m} N_p \mathcal{B} N_p^T + \\
& (G - A\mathcal{B}C^T)(\Lambda_0 - C\mathcal{B}C^T)^{-1}(G - A\mathcal{B}C^T)^T.
\end{aligned}
$$

As we will see in the next section, if Σ^s is chosen equal to $\mathcal{F}$, the realized bilinear model is in forward innovation form (See also Figure 3.2).

A major drawback of realization theory is the fact that the prior knowledge of the generalized covariances $\mathcal{K}^s_{pq}$ in the stochastic case and the Volterra kernel matrices $\mathcal{K}^d_{pq}$ in the deterministic case is required. We will see that the bilinear subspace identification algorithm, presented in the following sections, permits the calculation of the system matrices directly from the measured input/output data i.e. without prior knowledge of $\mathcal{K}^s_{pq}$ or $\mathcal{K}^d_{pq}$.

3.2.3 Kalman filter

In this section, we derive the Kalman filter equations of the bilinear system (3.1)-(3.3). The use of doing so will become clear in the following sections where it is shown that, as in linear subspace identification, a nonsteady-state Kalman filter estimate of the system can be obtained directly from the data i.e. without knowledge of the system matrices. Since bilinear systems are in fact a particular case of linear time-varying systems, where the A-matrix is function of the input ($A_k = A + \sum_{i=1}^{M} N_i u_{k,i}$), the Kalman filter equations for linear time-varying systems also hold for bilinear systems. Due to the assumption of the whiteness of the input u_k, the final result is quite simple and very similar to the Kalman filter equations of linear time-invariant systems.

In analogy with the linear Kalman filter (2.9)-(2.10), the Kalman filter of the bilinear system (3.1)-(3.3) has the form:

$$\widehat{x}_{k+1} = A\widehat{x}_k + N u_k \otimes \widehat{x}_k + B u_k + K(y_k - C\widehat{x}_k - D u_k) \qquad (3.13)$$

where the steady-state Kalman gain K is such that the trace of the state error covariance matrix is minimized:

$$\min_K \ \mathrm{tr}\ \mathbf{E}[(x_{k+1} - \widehat{x}_{k+1})^T (x_{k+1} - \widehat{x}_{k+1})].$$

The following theorem gives an expression for this Kalman gain.

Theorem 3.2.3 (Kalman filter)
The steady-state Kalman gain K of the system (3.1)-(3.3) is the solution of the following set of equations:

$$K = (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}, \qquad (3.14)$$

Bilinear forward innovation form:

$$x_{k+1} = Ax_k + Nu_k \otimes x_k + Bu_k + Ke_k, \quad (3.16)$$

$$y_k = Cx_k + Du_k + e_k \quad (3.17)$$

with

$$\mathbf{E}[e_k e_k^T] = \Lambda_0 - C\Sigma^s C^T, \quad (3.18)$$

$$\Sigma^s = A\Sigma^s A^T + \sum_{p=1}^{m} N_p \Sigma^s N_p^T + (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}(G - A\Sigma^s C^T)^T, \quad (3.19)$$

$$K = (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}. \quad (3.20)$$

Figure 3.2: **Forward innovation representation of the bilinear system (3.1)-(3.3).**

$$\Sigma^s = A\Sigma^s A^T + \sum_{p=1}^{m} N_p \Sigma^s N_p^T + (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}(G - A\Sigma^s C^T)^T. \quad (3.15)$$

Proof: See Appendix A.1. ■

The forward innovation representation of (3.1)-(3.3), corresponding to the Kalman filter equations (3.13)-(3.15), is given in Figure 3.2. For simplicity's sake we will, just as for linear subspace identification, continue the theoretical derivations with the forward innovation representation of the system.

3.3 Subspace structure of bilinear systems

We have seen in Section 2.3 that the matrix input/output equations (2.27)-(2.29) played a central role in linear subspace identification. In the present section we introduce, in a similar way, the generalized data Hankel matrices and the matrix input/output equations used in bilinear subspace identification.

3.3.1 System related matrices

- In Section 3.2.2 we defined the extended reversed reachability matrix Δ_i^d and Δ_i^c of the deterministic and stochastic bilinear system. Here, these matrices are redefined such that they contain the same elements, but in a different order::

$$\begin{aligned}\Delta_i^d &\stackrel{\text{def}}{=} \begin{pmatrix} B & A\Delta_{i-1}^d & N_1\Delta_{i-1}^d & \dots & N_m\Delta_{i-1}^d \end{pmatrix}, \quad \Delta_1^d = B,\\ \Delta_i^c &\stackrel{\text{def}}{=} \begin{pmatrix} G & A\Delta_{i-1}^c & N_1\Delta_{i-1}^c & \dots & N_m\Delta_{i-1}^c \end{pmatrix}, \quad \Delta_1^c = G.\end{aligned}$$

  The extended reachability matrix of the stochastic subsystem of the innovation form (3.16)-(3.17) is defined as:

$$\Delta_i^s \stackrel{\text{def}}{=} \begin{pmatrix} K & A\Delta_{i-1}^c & N_1\Delta_{i-1}^c & \dots & N_m\Delta_{i-1}^c \end{pmatrix}, \quad \Delta_1^c = K.$$

- The generalized bilinear observability matrix Γ_i for bilinear systems:

$$\Gamma_i \stackrel{\text{def}}{=} \begin{pmatrix} CA_{i-1} & 0 & \dots & 0 \\ \Gamma_{i-1} & 0 & \dots & 0 \\ 0 & \Gamma_{i-1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Gamma_{i-1} \end{pmatrix}, \quad \Gamma_1 \stackrel{\text{def}}{=} \begin{pmatrix} C & 0_{l\times mn} \end{pmatrix},$$

  and

$$A_i \stackrel{\text{def}}{=} \begin{pmatrix} AA_{i-1} & N_1A_{i-1} & \dots & N_mA_{i-1} \end{pmatrix}, \quad A_0 \stackrel{\text{def}}{=} I_n.$$

  Note that the system (3.16)-(3.17) is not necessarily observable if the generalized observability matrix Γ_n is of full rank. We also define the reversed linear observability matrix:

$$\Gamma_i^l \stackrel{\text{def}}{=} \begin{pmatrix} CA^{i-1} \\ \vdots \\ CA \\ C \end{pmatrix}. \tag{3.21}$$

- $$H_i^d \stackrel{\text{def}}{=} \left(\begin{array}{c|c|cccc} D & C\Delta_{i-1}^d & 0 & \dots & 0 \\ \hline 0 & H_{i-1}^d & 0 & \dots & 0 \\ \hline 0 & 0 & H_{i-1}^d & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & H_{i-1}^d \end{array}\right), \quad H_1^d \stackrel{\text{def}}{=} D,$$

$$H_i^s \stackrel{\text{def}}{=} \left(\begin{array}{c|c|cccc} I_l & C\Delta_{i-1}^s & 0 & \dots & 0 \\ \hline 0 & H_{i-1}^s & 0 & \dots & 0 \\ \hline 0 & 0 & H_{i-1}^s & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & H_{i-1}^s \end{array}\right), \quad H_1^s \stackrel{\text{def}}{=} I_l.$$

- $$L_i \stackrel{\text{def}}{=} \begin{pmatrix} \Lambda_0 & C\Delta_i^c & 0 & \dots & 0 \\ (\Delta_i^c)^T C^T & L_{i-1} & 0 & \dots & 0 \\ 0 & 0 & L_{i-1} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & L_{i-1}, \end{pmatrix}, \quad L_1 \stackrel{\text{def}}{=} \Lambda_0.$$

3.3.2 Generalized data block-Hankel matrices

The data matrices used in bilinear subspace identification are a generalization of the data matrices (2.19)-(2.20) used in linear subspace identification. They not only contain the inputs and the outputs of the system, but also specific products between these inputs and outputs. In this way, they directly contain the "bilinear information" on the system which is represented by the product terms between the inputs and outputs:

$$Y_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} Y_{i-1} \\ Y_{0|i-2} \\ U_{i-1} \odot Y_{0|i-2} \end{pmatrix}, \quad Y_{0|0} \stackrel{\text{def}}{=} Y_0, \tag{3.22}$$

$$U_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} U_{i-1} \\ U_{0|i-2} \\ U_{i-1} \odot U_{0|i-2} \end{pmatrix}, \quad U_{0|0} \stackrel{\text{def}}{=} U_0, \tag{3.23}$$

$$E_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} E_{i-1} \\ E_{0|i-2} \\ U_{i-1} \odot E_{0|i-2} \end{pmatrix}, \quad E_{0|0} \stackrel{\text{def}}{=} E_0 \tag{3.24}$$

and:

$$\begin{aligned} X_q &\stackrel{\text{def}}{=} \begin{pmatrix} x_q & x_{q+1} & \dots & x_{q+j-1} \end{pmatrix}, \\ U_q &\stackrel{\text{def}}{=} \begin{pmatrix} u_q & u_{q+1} & \dots & u_{q+j-1} \end{pmatrix}, \\ Y_q &\stackrel{\text{def}}{=} \begin{pmatrix} y_q & y_{q+1} & \dots & y_{q+j-1} \end{pmatrix}, \\ E_q &\stackrel{\text{def}}{=} \begin{pmatrix} e_q & e_{q+1} & \dots & e_{q+j-1} \end{pmatrix}. \end{aligned}$$

The Khatri-Rao product $\odot$ of two matrices F and G having the same number of columns p is defined in [83] as the columnwise Kronecker product[1]:

$$F \odot G \stackrel{\text{def}}{=} \begin{pmatrix} f_1 \otimes g_1 & f_2 \otimes g_2 & \dots & f_p \otimes g_p \end{pmatrix}.$$

Further we define the generalized inputs and outputs:

$$y_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} y_{i-1} \\ y_{0|i-2} \\ u_{i-1} \otimes y_{0|i-2} \end{pmatrix}, \qquad u_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} u_{i-1} \\ u_{0|i-2} \\ u_{i-1} \otimes u_{0|i-2} \end{pmatrix} \tag{3.25}$$

with initial conditions $y_{0|0} \stackrel{\text{def}}{=} y_0$ and $u_{0|0} \stackrel{\text{def}}{=} u_0$. The "past" and "future" generalized Hankel matrices can now be defined as:

$$Y_p \stackrel{\text{def}}{=} Y_{0|i-1}, \qquad Y_f \stackrel{\text{def}}{=} Y_{i|2i-1}, \tag{3.26}$$

$$U_p \stackrel{\text{def}}{=} U_{0|i-1}, \qquad U_f \stackrel{\text{def}}{=} U_{i|2i-1}, \tag{3.27}$$

$$E_p \stackrel{\text{def}}{=} E_{0|i-1}, \qquad E_f \stackrel{\text{def}}{=} E_{i|2i-1} \tag{3.28}$$

and

$$W_p \stackrel{\text{def}}{=} W_{0|i-1} = \begin{pmatrix} Y_{0|i-1} \\ U_{0|i-1} \end{pmatrix}.$$

Further we also define the following matrices:

$$U^P_{0|i-1} \stackrel{\text{def}}{=} \begin{pmatrix} U^P_{0|i-2} \\ U_{i-1} \odot U^P_{0|i-2} \end{pmatrix}, \quad U^P_{0|0} \stackrel{\text{def}}{=} \begin{pmatrix} I_{1\times j} \\ U_0 \end{pmatrix}$$

and

$$\bar{U}^P_{0|i-1} \stackrel{\text{def}}{=} U^P_{0|i-1}(m+1:end,:).$$

For these matrices we will use the same short-hand notation as before:

$$U^P_p \stackrel{\text{def}}{=} U^P_{0|i-1}, \qquad \bar{U}^P_p \stackrel{\text{def}}{=} \bar{U}^P_{0|i-1}, U^P_f \stackrel{\text{def}}{=} U^P_{i|2i-1}, \bar{U}^P_f \stackrel{\text{def}}{=} \bar{U}^P_{i|2i-1}.$$

Interesting to note is that the generalized data Hankel matrices (3.22)-(3.23) are also a generalization of the linear data block-Hankel matrices (2.19)-(2.20). Indeed, it can be seen that the first il rows of (3.22)-(3.23) are in fact nothing else than the rows of the linear data block-Hankel matrices:

$$Y^l_p \stackrel{\text{def}}{=} Y_p(1:il,:),$$

[1]Note that the Khatri-Rao product has priority over the classical matrix product: $AB \odot C = A(B \odot C)$

$$Y_f^l \stackrel{\text{def}}{=} Y_f(1:il,:).$$

An issue that will play an important role when considering the implementation aspects of bilinear subspace identification, and to which we will come back in Section 4.4, is the fact that the number of rows in the generalized data Hankel matrices (3.22)-(3.23) grows exponentially with the index i. For instance the number of rows in $Y_{0|i-1}$ is equal to $l/m((m+1)^i-1)$ and in $U_{0|i-1}$ equal to $(m+1)^i-1$.

3.3.3 Matrix input/output equations

Like for linear systems, it is possible to find input/output equations relating the different generalized data matrices. These equations are fundamental for subspace identification since they permit to see how the states and the future generalized outputs can be expressed as a *linear* combination of past and future generalized inputs and outputs.

Theorem 3.3.1 (Matrix input/output equations)
For the bilinear system (3.16)-(3.17) we have:

$$\begin{aligned} Y_p &= \Gamma_i U_p^P \odot X_p + H_i^d U_p + H_i^s E_p, && (3.29)\\ Y_f &= \Gamma_i U_f^P \odot X_f + H_i^d U_f + H_i^s E_f, && (3.30)\\ X_f &= A_i U_f^P \odot X_p + \Delta_i^d U_p + \Delta_i^s E_p. && (3.31) \end{aligned}$$

Proof: The proof directly follows from the definitions in Sections 3.3.1, 3.3.2 and the state-space equations of the bilinear system (3.16)-(3.17). ■

Corollary 3.3.2
An alternative form for (3.29)-(3.31) is:

$$\begin{aligned} Y_p^l &= \Gamma_i^l X_p + \Gamma_i^b \bar{U}_p^P \odot X_p + H_i^{d,b} U_p + H_i^{s,b} E_p, && (3.32)\\ Y_f^l &= \Gamma_i^l X_f + \Gamma_i^b \bar{U}_f^P \odot X_f + H_i^{d,b} U_f + H_i^{s,b} E_f, && (3.33)\\ X_f &= A^i X_p + A_i^b \bar{U}_p^P \odot X_p + \Delta_i^d U_p + \Delta_i^s E_p && (3.34) \end{aligned}$$

where

$$\begin{aligned} \Gamma_i^b &\stackrel{\text{def}}{=} \Gamma_i(1:il, n+1:(m+1)^i n),\\ H_i^{d,b} &\stackrel{\text{def}}{=} H_i^d(1:il,:),\\ H_i^{s,b} &\stackrel{\text{def}}{=} H_i^s(1:il,:),\\ A_i^b &\stackrel{\text{def}}{=} A_i(:, n+1:(m+1)^i n). \end{aligned}$$

3.4 A bilinear subspace identification algorithm

In this section we derive the theory of subspace identification for bilinear systems with white inputs. In order to increase the transparency of the section, the same structure has been used as in Section 2.4. One can also clearly distinguish three steps in the algorithm: the projection or LQ-decomposition step, the SV-decomposition step and finally the extraction of the state-space matrices.

3.4.1 First step: LQ-decomposition

In this section we will first analyze how a Kalman filter state estimate of the bilinear system (3.1)-(3.3) can be expressed as a linear combination of the generalized inputs and outputs (3.25). This result will then be used in the main bilinear subspace identification theorem where it shown how a specific projection of the previously defined generalized data Hankel matrices (3.22)-(3.23) is related to these Kalman filter states.

Theorem 3.4.1 (Nonsteady-state Kalman filter [50])
Given:

- *the initial state estimate* $\widehat{x}_0$*,*
- *the initial stochastic state covariance matrix* $\Sigma_0^s \overset{\text{def}}{=} \mathbf{E}[\widehat{x}_0^s(\widehat{x}_0^s)^T]$,
- *the input and output measurements* $u_0, y_0, \dots u_{i-1}, y_{i-1}$*,*

then the nonsteady-state Kalman filter state estimate $\widehat{x}_i$*, solution of the following recursive equations:*

$$\widehat{x}_{k+1} = A\widehat{x}_k + Nu_k \otimes \widehat{x}_k + Bu_k + K_k(y_k - C\widehat{x}_k - Du_k), \quad (3.35)$$

and

$$\begin{aligned} K_k &= (G - A\Sigma_k^s C^T)(\Lambda_0 - C\Sigma_k^s C^T)^{-1}, & (3.36)\\ \Sigma_{k+1}^s &= A\Sigma_k^s A^T + \sum_{p=1}^{m} N_p \Sigma_k^s N_p^T + \\ &\quad (G - A\Sigma_k^s C^T)(\Lambda_0 - C\Sigma_k^s C^T)^{-1}(G - A\Sigma_k^s C^T)^T, & (3.37) \end{aligned}$$

can be explicitely written as:

$$\widehat{x}_i = \left(\begin{array}{c|c|c} A_i - \Omega_i\Gamma_i & \Delta_i^c - \Omega_i H_i^d & \Omega_i \end{array} \right) \left(\begin{array}{c} u_{0|i-1}^P \otimes \widehat{x}_0 \\ \hline u_{0|i-1} \\ \hline y_{0|i-1} \end{array} \right) \quad (3.38)$$

$$\Sigma_i \quad = \quad A_i\Sigma_0^i A_i + (\Delta_i^c - A_i\Sigma_0^i\Gamma_i^T)(L_i - \Gamma_i\Sigma_0^i\Gamma_i^T)^{-1}(\Delta_i^c - A_i\Sigma_0^i\Gamma_i^T)^T \quad (3.39)$$

where $u_{0|i-1}$, $y_{0|i-1}$ and $u^P_{0|i-1}$ are the generalized inputs and outputs calculated from $u_0, y_0, \dots, u_{i-1}, y_{i-1}$ as defined in (3.25) and:

$$\Omega_i \quad \stackrel{\text{def}}{=} \quad (\Delta_i^c - A_i\Sigma_0^i\Gamma_i^T)(L_i - \Gamma_i\Sigma_0^i\Gamma_i^T)^{-1}, \qquad (3.40)$$

$$\Sigma_0^i \quad \stackrel{\text{def}}{=} \quad \begin{pmatrix} \Sigma_0^{i-1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \Sigma_0^{i-1} \end{pmatrix}, \quad \Sigma_0^0 \stackrel{\text{def}}{=} \Sigma_0^s. \qquad (3.41)$$

Proof: See Appendix A.2. ∎

The following theorem is the main subspace identification algorithm. It shows the link between a specific projection of the structured data matrices (3.26)-(3.27) and the Kalman filter states of that system.

Theorem 3.4.2 (Main bilinear subspace identification theorem [50])

Under the assumptions that:

- *the deterministic input u_k is white and uncorrelated with the noise e_k*
- *there is an infinite amount of measurements available ($j \to \infty$),*
- *the noise e_k is not identically zero,*

we have that the orthogonal projection $\mathcal{O}_i$, defined as:

$$\mathcal{O}_i \quad \stackrel{\text{def}}{=} \quad Y_f^l / W_p \;=\; \Gamma_i^l \widehat{X}_f \qquad (3.42)$$

equals the product between the linear reversed extended observability matrix Γ_i^l and the nonsteady-state Kalman filter estimate $\widehat{X}_f$ of the state sequence X_f, as defined in Theorem 3.4.1, with the initial conditions $\widehat{X}_0 = 0$ and $\Sigma_0^s = -\Sigma^d \stackrel{\text{def}}{=} -E[x_k^d(x_k^d)^T]$.

Proof: See Appendix A.3. ∎

As for linear subspace identification, the Kalman filter estimated state sequence $\widehat{X}_f$ calculated with projection (3.42) is the nonsteady-state solution of a bank of Kalman filters. Every element $\widehat{x}_i$ of $\widehat{X}_f$ is the nonsteady-state solution of the Kalman filter as defined by Theorem 3.4.1 with initial state estimate $\widehat{x}_0 = 0$ and $\Sigma_0^s = -\Sigma^d$. For a more detailed interpretation we refer to Figure 2.5.

As we have seen in Section 2.5, the projection Y_f^l/W_p can be calculated in a numerically efficient way with the following LQ-decomposition:

$$\begin{pmatrix} W_p \\ Y_f^l \end{pmatrix} \stackrel{\text{LQ}}{=} \begin{pmatrix} L_{bb} & 0 \\ L_{ab} & L_{aa} \end{pmatrix} \begin{pmatrix} Q_b^T \\ Q_a^T \end{pmatrix}.$$

The projection (3.42) then becomes:

$$\begin{aligned} Y_f^l/W_p &= L_{ab}L_{bb}^{\dagger}W_p \\ &= L_w W_p. \end{aligned}$$

3.4.2 Second step: SV-decomposition

In the previous step we have seen how the product between the reversed extended linear observability matrix Γ_i^l and the nonsteady-state Kalman filter estimate $\widehat{X}_f$ of the state sequence X_f can be calculated with the following orthogonal projection:

$$Y_f^l/W_p = \Gamma_i^l \widehat{X}_f.$$

In the second step of subspace system identification this term is split up into its components i.e. Γ_i^l and $\widehat{X}_f$. If the number of columns in the data block-Hankel matrices j were infinite, the projection Y_f^l/W_p would be a rank-n matrix where n is the order of the unknown system (3.1)-(3.3). In practice however, due to the noise and the finite amount of data, it will not be rank-deficient. A singular value decomposition is then used to approximate it by a rank-n matrix. Instead of calculating the SVD of Y_f^l/W_p, it is more advantageous to calculate the SVD of L_w, which has the same rank but is of much smaller dimensions:

$$\begin{aligned} W_1 L_w W_2 &= \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \\ &\approx U_1 S_1 V_1^T. \end{aligned} \tag{3.43}$$

The matrices W_1 and W_2 are user-defined weights that influence the state-space basis in which the resulting bilinear state-space model will be calculated. The order n of the system is estimated by inspecting the number of dominant singular values S_1. The observability matrix and the state sequence of the system can now be directly determined as:

$$\Gamma_i^l = W_1^{-1} U_1 S_1^{1/2}, \tag{3.44}$$

$$\widehat{X}_f = S_1^{1/2} V_1^T W_2^{-1} W_p. \tag{3.45}$$

3.4.3 Third step: recovering the state-space matrices

The last part of the algorithm consists in finding the state-space matrices A, B, C, D, N and the noise covariances Q, R, S from Γ_i^l and/or $\widehat{X}_f$. The problem of finding the system model and the noise model are treated separately:

System model A, B, C, D, N: Once the states are known, it becomes an easy problem to find the state-space matrices by solving the following least squares problem:

$$\begin{pmatrix} \widehat{X}_{i+1} \\ Y_i \end{pmatrix} = \begin{pmatrix} A & N & B \\ C & 0 & D \end{pmatrix} \begin{pmatrix} \widehat{X}_i \\ U_i \odot \widehat{X}_i \\ U_i \end{pmatrix} + \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}, \quad (3.46)$$

where ρ_w and ρ_v are the residual matrices. This set of equations is consistent, if solved in a least squares sense, when $\widehat{X}_i$ and $\widehat{X}_{i+1}$ are consecutive estimates of the same Kalman filter. This means that the corresponding elements of $\widehat{X}_i$ and $\widehat{X}_{i+1}$ should be consecutive solutions of the Kalman filter equations (3.35)-(3.37) i.e. with the same initial conditions and the same inputs (See also Figures 2.5 and 2.6 for a graphical representation). If $\widehat{X}_i$ and $\widehat{X}_{i+1}$ are calculated from the following projections, their initial condition $\widehat{X}_0$ and Σ_0^s are the same:

$$\begin{aligned} Y_{i|2i-1}^l / W_{0|i-1} &= \Gamma_i^l \widehat{X}_i \quad \longrightarrow \quad \widehat{X}_0 = 0,\ \Sigma_0^s = -\Sigma^d, \\ Y_{i+1|2i-1}^l / W_{0|i} &= \Gamma_{i-1}^l \widehat{X}_{i+1} \quad \longrightarrow \quad \widehat{X}_0 = 0,\ \Sigma_0^s = -\Sigma^d \end{aligned}$$

which guarantees the consistency of (3.46). Care should be taken about the fact that $\widehat{X}_i$ and $\widehat{X}_{i+1}$ are calculated in the same state-space basis. This can be achieved by first calculating Γ_i^l and $\widehat{X}_i$ from the first projection. Finally determine Γ_{i-1}^l as $\Gamma_{i-1}^l = \Gamma_i^l(l+1 : end, :)$ and calculate $\widehat{X}_{i+1}$:

$$\widehat{X}_{i+1} = (\Gamma_{i-1}^l)^\dagger Y_{i+1|2i-1}^l / W_{0|i}.$$

Note that, since we know that the solution contains a zero matrix, (3.46) could also be solved as a constrained linear least squares problem . This zero matrix also indicates that more general forms of nonlinearities in the state-space equations could be considered, for instance a product between states and inputs in the output equation. But, as we mentioned in the Introduction, this is not common at all in the literature.

Noise model Q, R, S: The noise covariances Q, R and S can be estimated from the residuals ρ_w and ρ_v of the least squares problem (3.46):

$$\begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}_i = \frac{1}{j}[\begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix} \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}^T] \geq 0$$

where the index i denotes a bias induced for finite i. We suspect and experienced that this bias disappears as $i \to \infty$. The proof would consist in showing that $A + Nu_k \otimes x_k - KC$ is a stable matrix. In that case, the nonsteady-state Kalman filter sequences $\widehat{X}_i$ and $\widehat{X}_{i+1}$ will have turned into steady-state Kalman sequences. As we will analyze into more details in Section 4.4, it is not possible in practice to choose large values for i since the number of rows in the generalized data block-Hankel matrices grows exponentially with the index i. As a consequence, the Kalman filter state estimate $\widehat{X}_i$ obtained from the algorithm, could be far from the steady-state. As will be illustrated in the examples of Chapter 5, this will lead to biased estimates of the noise covariance matrices Q, R and S.

3.4.4 N4SID algorithm

Figure 3.3 presents a bilinear N4SID subspace identification algorithm. It clearly consists of three steps namely an LQ-decomposition, an SV-decomposition and a final step where the state-space matrices are extracted. Although the estimates of A, B, C, D and N are consistent, the estimates of the noise covariance matrices Q, R, S will only be consistent if the number of block rows in the data matrices W_p is infinite ($i = \infty$).

In the present chapter, we have introduced subspace identification theory for bilinear systems. The main assumption we made is that the inputs are white and independent of the noise. In a similar way as for linear subspace identification it was shown how bilinear stochastic/deterministic realization theory, Kalman filtering of bilinear systems and the matrix input/output equations (3.29)-(3.31) are related to subspace identification of bilinear systems. The final algorithm is very similar to that used in linear subspace identification. It for instance also consists of three main steps, namely LQ-decomposition, SV-decomposition and extraction of the state-space matrices.

N4SID algorithm for bilinear systems with white inputs

1. Calculate the orthogonal projections with an **LQ-decomposition**:

$$\begin{aligned}\mathcal{O}_i &\overset{\text{LQ}}{=} Y^l_{i|2i-1}/W_{0|i-1} = L_w W_p,\\ \mathcal{O}_{i+1} &\overset{\text{LQ}}{=} Y^l_{i+1|2i-1}/W_{0|i}\end{aligned}$$

   and determine L_w.

2. Calculate the weighed **SV**-decomposition:

$$W_1 L_w W_2 \overset{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

   and

   - determine an estimate of the order n of the system by inspecting the number of dominant singular values S_1,
   - determine Γ^l_i and Γ^l_{i-1} as:

$$\Gamma^l_i = W_1^{-1} U_1 S_1^{1/2}, \qquad \Gamma^l_{i-1} = \Gamma^l_i(1 : end - l, :),$$

   - determine the Kalman filter state sequences:

$$\widehat{X}_i = S_1^{1/2} V_1^T W_2^{-1} W_p, \qquad \widehat{X}_{i+1} = \Gamma^{l,\dagger}_{i-1} \mathcal{O}_{i+1}.$$

3. Calculated the estimates of the system model A, B, C, D and the noise model Q, R, S from the following linear least squares problem:

   System model:

$$\begin{pmatrix} \widehat{X}_{i+1} \\ Y_i \end{pmatrix} = \begin{pmatrix} A & N & B \\ C & 0 & D \end{pmatrix} \begin{pmatrix} \widehat{X}_i \\ U_i \odot \widehat{X}_i \\ U_i \end{pmatrix} + \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}.$$

   Noise model:

$$\begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}_i = \frac{1}{j}[\begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix} \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}^T].$$

Figure 3.3: **N4SID algorithm for the identification of bilinear systems with white inputs. It consists of three steps namely an LQ-decomposition, an SV-decomposition and a final step where the state-space matrices are extracted. Although the estimates of A, B, C, D and N are consistent, the estimates of the noise covariance matrices Q, R, S will be consistent, only if the number of block rows in the data matrices W_p is infinite ($i = \infty$).**

Chapter 4

Subspace identification of bilinear systems with general inputs

In the previous chapter we have seen how a model of a bilinear system can be obtained using subspace identification. The algorithm assumed that the inputs of the system are white. Under that assumption, most of the results of linear subspace identification can be extended to bilinear systems. In the present chapter we will study an algorithm where this white input assumption is not necessary anymore. Although the resulting algorithm is a direct extension of the linear (Figure 2.7) and the bilinear subspace algorithm (Figure 3.3), it can only give unbiased estimates of the deterministic part of the system if the number of block rows in the generalized data Hankel matrices is infinite ($i \to \infty$).

4.1 Introduction

The problem we will be concerned with in this chapter is defined in Figure 4.1. It is the same as in the previous chapter, except that the input u_k is not assumed to be white any longer.

Problem formulation

Given *a large number of measurements of the input* u_k *and the output* y_k *generated by the unknown bilinear system:*

$$x_{k+1} = Ax_k + Nu_k \otimes x_k + Bu_k + w_k, \tag{4.1}$$
$$y_k = Cx_k + Du_k + v_k \tag{4.2}$$

with

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \geq 0, \tag{4.3}$$

determine *an estimate of the order* n*, the system matrices* A, B, C, D, N *up to within a similarity transformation and the noise covariance matrices* Q, S, R*.*

Figure 4.1: **The general bilinear subspace system identification problem.**

4.2 Bilinear subspace identification

In Chapter 2 and 3 we have seen that the first step of subspace identification, either for linear systems or bilinear systems with white inputs, mainly consists in isolating a rank-deficient term which is the product of the extended observability matrix Γ_i^l and a Kalman filter estimate $\widehat{X}_f$ of the state sequence X_f. This term is important because:

1. it can be easily isolated by making certain well chosen projections of the row space of matrices constructed out of measured input/output data only.

2. it is rank-deficient which means that, using an SVD, Γ_i^l and $\widehat{X}_f$ can be calculated.

3. from either Γ_i^l or $\widehat{X}_f$, the state-space matrices A, B, C, D, N can be easily found.

We have seen that the matrix input/output equations (2.28) for linear systems and (3.33) for bilinear systems were very critical to arrive at this result:

Linear: From the matrix input/output equation (2.28):

$$Y_f^l \quad = \quad \Gamma_i^l X_f + H_i^d U_f + H_i^s E_f$$

we can see that the oblique projection of the future outputs on the past inputs and outputs along the future inputs is indeed sufficient to isolate the term $\Gamma_i^l \widehat{X}_f$:

$$\begin{aligned} Y_f^l /_{U_f^l} W_p^l &= \Gamma_i^l \underbrace{X_f /_{U_f^l} W_p^l}_{=\widehat{X}_f} + H_i^d \underbrace{U_f^l /_{U_f^l} W_p^l}_{=0} + H_i^s \underbrace{E_f^l /_{U_f^l} W_p^l}_{=0} \\ &= \Gamma_i^l \widehat{X}_f . \end{aligned}$$

Bilinear with white inputs: In the case of bilinear systems with white inputs a similar projection was used. From the matrix input/output equation (3.33):

$$Y_f^l \quad = \quad \Gamma_i^l X_f + \Gamma_i^b \bar{U}_f^P \odot X_f + H_i^{d,b} U_f + H_i^{s,b} E_f .$$

it can be seen that, if the inputs are white and independent of the innovations e_k, the orthogonal projection of the future outputs Y_f^l onto the row space of the generalized inputs and output W_p equals:

$$\begin{aligned} Y_f^l / W_p &= \Gamma_i^l \underbrace{X_f / W_p}_{\widehat{X}_f} + \Gamma_i^b \underbrace{\bar{U}_f^P \odot X_f / W_p}_{=0} + H_i^{d,b} \underbrace{U_f / W_p}_{=0} + H_i^{s,b} \underbrace{E_f / W_p}_{=0} \\ &= \Gamma_f^l \widehat{X}_f . \end{aligned}$$

The second projection of the right hand side can be rewritten as:

$$\begin{aligned} (\bar{U}_f^P \odot X_f) / W_p &= (\bar{U}_f^P \odot X_f) W_p^T (W_p W_p^T)^{-1} W_p \\ &= \underbrace{\mathbf{E}[(\bar{u}_f^P \otimes x_f) w_p^T]}_{=0} \mathbf{E}^{-1}[w_p w_p^T]. \end{aligned}$$

Since u_f is white and since it only contains "future" values, it is uncorrelated with w_p.

In the case of bilinear systems with general inputs we will proceed in basically the same way, i.e. isolate a rank-deficient term by projecting the row space of generalized Hankel matrices generated from input and output data. Once again, our starting point will be the matrix input/output equations (3.29)-(3.34) of the previous chapter. Therefore we make the assumption that the bilinear state-space equations (4.1)-(4.3) can be rewritten in the forward innovation form (3.16)-(3.19) used in the previous chapter. There, due to the white input assumption, we could prove that (3.16)-(3.19) is indeed a forward innovation form. Here however, this is not the case anymore. However, for simplicity's sake and for analogy with the previous chapters, we will assume that (3.16)-(3.17) also hold for the bilinear system (4.1)-(4.2). Under that assumption we know that the matrix input/output equations (3.29)-(3.34) also hold.

When comparing the matrix input/output equation (2.28) of linear systems and the analogue relation for bilinear systems (3.33), we see that the main difference is the additional term $\Gamma_i^b U_f \odot X_f$. Therefore, for bilinear systems with general inputs one could suggest a projection similar to the one that is used in linear subspace identification, i.e. simply replacing the linear data Hankel matrices U_f^l and W_p^l by the generalized data Hankel matrices U_f and W_p defined in the previous chapter. However, as illustrated in the following, this would lead to a biased algorithm:

$$\begin{aligned} Y_f^l /_{U_f} W_p &= \Gamma_i^l \underbrace{X_f /_{U_f} W_p}_{\widehat{X}_f} + \Gamma_i^b \underbrace{(\bar{U}_f \odot X_f) /_{U_f} W_p}_{\neq 0} + H_i^{b,d} \underbrace{U_f /_{U_f} W_p}_{=0} + \\ &\quad H_i^{b,s} \underbrace{E_f /_{U_f} W_p}_{=0} \qquad (4.4) \\ &\neq \Gamma_i^l \widehat{X}_f . \end{aligned}$$

The problem is the second oblique projection of the right hand side which, if the inputs are not white, is not zero. As a consequence, the resulting projection is not rank-deficient and therefore does not allow for the extraction of Γ_i^l and $\widehat{X}_f$. In order to find a projection that eliminates the second term in the right hand side of (4.4), the knowledge of the row space of X_f is necessary. This seems like a vicious circle since X_f is exactly what we are trying to find by means of this projection.

One way to solve this problem is to look for a matrix, generated from the available input/output data, of which the row space contains an estimate of the state sequence X_f. In the case of linear systems and bilinear systems with white inputs it can be seen, from (2.34) and (3.38) respectively, that the row space of the estimated nonsteady-state Kalman filter state sequence $\widehat{X}_f$ is spanned by the row space of W_p^l in the linear case and W_p in the bilinear case. In the following theorem it will be proven that, if i is large enough, W_p is a good candidate for containing the state sequence X_f, also in the case of general inputs.

In the case of general inputs we also need a persistency of excitation condition which is different from the condition that is used in linear subspace identification and is in fact a generalization of the latter. This condition not only depends on the inputs but also on the outputs. One can therefore only talk about a persistently excited system and not only of a persistently exciting input.

Definition 4.2.1 (Persistently excited system)

The bilinear system (4.1)-(4.3) is persistently excited of order $2i$ *if the following matrix is of full row rank:*

$$\begin{pmatrix} U_{0|i-1} \\ U_{i|2i-1} \\ U_{i|2i-1} \odot W_{0|i-1} \end{pmatrix}.$$

Now the main theorem for subspace identification of bilinear systems with general inputs is presented.

Theorem 4.2.2 (Bilinear subspace identification [47])

Under the assumptions that:

- *the deterministic input is uncorrelated with the noise* e_k*,*
- *the bilinear system is persistently excited of order* $2i$ *(Definition 4.2.1),*
- *the number of available data points is large, so that* $i, j \rightarrow \infty$[1]*,*
- *the matrix* $A + Nu_k \otimes x_k - KC$ *is a stable matrix* $\forall k$*,*
- *the noise* e_k *is not identically zero,*

then we have that $\mathcal{O}_i$*, defined as:*

$$\mathcal{O}_i \stackrel{\text{def}}{=} Y_f^l / _{\begin{pmatrix} U_f \\ U_f \odot W_p \end{pmatrix}} W_p, \tag{4.5}$$

equals:

$$\mathcal{O}_i = \Gamma_i^l X_f$$

where Γ_i^l *is the reversed linear extended observability matrix (3.21) and* X_f *the state sequence.*

Proof: See Appendix A.4. ■

As for linear systems, the bilinear subspace identification problem can be interpreted as that of predicting the future output Y_f^l given the generalized past

[1]It should be noted that the projection (4.5), in the case that i is infinite, is in fact not defined.

inputs and outputs W_p, and the generalized future inputs U_f. With a predictor of the form:

$$\widehat{Y}_f^l = L_w W_p + L_{uw} U_f \odot W_p + L_u U_f \tag{4.6}$$

the least squares prediction can be found as the projection $Y_f^l / \binom{U_f}{U_f \odot W_p} W_p$, which can be calculated in a numerically efficient way with the following LQ-decomposition:

$$\begin{pmatrix} U_f \\ W_p \\ U_f \odot W_p \\ Y_f \end{pmatrix} \overset{\mathrm{LQ}}{=} \begin{pmatrix} L_{dd} & 0 & 0 & 0 \\ L_{cd} & L_{cc} & 0 & 0 \\ L_{bd} & L_{bc} & L_{bb} & 0 \\ L_{ad} & L_{ac} & L_{ab} & L_{aa} \end{pmatrix} \begin{pmatrix} Q_d^T \\ Q_c^T \\ Q_b^T \\ Q_a^T \end{pmatrix}. \tag{4.7}$$

Defining L as:

$$L \overset{\mathrm{def}}{=} \begin{pmatrix} L_{ad} & L_{ac} & L_{ab} \end{pmatrix} \begin{pmatrix} L_{dd} & 0 & 0 \\ L_{cd} & L_{cc} & 0 \\ L_{bd} & L_{bc} & L_{bb} \end{pmatrix}^{\dagger}$$

we have that:

$$\begin{aligned} L_u &= L(:, 1 : i_u), \\ L_w &= L(:, i_u + 1 : i_u + i_w), \\ L_{uw} &= L(:, end - i_u i_w + 1 : end) \end{aligned}$$

where i_u and i_w are the number of rows in U_f and W_p respectively. The projection $Y_f^l / \binom{U_f}{U_f \odot W_p} W_p$ can be written as:

$$Y_f^l / \binom{U_f}{U_f \odot W_p} W_p = L_w W_p.$$

4.3 Algorithm

The resulting bilinear subspace identification algorithm is given in Figure 4.2. One can also clearly distinguish the three typical steps of subspace identification namely the projection or LQ-decomposition, the SV-decomposition and finally the extraction of the state-space matrices.

Interesting to note is that this algorithm is a generalization of the linear subspace identification algorithm of Figure 2.7 and of the bilinear subspace identification algorithm for bilinear systems with white inputs of Figure 3.3. This means that, if the data comes from a linear system, the exact linear state-space matrices will be found:

N4SID for bilinear systems with general inputs

1. Calculate the oblique projections with an **LQ-decomposition**:

$$\mathcal{O}_i \overset{\text{LQ}}{=} Y^l_{i|2i-1} \Big/_{\begin{pmatrix} U_{i|2i-1} \\ U_{i|2i-1} \odot W_{0|i-1} \end{pmatrix}} W_{0|i-1} = L_w W_{0|i-1},$$

$$\mathcal{O}_{i+1} \overset{\text{LQ}}{=} Y^l_{i+1|2i-1} \Big/_{\begin{pmatrix} U_{i+1|2i-1} \\ U_{i+1|2i-1} \odot W_{0|i} \end{pmatrix}} W_{0|i}$$

and determine L_w.

2. Calculate the **SV-decomposition**:

$$W_1 L_w W_2 \overset{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

and

- determine an estimate of the order n of the system by inspecting the number of dominant singular values S_1,
- determine Γ^l_i and Γ^l_{i-1} as:

$$\Gamma^l_i = W_1^{-1} U_1 S_1^{1/2} \quad , \quad \Gamma^l_{i-1} = \Gamma^l_i(l+1 : end, :),$$

- determine the state sequences:

$$\widehat{X}_i = S_1^{1/2} V_1^T W_2^{-1} W_p \quad , \quad \widehat{X}_{i+1} = \Gamma^{l,\dagger}_{i-1} \mathcal{O}_{i+1}.$$

3. Calculate estimates of the system model A, B, C, D, N from the following linear least squares problem:

System model:

$$\begin{pmatrix} \widehat{X}_{i+1} \\ Y_{i|i} \end{pmatrix} = \begin{pmatrix} A & N & B \\ C & 0 & D \end{pmatrix} \begin{pmatrix} \widehat{X}_i \\ U_i \odot \widehat{X}_i \\ U_i \end{pmatrix} + \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}. \quad (4.8)$$

Figure 4.2: **N4SID algorithm for the subspace identification of general MIMO bilinear systems.**

Linear: As mentioned in Section 3.3.2 the data matrices used in bilinear subspace identification (3.22)-(3.23) are a generalization of the data block-Hankel matrices (2.19)-(2.20) used in linear subspace identification. They are a generalization in the sense that all the rows of the linear matrices are contained in the corresponding bilinear matrices. Although not occurring in the same order, the first i block rows of the linear data Hankel matrices are the same as the first i block rows of the corresponding bilinear data matrices. This comes down to omitting the product terms in the generalized data block-Hankel matrices. Therefore, when replacing the bilinear data matrices by their linear counterparts in the projection (4.5) and omitting the product term $U_f \odot W_p$, we find the projection (2.35) of linear subspace identification:

$$Y_f^l /_{\begin{pmatrix} U_f \\ U_f \odot W_p \end{pmatrix}} W_p \longrightarrow Y_f^l /_{U_f^l} W_p^l.$$

Bilinear with white inputs: The algorithm is also an extension of the results in the previous chapter, where the inputs were assumed to be white. In that case, $\binom{U_f}{U_f \odot W_p}$ is orthogonal to W_p, which means that the projection (4.5) reduces to (3.42):

$$Y_f^l /_{\begin{pmatrix} U_f \\ U_f \odot W_p \end{pmatrix}} W_p \longrightarrow Y_f^l / W_p.$$

Table 4.1 gives an overview of the projection used in the different subspace algorithms we have studied until now.

System class	Linear		Bilinear	
Input	General	White	General	White
Projection	$Y_f^l /_{U_f^l} W_p^l$	Y_f^l / W_p^l	$Y_f^l /_{\binom{U_f}{U_f \odot W_p}} W_p$	Y_f^l / W_p

Table 4.1: **Overview of the projection used in the different subspace algorithms studied in this thesis and classified according to the system class (linear/bilinear) and the type of input (general/white). One can see that the projections used in bilinear subspace identification are a generalization of the ones for linear subspace algorithms.**

i	# of rows in $Y_{0\|i-1}$
1	2
2	8
3	26
4	80

Table 4.2: **Number of rows d_i in the generalized data block-Hankel matrix $Y_{0|i-1}$ as a function of i: $d_i = \sum_{p=1}^{i}(m+1)^{p-1}l$.**

4.4 Implementation of bilinear subspace identification

In the Introduction of this thesis we mentioned that, due to the fact that the dimensions of the data matrices are an exponential function of the number of block rows i, the implementability of bilinear subspace identification is limited in practise. This can indeed be a problem, especially if i becomes large. For example, consider a bilinear system with 2 inputs and 2 outputs. Table 4.2 gives the number of rows in $Y_{0|i-1}$ as a function of i.

There are different ways to reduce the effect of this curse of dimensionality:

1. Where linear subspace identification assumes that $il > n$, with l the number of outputs and n the order of the system, here it is sufficient that i is such that $d_i = \sum_{p=1}^{i}(m+1)^{p-1}l > n$. For instance for a SISO bilinear system of order 12 we only need $i = 4$, since in that case $d_i = 15 > 12$. In the case of a SISO linear system of the same order, i would have to be at least 13. We see that in this way, the effect of the exponential growth of the matrices is reduced. A drawback however is that, since the index i will always have to be small (typically 3 or 4), the estimate of the noise covariances will not be very accurate. Indeed, as we have seen in the previous chapter, only for large values of i the Kalman filter will reach the steady-state and the bias on the estimate of the noise covariances Q, R, S will tend to zero.

2. One of the bottlenecks in subspace identification algorithms is the LQ-factorization. If one has to calculate the LQ-decomposition of very large matrices, one can easily reach the computer's memory capacity. In [108], [109] a block-processing method was proposed which allows to split up the problem into several smaller subproblems. If we are for instance interested in calculating the LQ-decomposition of an arbitrary matrix $A \in \mathbb{R}^{a \times j}$ where $j \gg a$, one can split up A into blocks as follows:

$$A = \begin{bmatrix} A(:,1:p) & A(:,p+1:2p) & \dots & A(:,j-p+1:j) \end{bmatrix}$$

where $p > a$, and chosen in such a way that the LQ-decomposition of every block separately fits into the memory of the computer. The resulting algorithm goes as follows:

(a) First calculate the LQ-decomposition of $A(:, 1 : p)$:

$$A(:, 1 : p) \quad = \quad L_1 Q_1^T$$

where $L_1 \in \mathbb{R}^{a \times a}$ is a lower triangular matrix and $Q_1 \in \mathbb{R}^{p \times a}$ such that $Q_1 Q_1^T = I_a$.

(b) Further calculate the LQ-decomposition of the concatenated matrix:

$$\begin{bmatrix} L_1 & A(:, p+1 : 2p) \end{bmatrix} \quad = \quad L_2 Q_2^T.$$

This step should be repeated for all blocks of A:

$$\begin{bmatrix} L_i & A(:, ip+1 : (i+1)p) \end{bmatrix} \quad = \quad L_{i+1} Q_{i+1}^T.$$

The L-component of the LQ-decomposition of A is then the last calculated L_i.

The main advantage of this method is that A never has to be entirely resident in the memory of the computer, but only blocks with $p \ll j$ columns.

3. Another way to increase the practical use of the bilinear subspace algorithms is to truncate the generalized data Hankel matrices in a certain way. By inspecting the structure of those matrices we see that to each row can be attributed a certain order, which is given by the number of multiplications with inputs u_k or outputs y_k that occur in each element of the row. For example an element $u_2 \otimes u_1 \otimes y_0$ is of order 3 because u and y occur 3 times. The order we just defined is in fact the order of the corresponding Volterra kernel in the input/output description (1.9). By examining the definitions (3.25) of the elements of $Y_{0|i-1}$ and $U_{0|i-1}$, we see that the maximum order that will occur in those data matrices is i and that all elements in the same row of $Y_{0|i-1}$ and $U_{0|i-1}$ have the same order. A reasonable way to truncate the data matrices could then be to eliminate all the rows with an order higher than a given value [19]. This is in fact the same as truncating the input/output description of the system to the Volterra kernels of a certain order. The truncated matrices will be denoted by Y_f^{β} where the truncation index β is the maximum order that is retained in the generalized data Hankel matrices. For example if we want to truncate $Y_{0|2}$ to the terms of order 1:

$$Y_{0|2} = \begin{pmatrix} Y_2 \\ Y_1 \\ Y_0 \\ \hline U_1 \odot Y_0 \\ U_2 \odot Y_1 \\ U_2 \odot Y_0 \\ \hline U_2 \odot U_1 \odot Y_0 \end{pmatrix} \xrightarrow{\mathbf{2}} Y_{0|2}^{\mathbf{2}} = \begin{pmatrix} Y_2 \\ Y_1 \\ Y_0 \\ \hline U_1 \odot Y_0 \\ U_2 \odot Y_1 \\ U_2 \odot Y_0 \end{pmatrix} \xrightarrow{\mathbf{1}} Y_{0|2}^{\mathbf{1}} = \begin{pmatrix} Y_2 \\ Y_1 \\ Y_0 \end{pmatrix}.$$

Interesting to note is that, if the matrices are truncated to order 1, we find the corresponding linear data matrices: $Y_f^l = Y_f^{\mathbf{1}}$, $U_f^l = U_f^{\mathbf{1}}$ and $W_p^l = W_p^{\mathbf{1}}$. Therefore, by truncating the data matrices to order 1, we find the projection of the linear subspace identification algorithm:

$$Y_f^l / _{\begin{pmatrix} U_f^{\mathbf{1}} \\ (U_f \odot W_p)^{\mathbf{1}} \end{pmatrix}} W_p^{\mathbf{1}} = Y_f^l /_{U_f^l} W_p^l.$$

4. Even if the data matrices are truncated up to a certain order, the dimensions can still become quite large, especially for high-order MIMO systems. Although from an analytical point of view our approach is correct, the fundamental reason why such large data matrices are needed in the subspace identification algorithm is the way these data matrices are constructed. Up till now, we have seen that the rows of the data matrices contain the inputs and outputs of the systems, as well as certain polynomial expansions of these inputs and outputs. The idea we put forward here is, instead of constructing the rows of the data matrices as *polynomial* expansions of the inputs and the outputs, one could instead use radial basis functions to transform the data in a nonlinear way. It is a well-known fact that, in this respect, radial basis functions are much more efficient than polynomial expansions since the number of terms needed to approximate an arbitrary nonlinear function is much smaller [13].

   Whereas in Theorem 4.2.2 we have seen that an estimate of the state sequence X_f can be found as a linear combination of the rows of W_p, where W_p is constructed from the input/output data and certain polynomial expansions of these inputs and outputs, we will now assume that the row space of the state sequence X_f is spanned by the row space of nonlinear transformations $\Phi_i(W_p)$ of W_p:

$$\begin{aligned} X_f &= \sum_{k=1}^{i} \alpha_k \Phi_k(W_p) \\ &= \alpha \Phi_{1|i}(W_p). \end{aligned}$$

   where $\alpha = \begin{pmatrix} \alpha_1 & \dots & \alpha_i \end{pmatrix}$ and $\Phi_{1|i} = \begin{pmatrix} \Phi_1^T & \dots & \Phi_i^T \end{pmatrix}^T$. Although the nonlinear functions Φ_k can be chosen arbitrarily, it is a well-known fact that radial basis functions Φ_k are very well suited to approximate nonlinear functions. They are described by:

$$\Phi_k(x) = \exp\left((x - c_k)^T \Sigma_k^{-1} (x - c_k)\right).$$

   The two parameters that characterize a radial basis function are therefore its center c and its width Σ. This leads us to the main reason why we consider radial basis functions here, instead of any other nonlinear basis function. Indeed, once the centers and the widths of the radial basis functions are fixed, the problem is linear in the parameters $\alpha_1, ..., \alpha_i$.

The centers c_k and the widths Σ_k can be chosen by using clustering techniques, such as the classical k-means algorithm. The procedure goes as follows: first i initial centers c are chosen in the data space, for instance i arbitrary columns of the W_p^l matrix. In a second step, the position of these centers is then adjusted with respect to the other data points. This can for instance be done by minimizing the sum of the distances of each data point x_i to the closest center c_p:

$$\min_{c_1,\dots,c_i} \sum_{i=1}^{j} ||x_i - c_p||^2.$$

The widths Σ_k are then determined as a function of the distances between the centers.

Since we consider it as important that this method is a generalization of linear subspace methods, we also use the non-transformed linear data Hankel matrix W_p^l to estimate the state sequence. Therefore we have that:

$$\widehat{X}_f = \alpha^l W_p^l + \alpha^b \Phi_{1|i}(W_p).$$

The resulting algorithm is very similar to the one presented in Figure 4.2. The only difference is the projection step where the matrix $\left(\begin{smallmatrix} W_p^l \\ \Phi_{1|i}(W_p^l) \end{smallmatrix}\right)$ is used instead of the matrix W_p.

In the previous chapter, the main assumption we made was that the inputs were assumed to be white and independent of the noise. Although this is a common situation in practise, it can be quite restrictive in certain cases. In the present chapter we have studied an algorithm where this white input assumption is not necessary anymore. Although the resulting algorithm is a direct extension of the linear (Figure 2.7) and the bilinear subspace algorithm (Figure 3.3), it can only give unbiased estimates of the deterministic part of the system matrices. A necessary condition for this to be true is that the number of block rows in the generalized data Hankel matrices should be infinite ($i \to \infty$).

Chapter 5

Some examples

In this chapter we make a direct comparison between linear and the newly presented bilinear subspace identification algorithms. For simple academic examples as well as real-life processes, we will analyze how the precision of a model obtained from bilinear subspace identification algorithms compare with that of a linear model. Both the subspace algorithms for bilinear systems with white inputs (Chapter 3) and the algorithms for bilinear systems with general inputs (Chapter 4) are studied.

5.1 White inputs

5.1.1 Second order system

In this example we show how a simple second order bilinear system with 2 inputs and 2 outputs can be identified using the algorithm of Figure 3.3. The system matrices are:

$$A = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.3 \end{pmatrix}, \quad N_1 = \begin{pmatrix} 0.6 & 0 \\ 0 & 0.4 \end{pmatrix}, \quad N_2 = \begin{pmatrix} 0.2 & 0 \\ 0 & 0.5 \end{pmatrix},$$
$$B = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \qquad C = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \qquad D = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and the noise covariance matrices

$$Q = \begin{pmatrix} 0.01 & 0 \\ 0 & 0.01 \end{pmatrix}, \quad R = \begin{pmatrix} 0.01 & 0 \\ 0 & 0.01 \end{pmatrix}, \quad S = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

The input signal we used is a pseudo random binary sequence (PRBS) of length 4095 and amplitude 1. In order to have an idea of the order of the system, one should examine the singular values of $\mathcal{O}_i = Y_{i|2i-1}/\binom{Y_{0|i-1}}{U_{0|i-1}}$ (see Figure 5.1). From this figure, we choose a second order bilinear model ($n = 2$), i.e. the

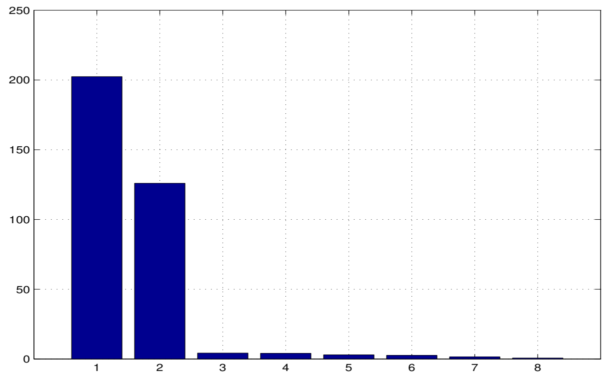


Figure 5.1: **Singular values of $\mathcal{O}_i$ with $i = 2$ and $j = 4095$.**

order of the original system. The identified model is validated by comparing the eigenvalues of A, N_1 and N_2 of the model with those of the original system. From Table 5.1 one can see that these eigenvalues are very close. If the number of available data points we use were infinite ($j = \infty$), they would be identical. As mentioned in Section 3.4.3, the estimate of the noise covariance matrices is

	Original system	Model
A	0.5, 0.3	0.5001, 0.2979
N_1	0.6, 0.4	0.4020, 0.5994
N_2	0.2, 0.5	0.1914, 0.5016

Table 5.1: **Eigenvalues of the A and the N matrices. Left: original system, right: model.**

biased since the i-parameter is small.

5.1.2 Binary distillation column

It is well known that distillation columns are very important industrial processes. However, due to their strongly nonlinear behavior, internal couplings, time delays, etc... they are particularly difficult to control. Therefore, it is of extreme importance to have a good model.

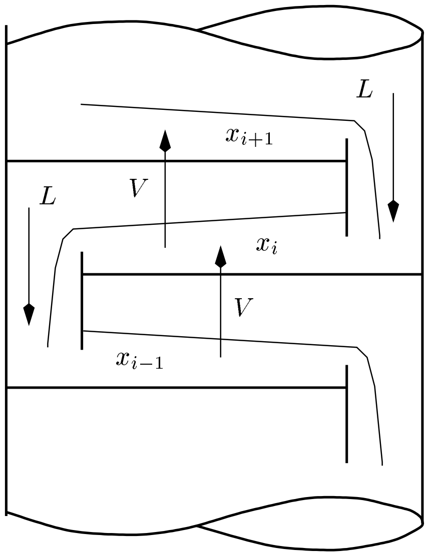


Figure 5.2: **Cross section of a tray in a distillation column. The arrows indicate the flows entering and leaving the tray. V is the vapor and L the liquid flow going down the column. The concentrations in the liquid phase on tray $i-1$, i and $i+1$ are x_{i-1}, x_i and x_{i+1} respectively.**

In this example we make a model of a two component distillation column on the basis of data obtained from a full-order nonlinear white box model. We show that a bilinear model can approximate the dynamics of the system in a better way than can be done with a linear model.

By analyzing the equations describing the physical behavior of a distillation column, the bilinear nature of the process is quite easy to show. A cross section of a tray in a distillation column is represented in Figure 5.2. In this case, we consider a binary distillation column (i.e. where two components have to separated). The partial mass balance of an arbitrary tray i of the column can be described by the following equation:

$$M\frac{dx_i}{dt} = Vy_{i-1} + Lx_{i+1} - Vy_i - Lx_i$$

where V is the gas flow going up in the column, L the liquid flow going down and M the mass of the liquid holdup on a tray. The concentration of the gas phase y_i, in thermodynamic equilibrium with the liquid phase, can be well approximated around an operating point by a linear function of the concentration

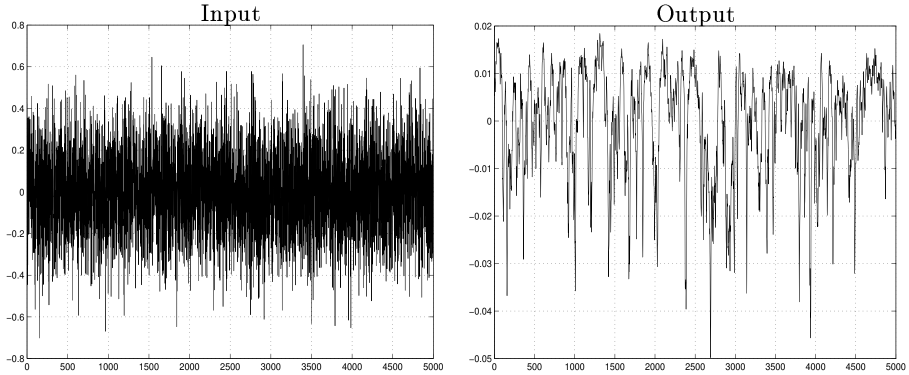


Figure 5.3: **Left: the input is a white signal on the reflux rate. Right: the output x_d is the variation on the distillate concentration (detrended).**

in the liquid phase x_i:

$$y_i = K_i x_i.$$

In that case the partial mass balance becomes:

$$\frac{dx_i}{dt} = \frac{K_i}{M} V x_{i-1} + \frac{1}{M} L x_{i+1} - \frac{K_i}{M} V x_i - \frac{1}{M} L x_i$$

which is a bilinear equation since the flows L and V are controlled variables and the concentrations on the different trays x_{i-1}, x_i and x_{i+1} are the states of the system.

Let us now make a direct comparison between linear and bilinear subspace identification. Using the same input/output data set we show that a bilinear model, obtained with the bilinear subspace identification algorithm of Figure 3.3, is more suited to approximate the dynamics of the distillation column than a model obtained with linear subspace methods.

The input we consider is the reflux rate of the column and is excited with a white signal, while the output is the purity x_d of the distillate collected at the top of the column. The available input/output data set is given in Figure 5.3. In order to make a correct comparison between the methods, both models should have approximately the same number of parameters. Since a bilinear model of a given order has more parameters than a linear model of the same order (due to the N-matrix), we choose a linear model of order 6 and a bilinear model of order 3. The real output is compared with the output simulated with the model. One can see that the 3th order bilinear model performs better than the 6th order linear model (Figure 5.4).

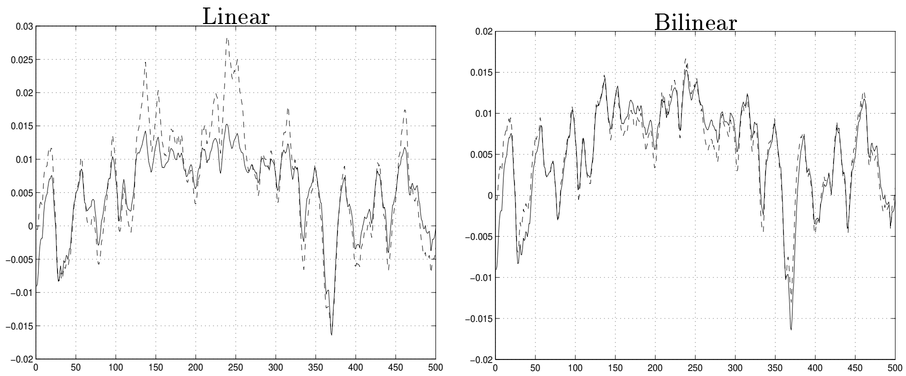


Figure 5.4: **The real output (full line) is compared with the output of the model (dashed line). Before identification, the mean value of the data was set to zero. Left: 6th order linear model. Right: 3th order bilinear model. The fit is clearly better with a bilinear model.**

5.2 General inputs

5.2.1 A bilinear MIMO system

Let us now consider the following 3th order deterministic bilinear system with 2 inputs and 2 outputs:

$$A = \begin{pmatrix} 0.9401 & 0.8612 & -0.3521 \\ 0.1127 & -0.2771 & -0.3007 \\ 0.0724 & -0.4536 & 0.3510 \end{pmatrix}, \quad B = \begin{pmatrix} -0.6515 & 0.1125 \\ -0.2860 & -0.1795 \\ -0.9473 & 0.1738 \end{pmatrix},$$

$$C = \begin{pmatrix} 0.6024 & -0.4619 & 0.4466 \\ -0.5745 & 1.0956 & 1.0746 \end{pmatrix}, \quad D = \begin{pmatrix} -1.7712 & 0.0771 \\ 0.8564 & 0.3883 \end{pmatrix},$$

$$N = \begin{pmatrix} -0.0269 & -0.1350 & -0.5859 & -0.9459 & 0.0005 & 0.2847 \\ 0.5049 & -0.5079 & 1.2083 & -0.4895 & -0.4791 & -0.6696 \\ 0.4183 & -0.7500 & 0.5602 & 0.0993 & 0.0132 & 0.4354 \end{pmatrix}.$$

The input which is used to excite the system is a low-pass filtered white noise (Figure 5.5). Now we use this data set to determine a linear model on the one hand, and a bilinear model on the other. Afterwards, both models are compared on the basis of a validation data set, different from the identification data set. In order to have approximately the same number of parameters in both models, we choose a linear model of order 5 and a bilinear model of order 3.

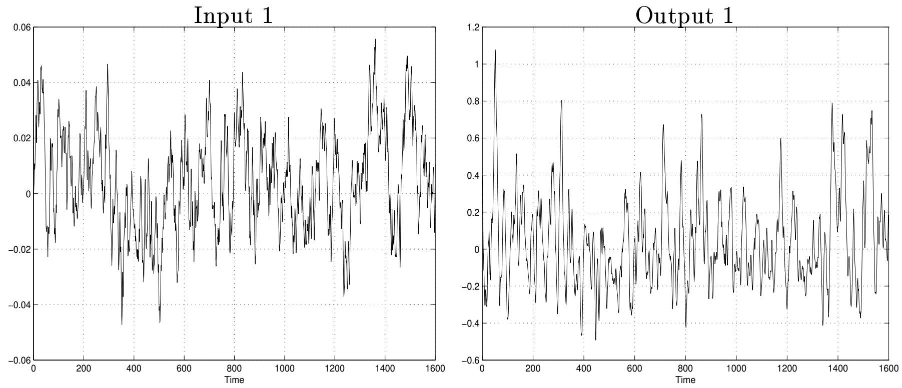


Figure 5.5: **Data sets: input 1 (left) and output 1 (right) of the system. The first 1000 data points are used for identification while the next 500 for validation.**

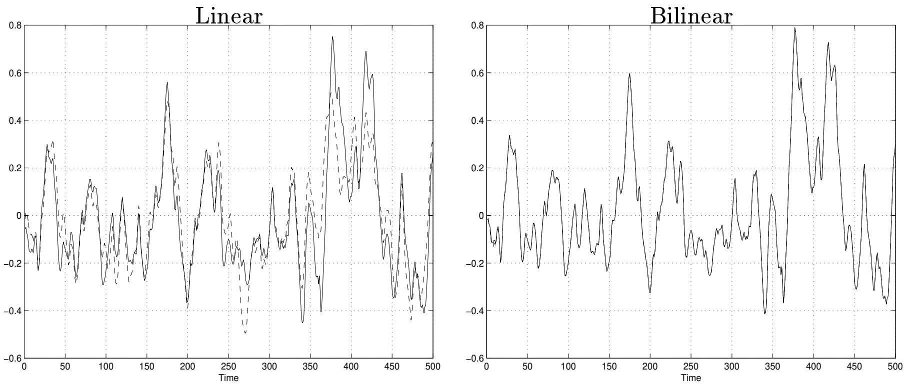


Figure 5.6: **This figure compares the output of the linear model with that of the bilinear model. The order of the bilinear model is that of the original system (3) while for the linear model we take a higher order (5). The full line is the real output and the dashed line the output of the model. Left: linear model, right: bilinear model. The bilinear model clearly gives the best results.**

5.2.2 Distillation column

In this example we reconsider the distillation example of Section 5.1.2. Here however, the input is a low-pass filtered white noise. The available input/output data set is given in Figure 5.7. The same data set was used in both cases but a 5th order linear model was chosen and a 3th order bilinear model. Once again, the real output is compared with the output simulated with the model. The

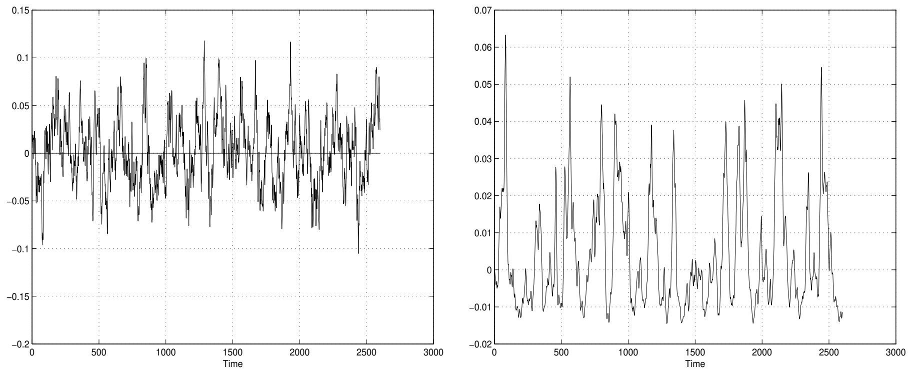


Figure 5.7: **Left: the input is the flow leaving the column at the base. Right: the output is the concentration of the base flow. The first 2000 points are used for identification, the next 500 for validation.**

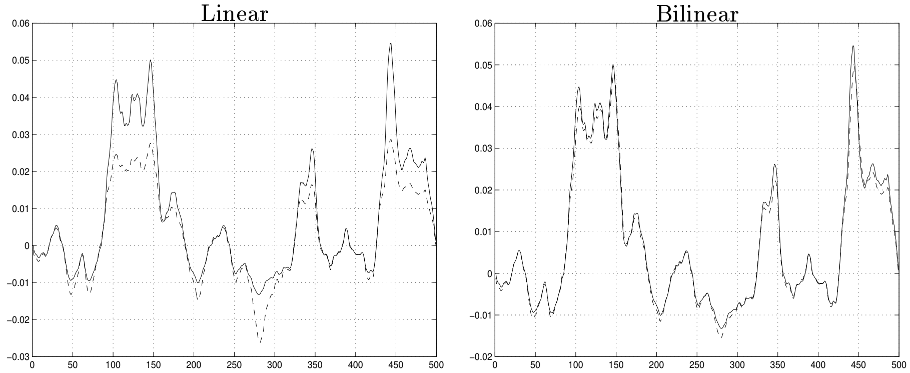


Figure 5.8: **The real output (full line) is compared with the output of the model (dashed line). Left: linear model of order 5, right: bilinear model of order 3. The bilinear model is clearly better than the linear model.**

results are depicted in Figure 5.8.

5.2.3 Linear vs. bilinear N4SID

Let us now make a final comparison between linear and bilinear subspace identification, by applying both methods to a set of measurements obtained from different real-life systems: measurements from a ball and beam experimental setup, a dryer, a flexible mechanical structure, an industrial liquid saturated

	MSE_{lin}	MSE_{bil}	n_{lin}	n_{bil}	β	j_{id}	j_{val}	m	l
B&B	1.0328	0.6536	1	1	2	500	500	1	1
Dryer	0.0246	0.0227	6	5	1	700	250	1	1
Flex. Str.	0.3056	0.3043	3	3	1	800	200	1	1
Exchanger	0.0667	0.0633	8	4	2	2000	1000	1	1
Winding	0.0957	0.0317	8	3	1	1500	1000	4	1

Table 5.2: **Mean square errors (MSE) for the validation data sets. n is the model order that was chosen (linear/bilinear), j_{id} and j_{val} the number of data points for the identification and the validation respectively, m and l the number of inputs and outputs. β is the truncation index for the data matrices used in the bilinear subspace algorithm (see Section 4.4). For both the linear and the bilinear method, different values for the system order n were examined (going from 1 to 10). For all experiments, the i-parameter was taken equal to the system order n. The conclusion of these simulations is that, especially for the systems with a strong nonlinear character, such as the ball & beam and the winding process, the bilinear subspace algorithm gives better results than the linear one.**

heat exchanger and a laboratory setup of a winding process. In order to allow a further comparison with other methods, all these data sets, together with some references to papers where they have been used, have been made freely available on the DAISY internet site [30]. By performing the same data pre-processing and using the same identification and validation sets, we can make a direct comparison between the precision of linear subspace identification (Algorithm p. 131 in [123]) and the bilinear subspace algorithm of Figure 4.2. The comparison is made on the basis of the MSE of the validation set:

$$MSE = \frac{1}{j_{val}} \sum_{k=1}^{N} (y_k^s - y_k)^2$$

where j_{val} is the size of the validation set and y_k^s the simulated output. All the data sets were pre-processed in the same way: detrending (i.e. mean value was set to zero) and scaling such that both the inputs and outputs have a covariance matrix equal to the unity matrix. The results are given in Table 5.2.

In this chapter we have made a direct comparison between linear and the newly presented bilinear subspace identification algorithms. For simple academic examples as well as real-life processes, we have analyzed how the precision of a model obtained from bilinear subspace identification algorithms compares with that of a linear model. Both the subspace algorithms for bilinear systems with white inputs (Chapter 3) and the algorithms for bilinear systems with general inputs (Chapter 4) have been studied.

Part II

Subspace-based control

Chapter 6

Model-free subspace-based LQG-design

In this chapter we present a new method, based on subspace identification techniques, to design an LQG-controller for unknown linear systems. We start by defining the model-free LQG-problem and split it up into three separate subproblems namely the system identification, the Kalman filter design and the LQ-controller design. Further we recall the classical solution of this LQG-problem on the one hand and present the new subspace-based way of solving it. Finally, the conditions under which both methods are equivalent are analyzed.

6.1 The model-free LQG-control problem

The class of systems we consider in this chapter are linear time-variant systems that can be described by the following state-space equations:

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{6.1}$$

$$y_k = Cx_k + Du_k + v_k \tag{6.2}$$

with

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq} \geq 0. \tag{6.3}$$

The problem we will tackle is a mixed control and identification problem. Starting from a set of measurements of the unknown system (6.1)-(6.3) one has to calculate an LQG-controller for this system. Before giving a formal definition

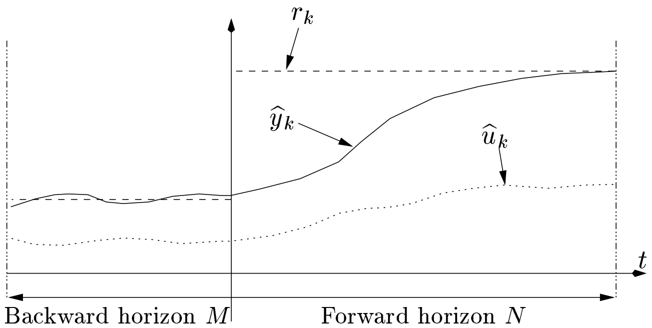


Figure 6.1: **This figure illustrates the principle of finite horizon linear quadratic Gaussian control. The aim is to find the future control sequence $\widehat{u}_k$ that minimizes a certain performance criterion J. r_k is the future reference output trajectory. $\widehat{y}_k$ finally, is the optimal prediction of the future outputs given the future inputs and the past inputs and outputs.**

of this so-called *model-free LQG-control problem*, we start by refreshing some notions of finite receding horizon LQG-control (Figure 6.1). The forward horizon N is the number of time steps over which the future control inputs are calculated (control horizon) and the output is predicted (prediction horizon). The backward horizon M is the number of data points that is used to predict the output and calculate the control input over the forward horizon. In LQG-control, the goal is to minimize the following quadratic performance criterion J:

$$J = \sum_{k=1}^{N} \{(\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k\} \tag{6.4}$$

where $\widehat{y}_k$ is the k-step-ahead optimal predicted output. The reference output trajectory is described by r_k and the matrices $Q_k \in \mathbb{R}^{l \times l}$ and $R_k \in \mathbb{R}^{m \times m}$ are user-defined non-negative definite weighings of the outputs and the inputs. The first term in this cost function penalizes the fact that the predicted outputs $\widehat{y}_k$ are different from the reference outputs r_k, while the second term keeps the energy in the inputs $\widehat{u}_k$ at a reasonable level. We will consider a receding horizon which means that only the first control input $\widehat{u}_1$ is implemented. At the next time step, the whole procedure is repeated.

The problem we are concerned with in the present chapter is described in Figure 6.2. The classical way to solve it is to split it up into different subproblems. The three typical subproblems one will encounter are:

System identification: Given measurements of the inputs u_k and the outputs y_k of the unknown system (6.1)-(6.3), one has to find an estimate

Model-free LQG-Control Problem

Given *a set of open-loop measurements of the inputs* $\widehat{u}_k$ *and the outputs* $y_k, k < 0$, *of the unknown system (6.1)-(6.3),* **find** *the input sequence* $\widehat{u}_f = (\widehat{u}_1, \dots, \widehat{u}_N)$ *that minimizes the quadratic cost function* J *over the forward horizon* N*:*

$$J = \sum_{k=1}^{N} \left\{ (\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k \right\}.$$

Figure 6.2: **Model-free LQG-control problem.**

of the system matrices A, B, C, D and the noise covariance matrices Q, R and S.

Kalman filter design: Since LQG is a state-space based control method, the knowledge of the state is mandatory. The second step of LQG-design then consists in finding an estimate of the state of the system, given the system parameters and measurements of the past inputs and outputs.

LQR-controller design: Once the Kalman filter estimate of the state is known, it is used to design a linear quadratic controller.

The reason for making this split up is either historical and for the sake of simplicity. It indeed seems more simple to consider one subproblem, rather than being ambitious and tackle the whole problem at once. This is also reflected in the literature, namely the problem as indicated in Figure 6.2 is hardly ever considered as such. The well-known separation principle in LQG-design is a typical example of this. It consists in splitting up the LQG-problem, given a model of the plant, in a Kalman filter and a LQ-controller. Most of the results in the literature focus on one of the subproblems for instance the system identification or the LQ-controller design.

The idea of computing an optimal LQG-controller directly from data, without any use of a model, has been developed by Hjalmarsson and collaborators (see [69], [68]) using a technique called Iterative Feedback Tuning (IFT). In [69], [68] a parameterized controller structure is used and implemented on the actual plant. Its parameters are optimized *iteratively* by collecting data on the closed-loop system, with the present controller active, and by computing an unbiased estimate of the gradient of the global optimization problem from these closed-loop data. Under weak conditions, this iterative scheme is shown to converge to at least a local minimum of the resulting nonlinear optimization problem.

In the present chapter, we introduce a new method, based on techniques of linear subspace identification (Chapter 2), to design an LQG-controller of an unknown system directly from the measured input and output data. The method does not follow the typical three-step method described above i.e. no explicit system identification is done since the state-space matrices are never calculated. Moreover, no Kalman filter equations nor LQR-equations have to be solved. The method has some very clear parallels with the subspace algorithms of Chapter 2 in the sense that first an LQ-decomposition of matrices constructed out of the available input/output data is calculated, followed by a singular value decomposition. This is the main idea of the present chapter (See also Figure 1.3).

The IFT-algorithms are mainly based on results from the field of prediction error system identification methods, and are therefore entirely different from the subspace-based methods developed here. In some ways, the difference between the IFT-methods and the presently discussed subspace-based LQG-algorithms, parallels the fundamental difference between parametric identification based on criterion minimization, and subspace identification based on projections and singular value decomposition. The advantages and drawbacks of the two identification methods in terms of optimality, parametric versus non-parametric, local minima versus uniqueness of the solution, etc... are transported from the realm of identification to the realm of control design.

6.2 LQG - The classical approach

The classical way of solving the LQG-problem, as we have seen in the previous section, is to split it up into three separate subproblems: system identification, Kalman filter design and LQ-controller design [87], [17]. In order to allow a better comparison with the new subspace-based LQG-algorithm of the next section, we shortly recall the classical solution to these three steps. For simplicity we assume that the output reference trajectory r_k is zero. We will come back to the more general case of arbitrary output references in Section 6.4 (Remark 6.4.3.

System Identification: It would lead us much too far only to give an overview of all the different system identification methods that exist in the literature (See also Section 1.1). Therefore we will simply assume that there is an infinite amount of data available, and that the state-space matrices A, B, C, D, Q, R, S are obtained with a consistent system identification method. This basically comes down to assuming that the state-space matrices are exactly known.

Kalman filter:

Given

- *the state-space matrices* A, B, C, D, Q, R *and* S,
- *the last* M *measurements of the inputs* u_k *and outputs* y_k, $k \in \{-M+1, \dots, 0\}$,
- *an initial state estimate* $\widehat{x}_{-M+1}$ *of the Kalman filter,*

then the optimal one-step-ahead state prediction $\widehat{x}_1$ *is the solution to the following recursive Kalman filter equations:*

$$\begin{aligned} \widehat{x}_{q+1} &= A\widehat{x}_q + Bu_q + K(y_q - C\widehat{x}_q - Du_q), && (6.5) \\ K &= (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}, && (6.6) \\ \Sigma^s &= A\Sigma^s A^T + (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1} \\ &\qquad (G - A\Sigma^s C^T)^T, && (6.7) \end{aligned}$$

LQ-controller:
Given

- *the state-space matrices* A, B, C *and* D *and the user-defined weighting matrices* Q_p *and* R_p, $p \in \{1, \dots, N\}$,
- *the optimal state estimate* $\widehat{x}_1$,
- *the initial condition* $P_0 = 0$ *for the LQ-controller Riccati equation,*

then the first input $\widehat{u}_1$ *of the input sequence that minimizes the performance criterion (6.4) is the solution to the following recursive LQ-controller equations:*

$$\begin{aligned} \widehat{u}_1 &= -L_c\widehat{x}_1, && (6.8) \\ L_c &= (R_1 + D^T Q_1 D + B^T P_N B)^{-1}(D^T Q_1 C + B^T P_N A), && (6.9) \\ P_{q+1} &= A^T P_q A + C^T Q_{N-q} C - (C^T Q_{N-q} D + A^T P_q B) \\ &(R_{N-q} + D^T Q_{N-q} D + B^T P_q B)^{-1}(C^T Q_{N-q} D + A^T P_q B)^T. && (6.10) \end{aligned}$$

A more graphical interpretation of the Kalman filter and the LQ-controller, of the above described LQG-controller, is given in Figure 6.3.

6.3 LQG - A subspace approach

In this subsection, we study the model-free LQG-problem (Figure 6.2) from another point of view. We use the subspace techniques from Chapter 2 to calculate an LQG-controller of the unknown system (6.1)-(6.3), directly from the measured inputs and output of that system.

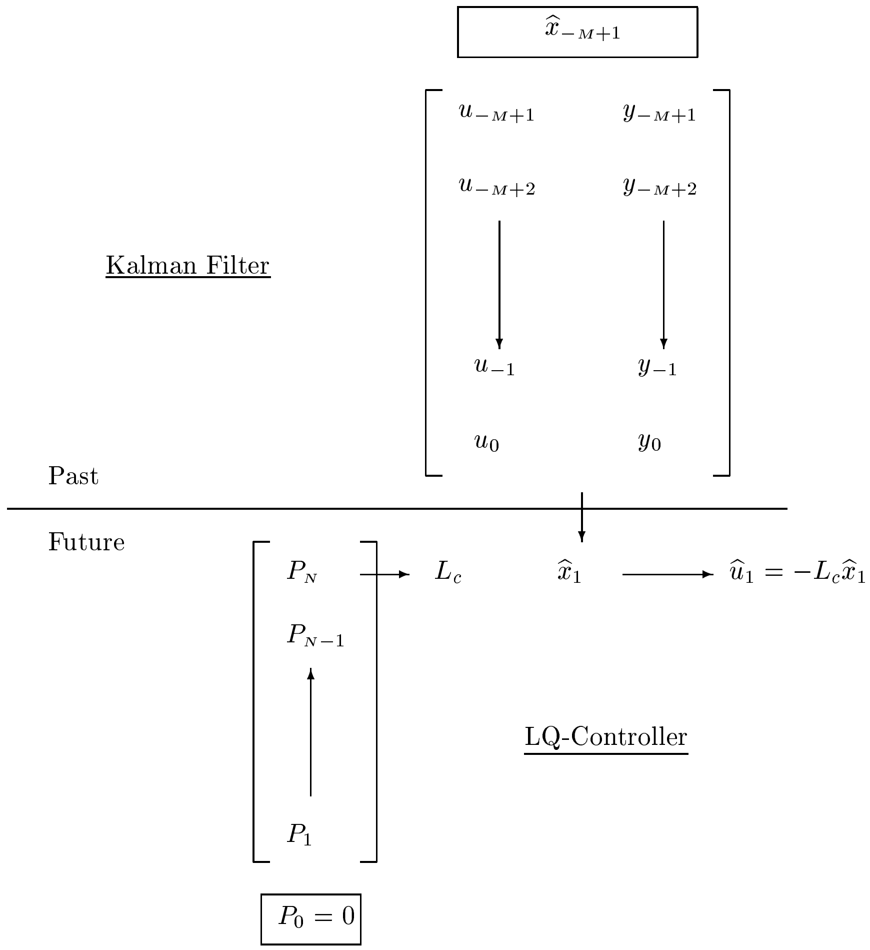


Figure 6.3: **Graphical interpretation of an LQG-controller. The last implemented control input and measured output are u_0 and y_0. The aim of the LQG-controller is to find the control input $\widehat{u}_1$ that minimizes the performance criterion (6.4). The horizontal line represents the separation between the past and the future i.e. the Kalman filter and the LQ-controller. Together with the system matrices A, B, C, D, Q, R, S, the Kalman filter uses the M last known values of the inputs u_k and the outputs y_k to determine the one-step-ahead predicted Kalman filter state $\widehat{x}_1$. In order to initialize the Kalman filter equations (6.5)-(6.7), an initial state estimate $\widehat{x}_{-M+1}$ is required. The solution of the LQ-controller (6.8)-(6.10) is the control input $\widehat{u}_1$ and requires the optimal one-step-ahead predicted state estimate $\widehat{x}_1$, which is provided by the Kalman filter, and P_0 which is the initial condition for the LQ-controller Riccati equation. The final LQG-controller output $\widehat{u}_1$ is then the LQ-controller gain L_c, multiplied by the one-step-ahead predicted Kalman filter state $\widehat{x}_1$.**

What inspired us to make the link between subspace identification and LQG-design is the following idea. As illustrated in Figure 6.3, state-space based controllers use the state as an intermediate variable between the backward horizon (past) and the forward horizon (future) to find the control input. Indeed, first one calculates the state estimate that minimizes the Kalman filter criterion, given past inputs and outputs. In a second step the inputs that minimize the LQ-controller criterion, given the state and the reference output, are calculated. This remarkable duality between the Kalman filter (6.5)-(6.7) and the LQ-controller (6.8)-(6.10) was already pointed out in [87]. Although not as explicit as in classical LQG-design, the above idea is also clearly present in subspace system identification. There the link between Kalman filtering and the projection of data matrices on the one hand, and the idea of past and future on the other, are well known (See also Section 2.4). The interesting thing is that in subspace identification the Kalman filter states can be found directly from the available input/output data, i.e. without the knowledge of the system parameters and thus without having to solve the Kalman filter equations (6.5)-(6.7). This property will be heavily exploited in the rest of the chapter.

Instead of minimizing the performance criterion (6.4) in the classical way, we start from the input/output equation (2.28). First we define the future input sequence $\widehat{u}_f$ and the corresponding optimal forward ahead predicted output sequence $\widehat{y}_f$ as:

$$\widehat{y}_f \stackrel{\text{def}}{=} \begin{pmatrix} \widehat{y}_1 \\ \widehat{y}_2 \\ \vdots \\ \widehat{y}_N \end{pmatrix}, \qquad \widehat{u}_f \stackrel{\text{def}}{=} \begin{pmatrix} \widehat{u}_1 \\ \widehat{u}_2 \\ \vdots \\ \widehat{u}_N \end{pmatrix}.$$

With these definitions, and assuming for the moment that the reference output $r_k = 0$, the optimal control criterion (6.4) can be written as:

$$J = \widehat{y}_f^T \bar{Q} \widehat{y}_f + \widehat{u}_f^T \bar{R} \widehat{u}_f$$

where the block-diagonal matrices $\bar{Q} \in \mathbb{R}^{Nl \times Nl}$ and $\bar{R} \in \mathbb{R}^{Nm \times Nm}$ are defined as:

$$\bar{Q} \stackrel{\text{def}}{=} \begin{pmatrix} Q_1 & 0 & \dots & 0 \\ 0 & Q_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Q_N \end{pmatrix}, \qquad \bar{R} \stackrel{\text{def}}{=} \begin{pmatrix} R_1 & 0 & \dots & 0 \\ 0 & R_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R_N \end{pmatrix}.$$

From (2.30) we know that the least squares estimate of the output sequence $\widehat{y}_f$ can be expressed as:

$$\widehat{y}_f = L_w w_p + L_u \widehat{u}_f$$

where

$$w_p = \begin{pmatrix} y_p \\ u_p \end{pmatrix}$$

with $u_p \in \mathbb{R}^{Mm}$ and $y_p \in \mathbb{R}^{Ml}$ the M last measured values of the inputs and the outputs:

$$y_p \stackrel{\text{def}}{=} \begin{pmatrix} y_{-M+1} \\ \vdots \\ y_{-1} \\ y_0 \end{pmatrix}, \qquad u_p \stackrel{\text{def}}{=} \begin{pmatrix} u_{-M+1} \\ \vdots \\ u_{-1} \\ u_0 \end{pmatrix}.$$

We have seen in Section 2.5 that the parameters L_w and L_u can be obtained from the first two steps of the subspace identification algorithm (Figure 2.7), i.e. an LQ- and an SV-decomposition. The performance criterion J can now be written as:

$$J = (L_w w_p + L_u \widehat{u}_f)^T \bar{Q} (L_w w_p + L_u \widehat{u}_f) + \widehat{u}_f^T \bar{R} \widehat{u}_f$$

which is a problem that can be easily solved by putting the trace of the derivative of J with respect to the input sequence $\widehat{u}_f$ to zero:

$$\mathrm{tr} \frac{\partial J}{\partial \widehat{u}_f} = 0.$$

The subspace-based LQG control sequence then becomes:

$$\widehat{u}_f = -(\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q} L_w w_p. \tag{6.11}$$

Since we adopt a receding horizon, only the first element $\widehat{u}_1$ of $\widehat{u}_f$ will be implemented:

$$\widehat{u}_1 = -\bar{L}_w(1:m,:) w_p \tag{6.12}$$

where $\bar{L}_w \stackrel{\text{def}}{=} (\bar{R} + L_u^T \bar{Q} L_u^T)^{-1} L_u^T \bar{Q} L_w$. This controller equation can also be written in a general way as:

$$u_k = -s_1 u_{k-1} - \ldots - s_M u_{k-M} + r_1 y_{k-1} + \ldots + r_M y_{k-M} \tag{6.13}$$

where the coefficients r_i and s_i are defined as:

$$\begin{aligned} r_i &\stackrel{\text{def}}{=} -\bar{L}_w(1:m, Ml - il + 1 : Ml - il + l), \\ s_i &\stackrel{\text{def}}{=} \bar{L}_w(1:m, M(l+m) - im + 1 : M(l+m) - im + m). \end{aligned}$$

For SISO systems, (6.13) can be easily transformed into a classical frequency domain RS-configuration (See Figure 6.4):

$$U(q) = \frac{R(q)}{S(q)} Y(q)$$

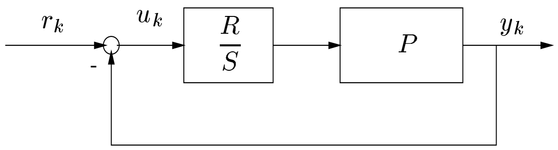


Figure 6.4: RS **controller structure.**

and

$$
\begin{aligned}
R(q) &= \sum_{i=1}^{M} r_i q^{-i}, \\
S(q) &= 1 + \sum_{i=1}^{M} s_i q^{-i}.
\end{aligned}
$$

with q the forward shift operator.

6.4 Algorithm

In this section we present the resulting model-free subspace-based LQG-algorithm. It will be shown that there are fare-going analogies with the subspace identification algorithms of Section 2.4. For instance, one can also clearly distinguish three basic steps i.e. an LQ-decomposition, an SV-decomposition and a third step which consists in the calculation of the controller parameters. The basic outline of the algorithm goes as follows:

Step 1: LQ-decomposition

In the subspace-based control law (6.11) one can clearly distinguish the term $L_w w_p$ which, as we have seen in Section 2.4.2, is equal to the product of the extended observability matrix Γ_N and a nonsteady-state Kalman filter estimate $\widehat{x}_f$. This term can be easily found from the oblique projection (2.35):

$$
\begin{aligned}
Y_f /_{U_f} W_p &= \Gamma_N \widehat{X}_f \\
&= L_w W_p.
\end{aligned}
$$

As recalled in Section 2.3.4, a very efficient way to calculate this projection is the following LQ-decomposition (2.46):

$$
\begin{pmatrix} W_p \\ U_f \\ Y_f \end{pmatrix} \overset{\mathrm{LQ}}{=} \begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} Q_1^T \\ Q_2^T \\ Q_3^T \end{pmatrix}. \tag{6.14}
$$

L_w and L_u can be easily found from:

$$L = \begin{pmatrix} L_{31} & L_{32} \end{pmatrix} \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix}^{\dagger}$$

as:

$$\begin{aligned} L_w &= L(:, 1 : i(m+l)), \\ L_u &= L(:, i(m+l)+1 : end). \end{aligned}$$

Step 2: Singular value decomposition

The extended observability matrix Γ_N and the Kalman filter state estimate $\widehat{x}_1$ can be found from the weighted singular value decomposition of L_w:

$$\begin{aligned} W_1 L_w W_2 &= \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \\ &\approx U_1 S_1 V_1^T \end{aligned}$$

from which follows:

$$\begin{aligned} \Gamma_N &= W_1^{-1} U_1 S_1^{1/2}, \\ \widehat{x}_1 &= S_1^{1/2} V_1^T W_2^{-2} w_p. \end{aligned}$$

Note that, like in subspace system identification, we weighed the above singular value decomposition with two weighting matrices W_1 and W_2. The reason for doing so will become clear in Chapter 8 where we show that making an appropriate choice for W_1 and W_2 leads to a balanced controller.

Step 3: Implementation of the control law

Finally the control law (6.12) becomes:

$$\widehat{u}_1 = -L_c \widehat{x}_1$$

where the LQ-controller feedback gain L_c can be calculated as:

$$L_c = \left[(\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q} \right] (1 : m, :) \Gamma_N.$$

Since we are dealing with a receding horizon, only the first element of $\widehat{u}_f$, i.e. $\widehat{u}_1$, will be implemented. At the next time-step one has to recalculate $\widehat{x}_1$ by inserting the new value for w_p in the above expression. Note that the SVD does not have to be recalculated.

The final model-free subspace-based LQG-algorithm can now be implemented as in Figure 6.5. Some final remarks on this algorithm:

Model-free subspace-based LQG-algorithm

LQ-decomposition: Construct the data block-Hankel matrices Y_f, U_f and $W_p = \binom{Y_p}{U_p}$ from the available open-loop input/output data and determine L_w and L_u from the following projection (with an LQ-decomposition):

$$Y_f / \begin{pmatrix} W_p \\ U_f \end{pmatrix} \stackrel{\text{LQ}}{=} L_w W_p + L_u U_f.$$

SV-decomposition: Calculate the following weighted **SVD**:

$$W_1 L_w W_2 \stackrel{\text{SVD}}{=} U_1 S_1 V_1^T$$

and derive the observability matrix Γ_N and the Kalman filter estimate $\widehat{x}_1$ of the system:

$$\begin{aligned} \Gamma_N &= W_1^{-1} U_1 S_1^{1/2}, \\ \widehat{x}_1 &= S_1^{1/2} V_1^T W_2^{-1} w_p. \end{aligned} \tag{6.15}$$

Controller: Implement the first input $\widehat{u}_1$ of the LQG-control sequence $\widehat{u}_f$:

$$\widehat{u}_1 = -L_c \widehat{x}_1$$

with $L_c = [(\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q}](1:m, :)\Gamma_N$ and:

$$w_p = \begin{pmatrix} y_{-M+1}^T & \dots & y_{-1}^T & y_0^T \mid u_{-M+1}^T & \dots & u_{-1}^T & u_0^T \end{pmatrix}^T.$$

In order to implement the following control input, calculate the new state estimate (6.15) with the new control input vector w_p.

Figure 6.5: **Model-free subspace-based LQG-algorithm. One can clearly distinguish the first two steps of subspace system identification (See Figure 2.7) i.e. the LQ-decomposition and the SV-decomposition.**

Remark 6.4.1 It is no coincidence that the term "subspace identification" is not present in the title of this chapter. Although the results are clearly based on techniques from the field of subspace identification, the system identification step is inherent in the way of solving the control design problem in the sense that the system parameters A, B, C, D and Q, R, S are never explicitly calculated. This is the reason why we referred to it as model-free LQG-design. Except for the user-defined parameters Q_i and R_i, the different steps in the design of the LQG-controller, i.e. the system identification step, the Kalman filter and the LQR-design, are replaced by an LQ- and an SV-decomposition (See also Figure 1.3).

Remark 6.4.2 The readers familiar with Model Predictive Control will certainly have noticed there is a link between the subspace-based LQG control law (6.11) and the MPC law:

$$\widehat{u}_f \quad = \quad -(\bar{R} + (H_N^d)^T \bar{Q} H_N^d)^{-1} (H_N^d)^T \bar{Q} \Gamma_N \widehat{x}_1 \tag{6.16}$$

where Γ_N is the extended observability matrix (2.14), H_N^d is the block Toeplitz matrix (2.17) containing the impulse response of the system and $\widehat{x}_1$ is the steady-state solution of the Kalman filter state equations (6.5)-(6.7). The relation between LQG and MPC has already been pointed out in [17].

As we will see in Theorem 6.5.1, under the conditions that the number of block rows in the past data block-Hankel matrix W_p, and therefore the length of the backward horizon is infinite ($M = \infty$), the MPC control law (6.16) and our subspace-based LQG-control law (6.11) are the same. Indeed, under these conditions, the following substitutions are allowed:

$$\begin{pmatrix} \widehat{x}_f \\ \Gamma_N \\ H_N^d \end{pmatrix} \quad \longleftrightarrow \quad \begin{pmatrix} S_1^{1/2} V_1^T W_2^{-1} w_p \\ W_1^{-1} U_1 S_1^{1/2} \\ L_u \end{pmatrix}.$$

The main difference between the MPC-approach and our method is the way the control law is calculated. Whereas the subspace-based LQG-algorithm of Figure 6.5 shows that the Kalman filter estimate $\widehat{x}_f$ can be found directly from an LQ- and an SV-decomposition, for MPC:

- an explicit system identification is necessary, which means that the system matrices A, B, C, D, Q, R, S have to be determined explicitly.
- the Kalman filter equations (6.5)-(6.7) have to be solved.

This is the main advantage of our approach, when compared to MPC and LQG; a Kalman filter state estimate of the unknown system and the LQ-feedback gain can be directly found from an LQ- and a SV-decomposition. The explicit calculation of the state-space matrices A, B, C, D, Q, R, S is not required.

Remark 6.4.3 The extension to output tracking is straightforward. Assume that the desired output trajectory r_k is given, the performance criterion (6.4) can be written as:

$$J = (\widehat{y}_f - r_f)^T \bar{Q}(\widehat{y}_f - r_f) + \widehat{u}_f^T \bar{R} \widehat{u}_f$$

where $r_f \stackrel{\mathrm{def}}{=} (r_1^T \ \dots \ r_N^T)^T$. The control law (6.11) then simply becomes:

$$\widehat{u}_f = (\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q}(r_f - L_w w_p).$$

In the classical LQG-framework, the extension to the output tracking case is much more involved [17].

6.5 Equivalence between the two approaches

If one compares the subspace-based LQG control algorithm of Figure 6.5 with the classical LQG control law (6.5)-(6.10), one might wonder what is the link between the two. Are they equivalent and under what circumstances? Fundamental differences between the two are that the classical LQG-problem requires that the user does a system identification and chooses an initial condition for the Kalman filter. For the subspace-based LQG-controller, no choice of initial conditions has to be made and the system identification step is inherent to the way of solving the problem. The only thing one has to do is an LQ- and an SV-decomposition of block-Hankel matrices containing input/output data of the unknown system. Of course, the following fundamental question then arises:

Equivalence problem:

What are

- *the state-space matrices A, B, C, D, Q, R, S and*
- *the initial condition $\widehat{x}_{-M+1}$ of the Kalman filter equations (6.5)-(6.7),*

such that the classical LQG-controller (6.5)-(6.10) gives the same control $\widehat{u}_1$ as the subspace-based LQG-control algorithm of Figure 6.5 ?

Although we do not have the general answer to the above question yet, under some particular conditions we were able to derive a solution. Two cases are considered, namely the infinite backward horizon case ($M = \infty$), and the case where the inputs u_k of the available data set are white zero-mean.

6.5.1 Infinite backward horizon

In the case the backward horizon M is infinite, it is possible to prove that the classical and the subspace-based LQG-algorithm are equivalent as defined above.

Theorem 6.5.1 *[40], [42]*
If the available data set is large ($j = \infty$) and the number of block rows M in the past data block-Hankel matrix W_p is infinite ($M = \infty$), then the subspace-based LQG-control algorithm (Figure 6.5) and the classical LQG-controller equations (6.5)-(6.10) produce the same value for $\widehat{u}_1$. The choice of the initial Kalman filter state estimate $\widehat{x}_{-M+1}$ does not play a role and can thus be made arbitrarily.

Proof: As mentioned in Section 2.4, the $L_w w_p$-term in the LQG-controller equation (6.11) equals $\Gamma_N \widehat{x}_1$ where Γ_N is the extended observability matrix and $\widehat{x}_1$ a nonsteady-state Kalman filter estimate corresponding to the Kalman filter equations (6.5)-(6.7). In the particular case where the backward horizon M is infinite, or equivalently if the number of block rows M in the past data block-Hankel matrix W_p is infinite, the Kalman filter estimate $\widehat{x}_1$ becomes steady-state. Steady-state in the sense that the influence of the error on the initial state estimate $\widehat{x}_{-M+1}$ has disappeared. This can be seen from Figure 2.5 for an infinite i-index, i.e. the number of block rows in the data block-Hankel matrix W_p. As a consequence, the choice of the initial Kalman filter estimate $\widehat{x}_{-M+1}$ does not play a role and can be made arbitrarily.

As mentioned in Remark 2.5.1 and 6.4.2, we have that $L_u = H_N^d$ in the case the backward horizon is infinite. The second part of the proof then consists in showing that $\widehat{u}_1$, the first element of the future control sequence $\widehat{u}_f$

$$\widehat{u}_f \quad = \quad -(\bar{R} + (H_N^d)^T \bar{Q} H_N^d)^{-1} (H_N^d)^T \bar{Q} \Gamma_N \widehat{x}_1,$$

equals the solution of the LQR-equations (6.8)-(6.10). We refer to Appendix A.5 for the mathematical details. ■

6.5.2 White inputs

Let us now consider the case where the inputs u_k of the available data set are white and zero-mean. This is a quite common situation since white signals are often used as inputs for system identification purposes. In this particular case, the equivalence between the classical and the subspace-based LQG-controller can also be made.

Theorem 6.5.2 *[40], [42]*
If the available data set is large ($j = \infty$) and the inputs u_k of the available data

are white and zero-mean, then the subspace-based LQG-algorithm of Figure 6.5 and the classical LQG-controller equations (6.5)-(6.10) with initial Kalman filter state estimate $\widehat{x}_{-M+1} = X_p Y_p^\dagger y_p$ give the same control input $\widehat{u}_1$.

Proof: The proof is very similar to that of Theorem 6.5.1. The $L_w w_p$-term in the LQG-controller equation (6.11) also equals $\Gamma_N \widehat{x}_1$ where Γ_N is the extended observability matrix and $\widehat{x}_1$ the Kalman filter estimate corresponding to the Kalman filter equations (6.5)-(6.7). The initial conditions of the Kalman filter are determined by the projection (See Theorem 2.4.3):

$$\widehat{X}_p = X_p /_{U_f} W_p.$$

If the inputs u_k are white zero-mean, this projection equals:

$$\begin{aligned} \widehat{X}_p &= X_p / Y_p \\ &= X_p Y_p^\dagger Y_p, \end{aligned}$$

which implies that the initial Kalman filter estimate $\widehat{x}_{-N+1} = X_p Y_p^\dagger y_p$. As mentioned in Remark 2.5.1, if the input u_k is white we have that $L_u = H_N^d$.

The proof of (6.8)-(6.10) is the same as the proof of Theorem 6.5.1 (see Appendix A.5). ∎

6.6 Example

Let us now use an example to compare the subspace-based LQG-control algorithm of Figure 6.5 and the classical LQG-controller (6.5)-(6.10). The example is partially borrowed from [65] and was also used in [125] and [122]. The plant corresponds to a discrete-time model of a laboratory setup of two circular plates rotated by an electrical servo motor with flexible shafts. The state-space matrices of the model (6.1)-(6.3) in innovation form are:

$$A = \begin{pmatrix} 4.40 & 1 & 0 & 0 & 0 \\ -8.09 & 0 & 1 & 0 & 0 \\ 7.83 & 0 & 0 & 1 & 0 \\ -4.00 & 0 & 0 & 0 & 1 \\ 0.86 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0.00098 \\ 0.01299 \\ 0.01859 \\ 0.0033 \\ -0.00002 \end{pmatrix},$$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad K = \begin{pmatrix} 2.3 \\ -6.64 \\ 7.515 \\ -4.0146 \\ 0.86336 \end{pmatrix},$$

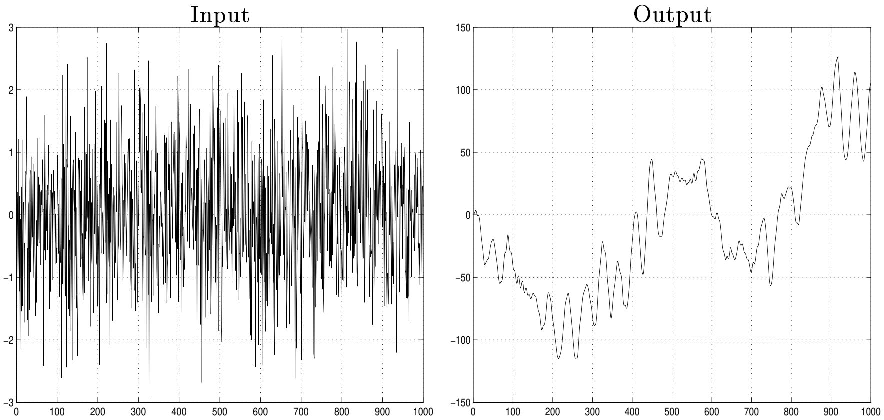


Figure 6.6: **Left: the input signal u_k is a white zero-mean stationary signal. Right: the corresponding output signal y_k. One can clearly see a trend on the output which is due to the integrator in the system.**

$D = 0$ and e_k is a Gaussian zero-mean white noise sequence with $\mathbf{E}[e_k^2] = 1/9$. Note that the plant has an integrator and is therefore marginally stable.

The objective of this example is to compare the subspace-based LQG-controller of Figure 6.5 and the classical LQG-controller (6.5)-(6.10). The comparison is made by applying both controllers to the same input/output data set of the unknown system (6.1)-(6.3). As shown in Theorem 6.5.1, as the length of the backward horizon M increases, the subspace-based controller should converge to the classical LQG-controller. This is shown by comparing the output of the closed-loop system, controlled by the classical LQG-controller on the one hand, and by the subspace-based controller on the other hand. The available input/output data set has a length of 1000 and is given in Figure 6.6. Based on this data set, the following two controllers were calculated:

Classical: The classical LQG-controller as described by (6.5)-(6.10). In order to allow a correct comparison with the subspace-based LQG-controller, the classical LQG-controller should function as a reference. Therefore, the system identification step was omitted and the state-space matrices A, B, C, D, Q, R, S were assumed to be known (See also Section 6.2).

Subspace-based: The subspace-based LQG-controller of Figure 6.5. The influence of the length of the backward horizon M on the controller is studied.

The user-defined tuning parameters R_i and Q_i, i.e. the input and output weighting matrices of the control criterion 6.4, were chosen the same for both controllers:

$$\begin{aligned} Q_i &= 1, \forall\, i, \\ R_i &= 10, \forall\, i. \end{aligned}$$

The length of the forward horizon N in the LQG-criterion is fixed for the different simulations and equal to $N = 10$. The comparison is made for different values of the backward horizon M. Figure 6.7 compares the response of the closed-loop system, controlled with the classical LQG-controller on the one hand, and the subspace-based LQG-controller on the other. The reference output r_k was set equal to zero for the different simulations. As proven in Theorem 6.5.1, one can clearly see that, as the length of the backward horizon M increases, the response of the subspace-based controller converges to that of the classical LQG-controller.

In this chapter we have presented a new class of methods, based on subspace identification techniques, for an efficient implementation of LQG-controllers of unknown linear systems. The resulting algorithm is very similar to the existing linear subspace identification algorithms, i.e. one can also clearly distinguish an LQ-decomposition and an SV-decomposition step. Further is was shown that, if the number of block rows in the past data matrices is infinite or if the inputs are white, the subspace-based LQG-controller is equivalent with the classical LQG-controller.

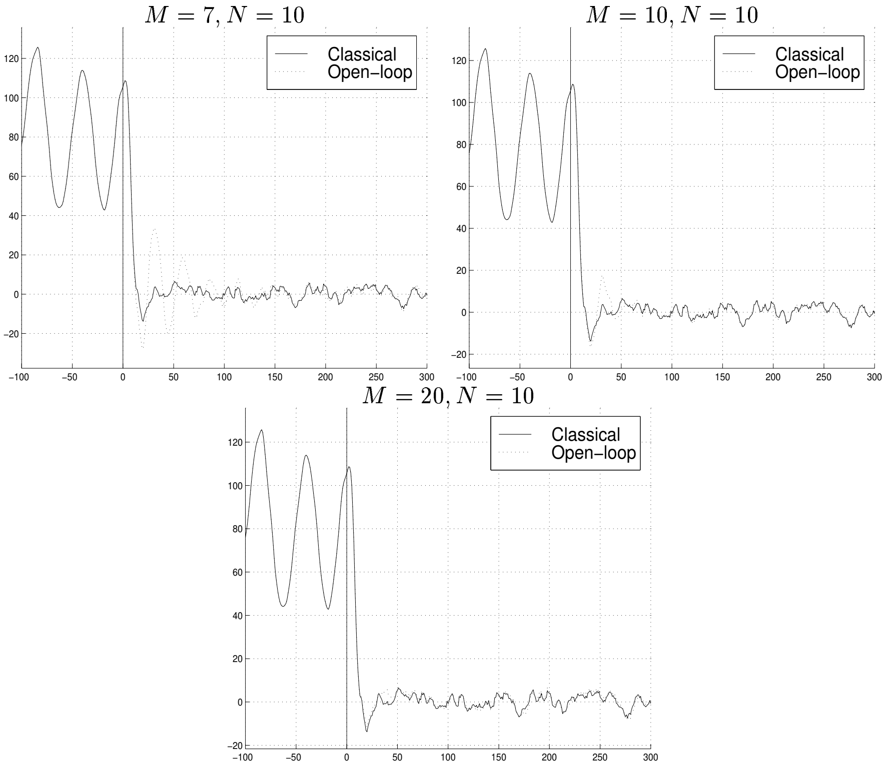


Figure 6.7: **The above figures compare the response of the system, controlled with the classical LQG-controller (full line) on the one hand, and with the subspace-based controller (dotted line) on the other hand. The comparison is made for different values of the backward horizon ($M = 7, 10, 20$) and a constant forward horizon ($N = 10$). The system is operating in open-loop up to time step 0 (vertical line). The measurements of the inputs u_k and the outputs y_k up to time $k = 0$ are then used to calculate the LQG-controller, which is then implemented from time step 1 on. Since the output reference signal r_k is chosen equal to zero, the output converges to zero due to the control action. As we could expect from Theorem 6.5.1 one can see that, as the size of the backward horizon M increases, the response of the system controlled by the subspace-based controller converges to that of the classical LQG-controller .**

Chapter 7

Closed-loop model-free subspace-based LQG-design

The main result of the previous chapter was that, based on techniques used in subspace system identification, an LQG-controller of an unknown system can be found in a very direct way from a set of measurements of the inputs and the outputs of that system. It was assumed that the available input and output data, with which the controller is calculated, were measured under open-loop conditions. In the present chapter we explain into more details why this condition has to be fulfilled and provide a solution for the case the data were measured in closed-loop.

7.1 Introduction

The problem treated in this chapter is very similar to the one of the previous chapter, namely we are also interested in the design of an LQG-controller for unknown linear time-invariant systems of the form:

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{7.1}$$
$$y_k = Cx_k + Du_k + v_k \tag{7.2}$$

with the stationary, ergodic, white Gaussian noise sequences $w_k \in \mathbb{R}^n$ and $v_k \in \mathbb{R}^l$ having the following covariance matrix:

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix}\begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}\delta_{pq} \geq 0. \tag{7.3}$$

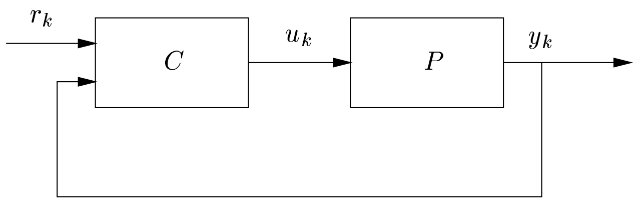


Figure 7.1: **The purpose of the present chapter is to design an LQG-controller for the unknown plant P which is operating in closed-loop. The preliminary controller C is assumed to be linear time-invariant, but not necessarily an LQG-controller. For the LQG-design we only dispose of a set of measurements of the control inputs u_k, the outputs y_k and the first N impulse response coefficients of the preliminary controller C.**

The main difference with Chapter 6 is that the data, which will be used to calculate the LQG-controller, are now assumed to be measured under closed-loop conditions. The provisional controller, that is not necessarily an LQG-controller, is also assumed to be linear time-invariant and described by the following state-space equations:

$$x_{k+1}^c = A_c x_k^c + B_c^y y_k + B_c^r r_k, \tag{7.4}$$
$$u_k = C_c x_k^c + D_c^y y_k + D_c^r r_k. \tag{7.5}$$

$r_k \in \mathbb{R}^l$ is the reference output, $x_k^c \in \mathbb{R}^{n_c}$ the state of the controller and n_c is the order of the provisional controller. The interaction between this controller and the plant is depicted in Figure 7.1. The initial assumptions are the same as in the previous chapter, i.e. the plant (7.1)-(7.3) is unknown and a large amount of measurements of the inputs u_k and outputs y_k is available. An additional assumption we make here is that the first N, where N is the length of the forward horizon, impulse response coefficients of the provisional controller (7.4)-(7.5) are known. This is a realistic assumption since in most practical situations the controller is known. If this is not the case, a preliminary impulse response test or system identification on the controller is easily done.

We further assume that the problem is well-posed in the sense that the output y_k should be uniquely determined by the reference input r_k, the disturbance inputs w_k, v_k and both the state of the controller x_k^c and the system x_k. This condition is satisfied when $I_l + DD_c^y$ is non-singular.

The main problem we are interested in, in this chapter, is referred to as the closed-loop model-free LQG-control problem and is defined in Figure 7.2.

Closed-loop model-free LQG-Control Problem

Given *a set of closed-loop measurements of the inputs* u_k *and the outputs* y_k, $k \leq 0$, *of the unknown system (7.1)-(7.2) and the first* N *impulse response coefficients of the controller (7.4)-(7.5),* **find** *the input sequence* $\widehat{u}_f = (\widehat{u}_1, \ldots, \widehat{u}_N)$ *that minimizes the quadratic cost function* J *over the horizon* N*:*

$$J = \sum_{k=1}^{N} (\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k. \tag{7.6}$$

Figure 7.2: **Closed-loop model-free LQG-control problem.**

7.2 Closed-loop subspace system identification

7.2.1 Problem definition

In this section we briefly recall a result from the literature [122], [124] that generalizes the open-loop linear subspace identification problem, as recalled in Chapter 2, to an analogous theory for closed-loop subspace identification. The problem definition is given in Figure 7.3.

7.2.2 Brief overview of the theory

As we have seen in Chapter 2, an important set of equations in subspace system identification theory are the matrix input/output equations (2.27)-(2.29). They represent the effect of the state x_k, the deterministic input u_k and the unknown

Closed-loop linear subspace identification problem

Given measurements of the inputs u_k and the outputs y_k of the unknown system (7.1)-(7.3) and the first i impulse response coefficients of the controller (7.4)-(7.5), **find** an estimate of the system matrices A, B, C, D and the noise related matrices Q, R, S.

Figure 7.3: **Closed-loop linear subspace identification problem.**

stochastic input e_k on the outputs y_k, and can be directly derived from the state-space equations (7.1)-(7.2). Since the controller (7.4)-(7.5) is also a linear time-invariant system, an analogous set of equations can be derived:

$$\begin{aligned} U_f &= \Gamma_c X_f^c + H_c^y Y_f + H_c^r R_f, & (7.7)\\ U_p &= \Gamma_c X_p^c + H_c^y Y_p + H_c^r R_p, & (7.8)\\ X_f^c &= A_c^i X_p^c + \Delta_c^y Y_f + \Delta_c^r R_f & (7.9)\end{aligned}$$

where the state sequence X_f^c of the controller is defined as:

$$X_f^c \stackrel{\text{def}}{=} \begin{pmatrix} x_i^c & x_{i+1}^c & \dots & x_{i+j-1}^c \end{pmatrix}. \tag{7.10}$$

Γ_c is the extended observability matrix of the controller, H_c^y and H_c^r the block-Toeplitz matrices containing the impulse response coefficients of the controller inputs y_k and r_k respectively. Δ_c^y and Δ_c^r are the extended reversed controllability matrices relative to the controller inputs y_k and r_k respectively. These matrices are defined in a similar way as the corresponding matrices (2.14)-(2.18) of the plant:

$$\Gamma_c \stackrel{\text{def}}{=} \begin{pmatrix} C_c \\ C_c A_c \\ \dots \\ C_c A_c^{i-1} \end{pmatrix}, \tag{7.11}$$

$$H_c^y \stackrel{\text{def}}{=} \begin{pmatrix} D_c^y & 0 & \dots & 0 \\ C_c B_c^y & D_c^y & \dots & 0 \\ \dots & \dots & \dots & \dots \\ C_c A_c^{i-2} B_c^y & C_c A_c^{i-3} B_c^y & \dots & D_c^y \end{pmatrix}, \tag{7.12}$$

$$H_c^r \stackrel{\text{def}}{=} \begin{pmatrix} D_c^r & 0 & \dots & 0 \\ C_c B_c^r & D_c^r & \dots & 0 \\ \dots & \dots & \dots & \dots \\ C_c A_c^{i-2} B_c^r & C_c A_c^{i-3} B_c^r & \dots & D_c^r \end{pmatrix}, \tag{7.13}$$

$$\Delta_c^y \stackrel{\text{def}}{=} \begin{pmatrix} A_c^{i-1} B_c^y & \dots & A_c B_c^y & B_c^y \end{pmatrix}, \tag{7.14}$$

$$\Delta_c^r \stackrel{\text{def}}{=} \begin{pmatrix} A_c^{i-1} B_c^r & \dots & A_c B_c^r & B_c^r \end{pmatrix}. \tag{7.15}$$

Note that, in order to avoid overload of notation, the index i has been omitted in the left hand side of the above definitions.

If the data were measured in open-loop, the subspace identification algorithms of Chapter 2 would work fine. However, if the data is recovered under closed-loop conditions, the estimated model will be biased. The reason for this is that, due to the feedback, the inputs U_f are correlated with the residuals E_f. In that case, the solution L_w, L_u of the least squares problem (2.31):

$$\min_{L_w, L_u} ||Y_f - \begin{pmatrix} L_w & L_u \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix} ||_F^2$$

will be asymptotically ($j \to \infty$) biased. This was shown and illustrated with simulation examples in [122], [124].

However, there is a way to get around this problem [124]. The "trick" is to replace the future input Hankel matrix U_f by a new matrix M_f which is not correlated with the residuals E_f. The philosophy behind this approach is very similar to that of the well known instrumental variable method (See e.g. [64]). The problem that remains to be solved now is how to choose M_f. Substituting U_f from the system matrix input/output equation (2.28) with U_f from the controller matrix input/output equation (7.7) we have:

$$Y_f \quad = \quad T_c \left\{ \Gamma X_f + H^d M_f + H^s E_f \right\} \tag{7.16}$$

where we define $T_c \stackrel{\text{def}}{=} (I - H^d H^y_c)^{-1}$ and $M_f \stackrel{\text{def}}{=} U_f - H^y_c Y_f$. Note that T_c is invertible due to the initial assumption that the problem is well-posed. The matrix M_f can be directly calculated from its definition since we assumed that the first i impulse response coefficients of the controller (7.4)-(7.5), and therefore H^y_c, are known. From (7.7) it can be seen that M_f can also be written as $M_f = R_f - \Gamma_c X^c_f$. Since $R_f E^T_f = 0$ and $X^c_f E^T_f = 0$, it is easy to see that:

$$M_f E_f^T \quad = \quad 0.$$

In this sense M_f can be considered as a matrix containing the instrumental variables since the elements of M_f are uncorrelated with the innovations e_k. The closed-loop subspace identification problem now consists in finding the optimal linear prediction $\widehat{Y}_f$ of the future outputs Y_f, given the past inputs and outputs W_p and the instrumental variables M_f. Using a least squares criterion:

$$\min_{L^c_w, L^c_u} ||Y_f - \begin{pmatrix} L^c_w & L^c_u \end{pmatrix} \begin{pmatrix} W_p \\ M_f \end{pmatrix} ||^2_F \tag{7.17}$$

the predictor parameters L^c_w and L^c_u can found by projecting the future outputs Y_f into the past input and output space W_p and the instrumental variable space M_f. This can be easily calculated with an LQ-decomposition (See Section 2.5):

$$\begin{aligned} \widehat{Y}_f &\stackrel{\text{LQ}}{=} Y_f / \begin{pmatrix} W_p \\ M_f \end{pmatrix} \\ &= L^c_w W_p + L^c_u M_f. \end{aligned} \tag{7.18}$$

When applied to closed-loop data the projection (7.18), unlike the projection (2.31) used in open-loop subspace identification, leads to asymptotically ($j \to \infty$) unbiased estimates of L^c_w and L^c_u.

It comes as no surprise that, in the present closed-loop subspace identification framework, a similar theorem can be found as the open-loop Theorem 2.4.3 [122]. The proof of the following theorem can be found in [122]:

Theorem 7.2.1 (Main closed-loop subspace identification theorem)
Under the assumptions that:

- *the input u_k is persistently exciting of order $2i$ (Definition 2.4.2),*
- *the number of available data points is large, so that $j \to \infty$,*
- *the noise e_k is not identically zero,*
- *the first i impulse response coefficients of the controller (7.4)-(7.5) are known,*

then we have that the weighted oblique projection defined as:

$$Y_f /_{M_f} W_p = \mathcal{G}_i \widehat{X}_f, \tag{7.19}$$

equals the product between $\mathcal{G}_i \stackrel{\text{def}}{=} T_c \Gamma_i$, with $\widehat{X}_f$ the nonsteady-state Kalman filter estimate of X_f with $X_p /_{M_f} W_p$ as the initial state (See Figure 2.5 and Theorems 2.4.1 and 2.4.3).

7.2.3 Algorithm

The closed-loop version of the open-loop subspace identification algorithm of Figure 2.7 is given in Figure 7.4. The only difference with the open-loop algorithm is that the data block-Hankel matrix containing the future output U_f is replaced by M_f in step 1.

7.3 Closed-loop subspace-based LQG-controller

We have seen that it is possible to calculate an asymptotically unbiased estimate of L_w^c and L_u^c from the projection (7.18). However, as we have seen in Chapter 6, the matrices we are really interested in for the subspace-based LQG-controller design are L_u and L_w, for which only an asymptotically biased estimate can be found from the least squares problem (2.31). It is possible however to calculate consistent estimates of L_w and L_u, based on the knowledge of L_w^c, L_u^c and H_c^y. From Theorem 6.5.1 we know that, if the number of block rows M in the past data block-Hankel matrices W_p is infinite, we have that $L_u = H_N^d$. In that case, it can be seen from the closed- and open-loop matrix input/output equations (2.28),(2.30) and (7.16),(7.18) that L_w^c, L_u^c and L_w, L_u are related by the following set of equations:

$$L_w^c = T_c L_w, \tag{7.20}$$

Closed-loop subspace algorithm using the states [122]

1. Calculate the oblique projections with an **LQ-decomposition**:

$$\begin{aligned}\mathcal{O}_i &\overset{\text{LQ}}{=} Y_{i|2i-1}/_{M_{i|2i-1}} W_{0|i-1} = L_w^c W_{0|i-1},\\ \mathcal{O}_{i+1} &\overset{\text{LQ}}{=} Y_{i+1|2i-1}/_{M_{i+1|2i-1}} W_{0|i}\end{aligned}$$

   and determine L_w^c.

2. Calculate the weighted **SVD**:

$$W_1 L_w^c W_2 \overset{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

   and

   - determine an estimate of the order n of the system by inspecting the number of dominant singular values S_1,
   - determine $\mathcal{G}_i$ and $\mathcal{G}_{i-1}$ as:

$$\begin{aligned}\mathcal{G}_i &= W_1^{-1} U_1 S_1^{1/2},\\ \mathcal{G}_{i-1} &= \mathcal{G}_i(1 : end - l, :).\end{aligned}$$

   - determine the state sequences:

$$\widehat{X}_i = S_1^{1/2} V_1^T W_2^{-1} W_p, \qquad \widehat{X}_{i+1} = \mathcal{G}_{i-1}^{\dagger} \mathcal{O}_{i+1}.$$

3. Calculate an estimate of the system model A, B, C, D and the noise covariance matrices Q, R, S from the following linear least squares problem:

   System model:

$$\begin{pmatrix} \widehat{X}_{i+1} \\ Y_{i|i} \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} \widehat{X}_i \\ U_{i|i} \end{pmatrix} + \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}.$$

   Noise model:

$$\begin{pmatrix} Q & S \\ S^T & R \end{pmatrix}_i = \frac{1}{j}[\begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix} \begin{pmatrix} \rho_w \\ \rho_v \end{pmatrix}^T].$$

Figure 7.4: **A schematic overview of a simple closed-loop subspace system identification algorithm. The only difference with the open-loop algorithm of Figure 2.7 is that the matrix with the future inputs U_f is replaced by a matrix M_f, calculated from the available inputs, outputs and controller impulse response.**

$$L_u^c = T_c L_u.$$

From the definition of T_c it is now easy to derive that the open-loop parameters L_u and L_w can be calculated as:

$$L_u = L_u^c(I + H_c^y L_u^c)^{-1}, \tag{7.21}$$
$$L_w = (I - L_u H_c^y)L_w^c. \tag{7.22}$$

The final closed-loop model-free subspace-based LQG-algorithm can now be implemented as in Figure 7.5.

7.4 Example

Now, the same example as the one in Section 6.6 is used to compare the presently discussed closed-loop subspace-based LQG-control algorithm with the open-loop subspace-based LQG-algorithm of Chapter 6. The configuration of plant and controller is the one depicted in Figure 7.1. The controller state-space matrices are:

$$A_c = \begin{pmatrix} 2.65 & -3.11 & 1.75 & -0.39 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad B_c^y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad C_c^T = \begin{pmatrix} 0.4135 \\ -0.8629 \\ 0.7625 \\ -0.2521 \end{pmatrix},$$

$D_c^y = -0.61$, $D_c^r = 1$ and $B_c^r = 0_{4\times 1}$. The input/output data set that will be used in the algorithm is generated by applying a zero-mean white signal to the reference r_k of length 1000 (Figure 7.6). On this data set the following three algorithms were applied:

Classical: The classical LQG-algorithm to solve the LQG-control problem as described by Equations (6.5)-(6.10). Just like in the example of Section 6.6, for the classical LQG-controller we did not identify the state-space matrices A, B, C, D, Q, R, S from the data but assumed that they were known. The reason for doing so is that, in this way, the classical LQG-controller can be used as a reference controller.

Open-loop: The open-loop subspace-based LQG-algorithm presented in Figure 6.5. Since closed-loop input/output data was used, this algorithm will give biased results.

Closed-loop: The closed-loop subspace-based LQG-algorithm of Figure 7.5.

We take the forward horizon in the LQG criterion equal to $N = 10$. For the three algorithms, the same value of the user-defined matrices $R_i = 10$ and

Closed-loop model-free subspace-based LQG-controller

LQ-decomposition: Construct the data block-Hankel matrices $Y_f, U_f, W_p = \binom{Y_p}{U_p}$ and the matrix $M_f = U_f + H_c^y Y_f$ from the available input/output data and the N first elements of the preliminary controller impulse response. Determine L_w^c and L_u^c from the following projection (with an LQ-decomposition):

$$Y_f / \begin{pmatrix} W_p \\ M_f \end{pmatrix} \overset{\text{LQ}}{=} L_w^c W_p + L_u^c M_f.$$

Derive the open-loop predictor parameters:

$$\begin{aligned} L_u &= L_u^c (I + H_c^y L_u^c)^{-1}, \\ L_w &= (I - L_u H_c^y) L_w^c. \end{aligned}$$

SV-decomposition: Calculate the following weighted SVD:

$$W_1 L_w W_2 \overset{\text{SVD}}{=} U_1 S_1 V_1^T$$

and derive the observability matrix Γ_N and the estimated Kalman filter state estimate $\widehat{x}_1$ of the system:

$$\begin{aligned} \Gamma_N &= W_1^{-1} U_1 S_1^{1/2}, \\ \widehat{x}_1 &= S_1^{1/2} V_1^T W_2^{-1} w_p. \end{aligned} \tag{7.23}$$

Controller: Implement the first element $\widehat{u}_1$ of the LQG-control sequence $\widehat{u}_f$:

$$\widehat{u}_1 = -L_c \widehat{x}_1$$

with $L_c = [(\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q}](1:m,:)\Gamma_N$ and:

$$w_p = \begin{pmatrix} y_{-M+1}^T & \ldots & y_{-1}^T & y_0^T \mid u_{-M+1}^T & \ldots & u_{-1}^T & u_0^T \end{pmatrix}^T.$$

In order to implement the following control input, calculate the new state estimate (7.23) with the new control input vector w_p.

Figure 7.5: **Closed-loop model-free subspace-based LQG-control algorithm. Given a set of closed-loop measurements of the inputs u_k and the outputs y_k of an unknown process, and the knowledge of the preliminary controller, this algorithm calculates an LQG-controller. The main difference with its open-loop version (Figure 6.5) is that U_f in the projection is replaced by M_f.**

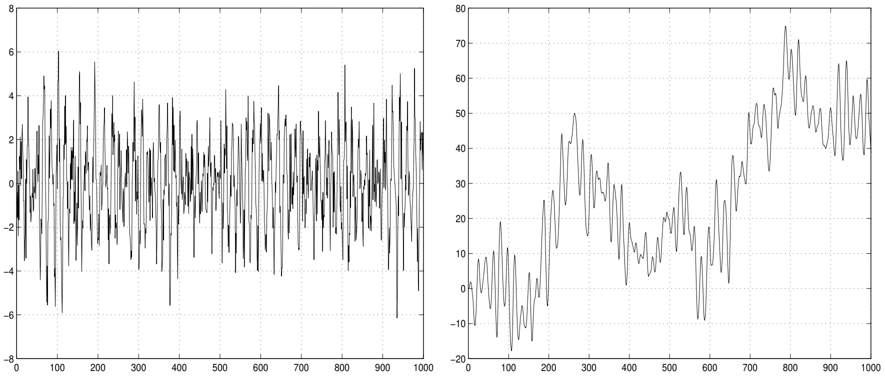


Figure 7.6: **Left: the reference input signal r_k is a white Gaussian zero-mean signal. Right: the corresponding output signal y_k.**

$Q_i = 1$ was used. Different values for the backward horizon M were used. Figure 7.7 compares the response of the closed-loop system controlled with the classical LQG-controller (6.5)-(6.10), the open-loop subspace-based LQG-controller (Figure 6.5) and the closed-loop subspace-based LQG-controller (Figure 7.5).

Up to time-step 0, the system is operating under feedback of the preliminary controller (7.4)-(7.5). The data up to that time is, together with the N first impulse response coefficients of the controller (7.4)-(7.5), used to calculate the new LQG-controller. From time-step 1 on, this LQG-controller is then implemented. The reference r_k of this controller is set to 0. One can see that the closed-loop subspace-based controller converges to the classical LQG-controller as the size of the backward horizon M increases (See Theorem 6.5.1). It is clear that, due to the fact that the data was measured in closed-loop, the open-loop subspace-based LQG-controller is biased and may even become unstable.

In this chapter we have presented an extension of the subspace-based LQG-algorithm of Chapter 6 for the case where the available input/output data were measured under closed-loop conditions. This algorithm can therefore be used when one is for instance interested in calculating an LQG-controller for an unknown system which is previously controlled by a known PID controller. The algorithm is a direct extension and generalization of the open-loop algorithm of Chapter 6.

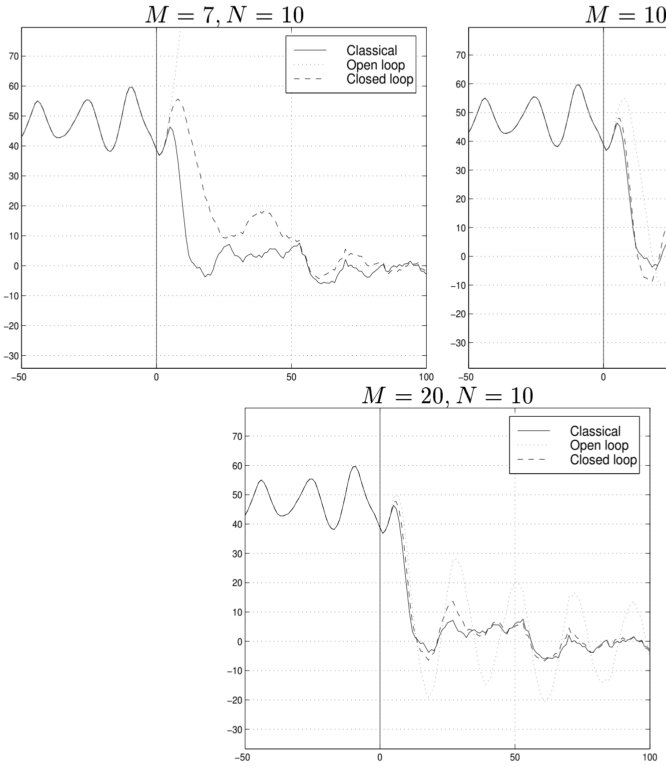


Figure 7.7: **The above figures compare the response of the closed-loop system controlled with the classical LQG-controller (full line), the open-loop subspace-based LQG-controller (dotted line) and the closed-loop subspace-based LQG-controller (dashed line). The comparison is made for different values of the backward horizon ($M = 7, 10, 20$) and a constant forward horizon ($N = 10$). The controller is calculated based on the data up to time step 0 (vertical line). From time step 1 on the new LQG-controller is implemented. The reference signal r_k is equal to zero. One can see that the closed-loop subspace-based controller converges to the classical LQG-controller as the size of the backward horizon M increases (See Theorem 6.5.1). It is clear that, due to the fact that the data was measured in closed-loop, the open-loop subspace-based LQG-controller is biased and may even become unstable.**

Chapter 8

Subspace-based balanced controller reduction

In Chapter 6 a new algorithm was derived, based on techniques from subspace system identification, for an efficient implementation of LQG-controllers, directly from input and output measurements of an unknown system. As we will discover in the present chapter, the order of these controllers is equal to the length of the backward horizon M, which can be quite large. The objectives of this chapter are twofold. First an extension of the subspace-based LQG-algorithm of Figure 6.5 will be presented that directly calculates a reduced-order LQG-controller. Secondly, we will analyze how a specific choice of the weighting matrices W_1 and W_2 can influence this reduction. We will see that, by choosing these weighting matrices in an appropriate way, it is possible to find a balanced LQG-controller, directly from the input/output data.

8.1 Introduction

The problem with the subspace-based LQG-controller (6.13) of Chapter 6 is that the order of the controller is equal to the number of block rows M in the past data block-Hankel matrices:

$$u_k = -\sum_{i=1}^{M} s_i u_{k-i} + \sum_{i=1}^{M} r_i y_{k-i}. \tag{8.1}$$

We have seen in Theorems 6.5.1 and 6.5.2 that, if the available input sequence u_k is not white, the backward horizon M has to be infinite in order to guarantee the optimality of the controller. In practise this implies that the order of the controller can be quite large indeed. When designing a linear controller, one of the important features to look after is the order of the controller. Indeed, it is the main characteristic of the complexity of the controller. Among the straightforward reasons to keep this complexity low are the implementational cost and the computational time.

In the present chapter we have two main goals:

- First we will analyze how the subspace-based LQG-algorithm of Figure 6.5 can be modified such that, instead of calculating an LQG-controller of order M, we directly find a controller of a given lower order n_r:

$$x_{k+1}^c = A_c x_k^c + \begin{pmatrix} B_c^y & B_c^r \end{pmatrix} \begin{pmatrix} y_k \\ r_k \end{pmatrix}, \tag{8.2}$$

$$u_k = C_c x_k^c + \begin{pmatrix} D_c^y & D_c^r \end{pmatrix} \begin{pmatrix} y_k \\ r_k \end{pmatrix} \tag{8.3}$$

  where the state $x_k^c \in \mathbb{R}^{n_r}$.

- Further we will study the influence of the weighting matrices W_1 and W_2 on the resulting controller. The basic idea comes from subspace system identification where it was shown that [123], [120], by choosing the weighting matrices W_1 and W_2 in the subspace identification algorithm of Figure 2.7 in a specific way, the obtained state-space model is in open-loop balanced from. This idea is extended in Section 8.4 where we show that, by introducing weighting matrices in an a similar way as for subspace system identification, it is possible to obtain a balanced LQG-controller directly from the input/output data.

The problem treated in the present chapter can be stated in a general way as in Figure 8.1.

The chapter is divided into three main parts. In Section 8.2 the main results of classical open- and closed-loop balanced truncation are recalled. Section 8.3 focuses on the available results on balanced subspace system identification, results which are then used in Section 8.4 for a new subspace-based balanced controller reduction algorithm.

8.2 Classical balanced controller reduction

In the literature there exist different methods to calculate a low-order controller. In a lot of cases, they can be split up into two main categories (See Figure 8.2) [6]:

Model-free low-order LQG-control problem

Given *a set of measurements of the inputs* u_k *and the outputs* y_k *of the unknown process (6.1)-(6.3),* **find** *a controller (8.2)-(8.3) of a given order* $n_r < M$ *that minimizes the quadratic cost function* J *over the forward horizon* N*:*

$$J = \sum_{i=1}^{N}(\widehat{y}_{k+i} - r_{k+i})^T Q_i (\widehat{y}_{k+i} - r_{k+i}) + \widehat{u}_{k+i}^T R_i \widehat{u}_{k+i}.$$

Figure 8.1: **The model-free low-order LQG-control problem.**

Plant reduction: The oldest method consists in finding a reduced-order plant model, which is then used to design an LQG-controller. This technique is referred to as *open-loop* controller reduction since the reduction is done without taking into account the effect of the loop. This way of proceeding is now generally viewed as dangerous. The danger lies in the fact that model reduction on a plant may discard dynamics that seem insignificant in open-loop, but may be excited by the controller to such an extent that the closed-loop becomes unstable. This phenomenon is called *spill-over*. Open-loop *balanced* truncation (OLBT), introduced by Moore in the early eighties [98] (See Section 8.2.1), is one of the more popular open-loop model reduction techniques, mainly due to its simplicity and the fact that an upper bound on the reduction error is provided [61].

Controller reduction: An alternative to the previous method is to first design an LQG-controller based on the high-order plant model. The resulting controller, which will be of a high order too, is then reduced in a second step. One could of course use an open-loop model reduction technique (Section 8.2.1) to reduce the controller but this way of proceeding would also lead to the spill-over phenomenon. A major difference with the plant reduction approach is that, at the moment of the reduction of the controller, the plant model is available. This is not the case for the plant reduction approach where, at the moment of the reduction, the controller is not known yet. Among the methods following the closed-loop reduction philosophy we mention the work of Ceton et al. [22] (See Section 8.2.2), which is in fact the closed-loop extension of the work of Moore [98]. The aim of this method is basically to take into account the influence of the plant when reducing the controller, and is referred to as closed-loop balanced truncation (CLBT). In the framework of closed-loop LQG-controller reduction we also mention the work of Verriest [128] and Jonckheere [76]. The idea there is to balance the solutions of the Kalman filter and the control Riccati equations.

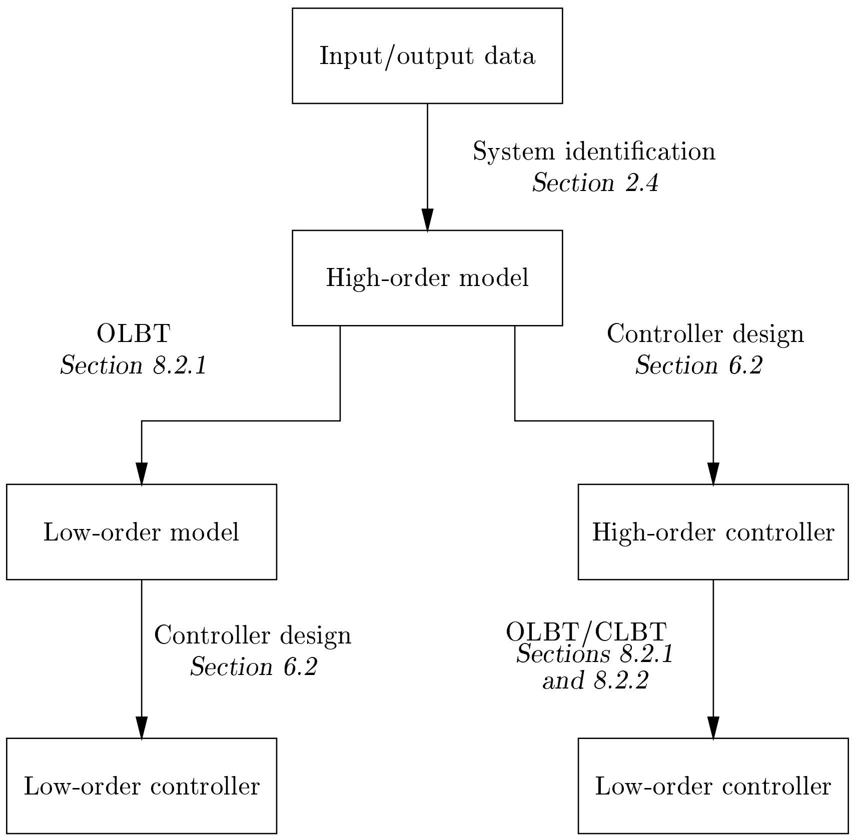


Figure 8.2: **One can distinguish between mainly two categories of classical controller reduction techniques. Both start from a high-order model of the plant to be controlled. The first method (plant reduction) starts by reducing the plant model which is then used to design a low-order controller. The plant reduction step is typically an open-loop reduction method (See Section 8.2.1) since the effect of the controller is not considered. A very popular method is open-loop balanced truncation [98] (OLBT). The second method on the other hand (controller reduction) first uses the high-order plant model to design a controller. This controller, which will be of the same order as the high-order plant model, is then reduced to a low-order controller. Of course, this reduction could be done with an open-loop reduction method. A better method however is closed-loop reduction because, contrary to open-loop reduction, it takes into account the fact that the controller is operating in a loop. In Section 8.2.2 we will briefly discuss the closed-loop balanced truncation technique of [22] (CLBT).**

In this section we recall the classical balanced plant/controller reduction theory. In a first step the open-loop model reduction framework [98] is recalled (Section 8.2.1). Further the closed-loop balanced controller reduction scheme of [22] is briefly presented (Section 8.2.2).

8.2.1 Open-loop balanced model reduction

Open-loop balanced model reduction was introduced by Moore [98] in the early eighties. Even now it is still a very popular model reduction method, probably due to its simplicity and the fact that a bound on the reduction error can be easily calculated [61]. The intuition behind balancing is based on modal analysis of the system. The modes that are well observable have an important contribution to the output in the sense that a lot of the energy contained in that mode is transferred to the output of the system. These modes should therefore be retained when reducing the system. Vice versa, the modes that are well controllable need little energy from the input to be controlled and should also be retained. The problem one will typically run into when the system is not balanced, is that the modes that are well observable do not necessarily have a good controllability, and vice versa. The observability and controllability of the system modes can be quantified by analyzing the observability Grammian $\mathcal{O}$ and the controllability Grammian $\mathcal{C}$ respectively. These are the solution of the following Lyapunov equations:

$$\mathcal{O} = A^T \mathcal{O} A + C^T C, \tag{8.4}$$
$$\mathcal{C} = A \mathcal{C} A^T + B B^T. \tag{8.5}$$

The solution to this problem consists in balancing the system, i.e. finding a similarity transformation $[A, B, C] \xrightarrow{\widetilde{T}} [\widetilde{A}, \widetilde{B}, \widetilde{C}]$ such that, in the new state-space basis, the observability and the controllability Grammians are equal and diagonal:

$$\widetilde{\mathcal{O}} = \widetilde{\mathcal{C}} = \Sigma$$

where $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ is a diagonal matrix with the diagonal elements sorted in decreasing order. In this case the observability and the controllability of mode i, quantified by the corresponding diagonal elements σ_i, is the same. To reduce the controller from order n to a lower order, it then suffices to truncate the state-space matrices of the balanced system such that the states corresponding to the largest σ_i-values are retained:

$$\left[\begin{array}{cc|c} \tilde{A}^1 & \tilde{A}^{12} & \tilde{B}^1 \\ \tilde{A}^{21} & \tilde{A}^2 & \tilde{B}^2 \\ \hline \tilde{C}^1 & \tilde{C}^2 & 0 \end{array}\right] \rightarrow \left[\begin{array}{c|c} \tilde{A}^1 & \tilde{B}^1 \\ \hline \tilde{C}^1 & 0 \end{array}\right].$$

One way to obtain a reduced-order controller is to simply use an open-loop reduction method. In this case, the controller is considered as any system of which the order has to be reduced, regardless of the process that it is controlling. The inputs of the controller are the reference signal r_k and the plant output y_k, while the outputs of the controller are the controlled inputs u_k of the plant (see Figure 8.3).

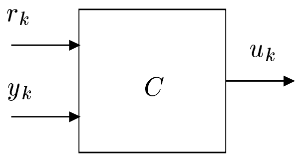


Figure 8.3: **With open-loop controller reduction, the controller is reduced regardless of the plant it has to control. The typical problem one will encounter in this case is spill-over.**

8.2.2 Closed-loop balanced controller reduction

The problem, when reducing a controller as described above, is that one completely neglects the fact that the controller is operating in a loop. As already mentioned, although the reduced-order controller might mimic the behaviour of the full-order controller in a very accurate way, the behavior of the closed-loop system can be completely destroyed. This phenomenon is often referred to as *spill-over*.

Since the main constraint on the controller reduction is that the performance specifications of the closed-loop should be maintained, one might think of balancing the Grammians of the closed-loop system. The state-space equations of the closed-loop system, in the case of a plant (6.1)-(6.3) and a controller (8.2)-(8.3), are given by:

$$\begin{aligned} \begin{pmatrix} x_{k+1} \\ x_{k+1}^c \end{pmatrix} &= A_{cl} \begin{pmatrix} x_k \\ x_k^c \end{pmatrix} + B_{cl} \begin{pmatrix} r_k \\ e_k \end{pmatrix}, && (8.6) \\ y_k &= C_{cl} \begin{pmatrix} x_k \\ x_k^c \end{pmatrix} + D_{cl} \begin{pmatrix} r_k \\ e_k \end{pmatrix} && (8.7) \end{aligned}$$

where

$$\begin{bmatrix} A_{cl} & B_{cl} \\ C_{cl} & D_{cl} \end{bmatrix} = \left[\begin{array}{cc|cc} A & BC_c & BD_c^r & K \\ B_c^y C & A_c + B_c^y DC_c & B_c^y DD_c^r + B_c^r & B_c \\ \hline C & DC_c & DD_c^r & I \end{array} \right]. \quad (8.8)$$

The corresponding closed-loop observability and controllability Grammians are the solutions of the following Lyapunov equations:

$$\begin{aligned}\mathcal{O}_{cl} &= A_{cl}^T \mathcal{O}_{cl} A_{cl} + C_{cl}^T C_{cl}, \\ \mathcal{C}_{cl} &= A_{cl} \mathcal{C}_{cl} A_{cl}^T + B_{cl} B_{cl}^T,\end{aligned}$$

and can be split up according to the states of the plant (p) and the controller (c):

$$\mathcal{O}_{cl} = \begin{pmatrix} \mathcal{O}_{pp} & \mathcal{O}_{pc} \\ \mathcal{O}_{pc}^T & \mathcal{O}_{cc} \end{pmatrix}, \qquad \mathcal{C}_{cl} = \begin{pmatrix} \mathcal{C}_{pp} & \mathcal{C}_{pc} \\ \mathcal{C}_{pc}^T & \mathcal{C}_{cc} \end{pmatrix}.$$

$\mathcal{O}_{cc}$ and $\mathcal{C}_{cc}$ are the parts of the closed-loop Grammians corresponding to the controller and are directly linked to the observability and the controllability of the controller states. Although it is perfectly possible to find a state-space basis of the closed-loop system such that $\mathcal{O}_{cl}$ and $\mathcal{C}_{cl}$ are balanced, it would not be possible to find the reduced-order controller from the resulting balanced state-space matrices. Indeed, due to the cross-terms $\mathcal{O}_{pc}$ and $\mathcal{C}_{pc}$, the states of the plant and the controller would be mixed up which would make it impossible to separate the parts related to the controller and those to the plant.

To get around this problem Ceton et al. [22] proposed a method where, instead of balancing the entire Grammians $\mathcal{O}_{cl}$ and $\mathcal{C}_{cl}$ of the closed-loop system, only those parts of the Grammians corresponding to the states of the controller are balanced. This is the whole idea of closed-loop balanced truncation (CLBT): find a similarity transformation $\widetilde{T}$ of the controller $[A_c, B_c, C_c] \xrightarrow{\widetilde{T}} [\widetilde{A}_c, \widetilde{B}_c, \widetilde{C}_c]$ such that the closed-loop Grammians $\widetilde{\mathcal{O}}_{cl}$ and $\widetilde{\mathcal{P}}_{cl}$ have the following form:

$$\begin{pmatrix} \widetilde{\mathcal{O}}_{pp} & \widetilde{\mathcal{O}}_{pc} \\ \widetilde{\mathcal{O}}_{pc}^T & \Sigma \end{pmatrix} = \begin{pmatrix} \widetilde{\mathcal{C}}_{pp} & \widetilde{\mathcal{C}}_{pc} \\ \widetilde{\mathcal{C}}_{pc}^T & \Sigma \end{pmatrix}$$

where Σ is again a diagonal matrix.

8.3 Balanced subspace system identification

A nice feature of subspace system identification is that a model can be obtained that is directly in balanced form. Therefore, contrary to most other system identification methods, no additional balancing operation is necessary after the identification. The link between subspace system identification and balanced model reduction has been made by several authors. The earliest result stems from Moonen and Ramos [97] where it was shown that, for a purely deterministic system (2.4)-(2.5), it is possible to obtain a balanced state-space model directly from the data. A nice generalization of this result for systems with deterministic as well as stochastic inputs (2.9)-(2.10) was presented in [120], [123].

The operation of balancing a system is in fact nothing else than looking for a state-space basis of the system such that the observability and the controllability Grammians are equal and diagonal. It can be seen that, in the case of subspace identification, the step in the algorithm where the state-space basis of the system is fixed is the SVD-step (See Figure 2.7). At this point the weighting matrices W_1 and W_2 come into the play, since they directly influence the state-space basis of the model. The whole idea of balancing, within subspace system identification, is that of choosing the weighting matrices W_1 and W_2 in such a way that the state-space basis of the resulting model is balanced. How these matrices can be calculated is given in the following theorem which is a modified version of the one that can be found in [123] (p. 145) and [120].

Theorem 8.3.1 (Balanced subspace system identification)

If in the subspace system identification algorithm of Figure 2.7

- *the number of block rows in the past and future data block-Hankel matrices (2.22)-(2.23) is infinite ($i = \infty$) and*
- *the weighting matrices W_1 and W_2 of the subspace identification algorithm are chosen such that:*

$$\begin{aligned} W_1 &= I, \\ W_2 &= \begin{pmatrix} H^d & H^s \\ I & 0 \end{pmatrix}, \end{aligned}$$

then the resulting model A, B, C, D, Q, R, S is open-loop balanced.

Proof: See Appendix A.6 ■

The matrices W_1 and W_2 in Theorem 8.3.1 are a function of the block-Toeplitz matrices H^d and H^s containing the impulse response of the deterministic and the stochastic input respectively. As we mentioned in Remark 2.5.1, if the number of block rows i in the data block-Hankel matrices is infinite, H^d and H^s can be directly found from the LQ-decomposition of the subspace identification algorithm. Due to the way H^s is calculated it is a lower triangular matrix with non-zero elements on the diagonal. Therefore, both W_1 and W_2 are of full rank, which is a necessary condition for the subspace identification algorithm of Figure 2.7.

8.4 Subspace-based balanced controller reduction

As we mentioned in the introduction of this chapter, the problem with the subspace-based LQG-controller of Figure 6.5 is that its order is equal to the number of block-rows M in the past data matrices Y_p and U_p. From Theorem 6.5.2 and from the example in Section 6.6, it is clear that the larger M is chosen, the closer the controller will be to the optimal LQG-controller. As a result, the order of the subspace-based controller (8.1) can be very high in practise.

The objective of this section is twofold:

Section 8.4.1: The subspace-based LQG-algorithm of Figure 6.5 is modified such that the resulting LQG-controller is in the low-order state-space form (8.2)-(8.3) instead of the high-order form (8.1).

Sections 8.4.2 and 8.4.3: Next it will be shown how W_1 and W_2 have to be chosen in order to have a balanced controller. In Section 8.3 we have recalled how, by making an appropriate choice of the weighting matrices W_1 and W_2 in the subspace identification algorithm of Figure 2.7, it is possible to obtain a model that is balanced in the sense of [98]. Here, we extend this idea by introducing weighting matrices in the subspace-based LQG-design algorithm (Figure 6.5) in a similar way as it is done in subspace system identification. By doing so, it is possible to find an LQG-controller that is open-loop balanced (Section 8.4.2) or closed-loop balanced (Section 8.4.3), directly from the available input/output data.

This once again clearly illustrates the far going analogies between subspace system identification and subspace-based LQG-controller design (See also Figure 8.4).

8.4.1 Controller reduction

Let us now analyze how the three basic steps of the subspace-based LQG-algorithm of Figure 6.5 should be modified in order to find a controller with a state-space form (8.2)-(8.3) instead of the subspace-based input/output controller form (8.1).

LQ-decomposition: From (6.11) it can be derived that:

$$\widehat{U}_f = -(\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q} L_w W_p$$

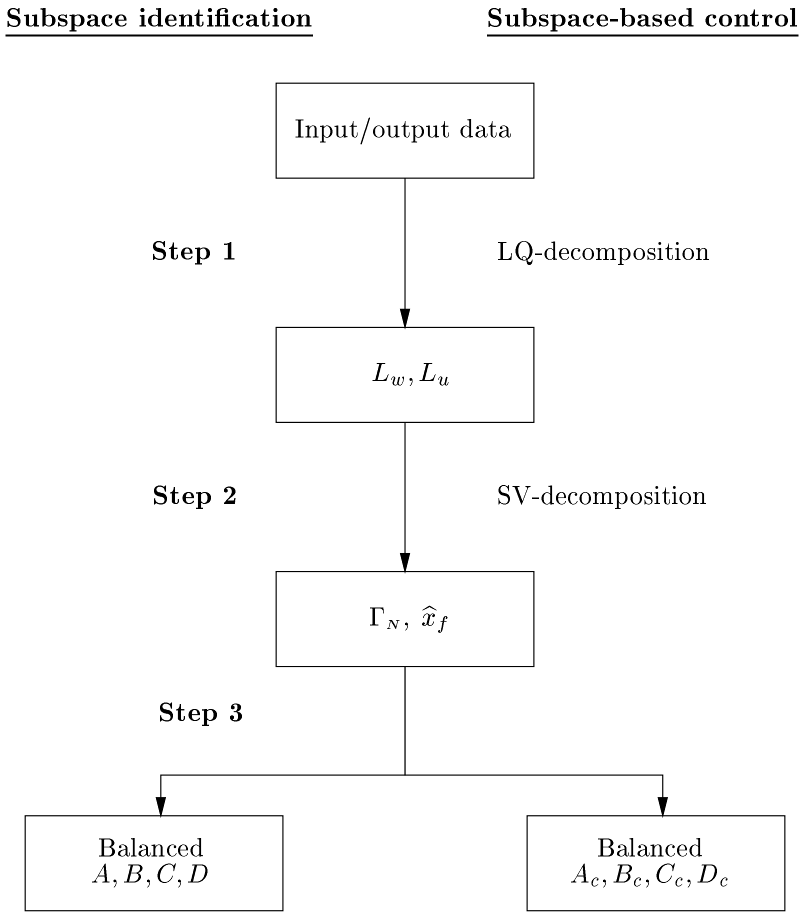


Figure 8.4: **This figure illustrates the analogy between subspace system identification and model-free subspace-based LQG-design. One can clearly distinguish the same three steps in both approaches, namely a projection or LQ-decomposition, a singular value decomposition and a final step where the system parameters A, B, C, D or the controller parameters A_c, B_c, C_c, D_c are extracted. The first two steps in the subspace identification and the subspace-based control algorithm are identical. Only the third step, i.e. the way the controller or system parameters are extracted, is different.**

which, using Theorem 2.4.3 and defining $W_c \stackrel{\text{def}}{=} (\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q}$, can also be written as:

$$\widehat{U}_f = -W_c \Gamma \widehat{X}_f$$

where $\widehat{X}_f$ is a Kalman filter estimate of the plant state sequence X_f and

Γ is the plant observability matrix. $\widehat{U}_f$ should not be confused with the future input data block-Hankel matrix U_f defined in (2.22). $\widehat{U}_f$ is in fact an artificially constructed matrix where every column can be interpreted as the autonomous response of the controller ($y_k = 0, r_k = 0$) with its initial state equal to the corresponding column of $\widehat{X}_f$. On the other hand, we know from the controller matrix input/output equation that the autonomous response of the controller can also be expressed as ($R_f = 0, Y_f = 0$):

$$\widehat{U}_f = \Gamma_c X_f^c.$$

Observing that, in the case of an LQG-controller, the states of the controller are in fact the Kalman filter states of the plant ($X_f^c = \widehat{X}_f$), we see that the observability matrix Γ_c and that of the plant Γ are related by the following equation:

$$\Gamma_c = -W_c \Gamma.$$

Finally, we have that the first step of the reduced-order subspace-based LQG-algorithm (Figure 6.5) still consists in calculating the following projection:

$$Y_f / \begin{pmatrix} W_p \\ U_f \end{pmatrix} \overset{\text{LQ}}{=} L_w W_p + L_u U_f,$$

and deriving the parameters L_w and L_u. Also calculate $W_c \overset{\text{def}}{=} (\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q}$.

SV-decomposition: From the weighted singular value decomposition:

$$W_1 W_c L_w W_2 \overset{\text{SVD}}{=} \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}, \tag{8.9}$$

the extended observability matrix of the controller Γ_c and the estimated Kalman filter state sequence $\widehat{X}_f$ can be directly found:

$$\begin{aligned} \Gamma_c &= -W_1^{-1} U_1 S_1^{1/2}, \\ \widehat{X}_f &= S_1^{1/2} V_1^T W_2^{-1} W_p. \end{aligned} \tag{8.10}$$

In the next sections we will come back to the problem of choosing W_1 and W_2. As we have seen in Theorem 2.4.3, every column of $\widehat{X}_f$ is a nonsteady-state Kalman filter estimate of the plant state, solution of the recursive Kalman filter equations (2.9)-(2.10) which can be rewritten as:

$$\widehat{x}_{k+1} = A_f \widehat{x}_k + B_f^u u_k + B_f^y y_k \tag{8.11}$$

where

$$A_f \overset{\text{def}}{=} A - KC,$$

$$\begin{aligned} B_f^u &\stackrel{\text{def}}{=} B - KD, \\ B_f^y &\stackrel{\text{def}}{=} K. \end{aligned}$$

In analogy with the matrix input/output equation (2.29) of the plant, one can find a similar equation for the above Kalman filter equation:

$$\widehat{X}_f = A_f^{M-1}\widehat{X}_p + \Delta_f W_p \tag{8.12}$$

where Δ_f is the extended reversed controllability matrix of the Kalman filter corresponding to the outputs y_k and the inputs u_k, and is defined as:

$$\begin{aligned} \Delta_f \stackrel{\text{def}}{=} \big(& A_f^{M-1}B_f^y \quad \ldots \quad A_f B_f^y \quad B_f^y \mid \\ & A_f^{M-1}B_f^u \quad \ldots \quad A_f B_f^u \quad B_f^u \big). \end{aligned}$$

If the number of block rows M in the past data matrix W_p is infinite, and knowing that the Kalman filter (8.11) of a linear time-invariant system (6.1)-(6.3) is always stable (A_f Hurwitz), we have that the estimated state sequence (8.12) is the steady-state Kalman filter state sequence and can be written as:

$$\widehat{X}_f = \Delta_f W_p.$$

Comparing this equation with the Kalman filter sequence (8.10) we have that the extended reversed controllability matrix of the Kalman filter Δ_f can be easily found from the singular value decomposition:

$$\Delta_f = S_1^{1/2} V_1^T W_2^{-1}.$$

Controller state-space matrices: Let us now try to recover the controller state-space matrices A_c, B_c, C_c, D_c. We will proceed in two steps: first A_c and C_c are determined, then B_c and D_c.

A_c and C_c: Using the shift invariance property of the controller observability matrix Γ_c, the state-space matrices A_c and C_c can be found:

$$\begin{aligned} A_c &= \Gamma_c(1:end-m,:)^{\dagger}\Gamma_c(m+1:end,:), \\ C_c &= \Gamma_c(1:m,:). \end{aligned}$$

B_c and D_c: From the Kalman filter equation (8.11) and the LQR control law $u_k = -L_c\widehat{x}_k + r_k$, it is easy to see that the state-space representation (8.2)-(8.3) of an LQG-controller can be written as:

$$\begin{aligned} x_{k+1}^c &= (A_f - B_f^u L_c)x_k^c + B_f^u r_k + B_f^y y_k, \\ u_k &= -L_c x_k^c + r_k. \end{aligned}$$

It follows that:

$$\begin{aligned}
B_c^r &= B_f^u = \Delta_f(:, end - m + 1 : end), \\
B_c^y &= B_f^y = \Delta_f(:, (M-1)l + 1 : Ml), \\
D_c^r &= 0_m, \\
D_c^y &= I_{m \times l}.
\end{aligned}$$

The entire subspace-based reduced-order LQG-control algorithm is given in Figure 8.5.

8.4.2 Subspace-based open-loop balanced controller reduction

Until now we did not mention how the weighting matrices W_1 and W_2 could be chosen in a relevant way. As was recalled in Section 8.3, it is known [123], [120] that, in the case of subspace system identification, W_1 and W_2 are in fact interesting design parameters since they influence the state-space basis of the identified model. It is even possible to choose W_1 and W_2 such that the model is open-loop balanced. In this section we show that, in an analogue way as for subspace system identification, it is possible to choose W_1 and W_2 such that the resulting controller is open-loop balanced.

Theorem 8.4.1 (Subspace-based OLBT)
If in the subspace-based LQG-algorithm of Figure 8.5

- *the number of block rows in the past and future data block-Hankel matrices is infinite ($M, N = \infty$) and*
- *the weighting matrices W_1 and W_2 are chosen such that:*

$$W_1 = I, \tag{8.13}$$

$$W_2 = \begin{pmatrix} I & 0 \\ H_c^y & H_c^r \end{pmatrix}, \tag{8.14}$$

then the resulting controller (8.2)-(8.3) is open-loop balanced in the sense of Moore [98].

Proof: See Appendix A.7. ■

Subspace-based reduced-order LQG-controller algorithm

LQ-decomposition: Construct the data block-Hankel matrices Y_f, U_f and W_p from the available input and output measurements, and determine L_w and L_u by calculating the projection (preferably with a LQ-decomposition):

$$Y_f / \begin{pmatrix} W_p \\ U_f \end{pmatrix} \stackrel{\text{LQ}}{=} \begin{pmatrix} L_w & L_u \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix}.$$

Also calculate $W_c = (\bar{R} + L_u^T \bar{Q} L_u)^{-1} L_u^T \bar{Q}$.

SV-decomposition: Calculate the following singular value decomposition:

$$W_1 W_c L_w W_2 \stackrel{\text{SVD}}{=} U_1 S_1 V_1^T$$

and derive the controller extended observability matrix Γ_c and the Kalman filter extended reversed controllability matrix Δ_f:

$$\begin{aligned} \Gamma_c &= -W_1^{-1} U_1 S_1^{1/2}, \\ \Delta_f &= S^{1/2} V_1^T W_2^{-1}. \end{aligned}$$

Controller matrices: Calculate the reduced-order LQG-controller state-space matrices:

$$\begin{aligned} A_c &= \Gamma_c(1 : end - m, :)^{\dagger} \Gamma_c(m+1 : end, :), \\ B_c &= \begin{pmatrix} \Delta_f(:, (M-1)l + 1 : Ml) & \Delta_f(:, end - m + 1 : end) \end{pmatrix}, \\ C_c &= \Gamma_c(1 : m, :), \\ D_c &= \begin{pmatrix} 0_{m \times l} & I_m \end{pmatrix}. \end{aligned}$$

Figure 8.5: **Subspace-based algorithm for designing a low-order LQG-controller starting from an input/output data set of the unknown system.**

In the above theorem, H_c^y and H_c^r are the block-Toeplitz matrices containing the impulse response of the controller to y_k and r_k, and can be written as:

$$H_c^y = \begin{pmatrix} h_0^y & 0 & 0 & \dots & 0 \\ h_1^y & h_0^y & 0 & \dots & 0 \\ h_2^y & h_1^y & h_0^y & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ h_{M-1}^y & h_{M-2}^y & \dots & h_1^y & h_0^y \end{pmatrix}$$

and

$$H_c^r = \begin{pmatrix} h_0^r & 0 & 0 & \dots & 0 \\ h_1^r & h_0^r & 0 & \dots & 0 \\ h_2^r & h_1^r & h_0^r & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ h_{M-1}^r & h_{M-2}^r & \dots & h_1^r & h_0^r \end{pmatrix}.$$

The interesting thing is that the impulse response coefficients h_i^y and h_i^r can be directly found from the LQG-control law (6.12). The N future control inputs given the last M past measured inputs and outputs are given by:

$$\widehat{u}_f = -\bar{L}_w \begin{pmatrix} y_p \\ u_p \end{pmatrix}$$

where $\bar{L}_w$ was defined as $\bar{L}_w \stackrel{\text{def}}{=} (\bar{R}+L_u^T\bar{Q}L_u)^{-1}L_u^T\bar{Q}L_w$. Therefore, the impulse response coefficients of the controller to the output y_k and the reference input r_k are given by:

$$\begin{aligned} h_i^y &= -\bar{L}_w[(i-1)m+1:im,(M-1)l+1:Ml)], & h_0^y &= 0_{m\times l}, \\ h_i^r &= -\bar{L}_w[(i-1)m+1:im, end-m+1:end)], & h_0^r &= I_m. \end{aligned}$$

8.4.3 Subspace-based closed-loop balanced controller reduction

Theorem 8.4.2 (Subspace-based CLBT)
If in the subspace-based LQG-algorithm of Figure 8.5

- *the number of block rows in the past and future data block-Hankel matrices is infinite ($M, N = \infty$) and*
- *the weighting matrices W_1 and W_2 are chosen such that:*

$$W_1 = (I - H^d H_c^y)^{-1} H^d, \tag{8.15}$$

$$W_2 = \begin{pmatrix} -H_c^y & I \\ I & -H^d \end{pmatrix}^{-1} \begin{pmatrix} H_c^r & O \\ 0 & H^s \end{pmatrix} \tag{8.16}$$

then the resulting controller (8.2)-(8.3) is closed-loop balanced in the sense of [22].

Proof: See Appendix A.8. ■

As we mentioned in Remark 2.5.1 of Section 2.5, if the number of block rows i in the data block-Hankel matrices is infinite, H^d and H^s can be directly found from the LQ-decomposition. H_c^y and H_c^r can be calculated as in Theorem 8.4.1.

In this chapter, an extension of the subspace-based LQG-algorithm of Figure 6.5 was be presented that directly calculates a reduced-order LQG-controller. It was also shown how a specific choice of the weighting matrices W_1 and W_2 can influence this reduction. By choosing these weighting matrices in an appropriate way, it is possible to find a balanced LQG-controller, directly from the input/output data.

Chapter 9

Model-free subspace-based bilinear predictive control

The main result of Part I of this thesis was that linear subspace identification was generalized, to a certain extend, to bilinear subspace identification methods. In Part II a new class of LQG-algorithms was presented, based on techniques of subspace identification. In this chapter we combine the bilinear subspace identification theory of Chapters 3 and 4 on the one hand, and the linear model-free subspace-based LQG-algorithm of Chapter 6 on the other hand. The result is a new class of algorithms for designing predictive controllers of unknown bilinear systems, directly from measurements of the inputs and the outputs of this system.

9.1 Introduction – problem description

In this chapter we will be interested in the design of predictive controllers for unknown bilinear systems, directly from measured inputs and outputs of that system. The class of bilinear systems we consider can be described by the following state-space equations:

$$x_{k+1} = Ax_k + Nu_k \otimes x_k + Bu_k + w_k, \tag{9.1}$$

$$y_k = Cx_k + Du_k + v_k \tag{9.2}$$

where the state $x_k \in \mathbb{R}^n$, the input $u_k \in \mathbb{R}^M$ and the output $y_k \in \mathbb{R}^l$. The innovations $e_k \in \mathbb{R}^l$ are a stationary, ergodic, zero-mean, Gaussian, white

Model-free bilinear predictive control problem

Given *a set of open-loop measurements of the inputs* u_k *and the outputs* y_k *of the unknown bilinear system (9.1)-(9.3),* **find** *the control sequence* $\widehat{u}_f = (\widehat{u}_1, \ldots, \widehat{u}_N)$ *that minimizes the cost function* J *over the forward horizon* N*:*

$$J = \sum_{k=1}^{N} \{(\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k\} \qquad (9.4)$$

Figure 9.1: **Model-free bilinear predictive control problem.**

signal with covariance matrix:

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q \\ v_q \end{pmatrix}^T] = \begin{pmatrix} Q & S \\ S^T & R \end{pmatrix} \delta_{pq}. \qquad (9.3)$$

The controller we search to design minimizes the following cost function over the forward horizon N:

$$J = \sum_{k=1} (\widehat{y}_k - r_k)^T Q_k (\widehat{y}_k - r_k) + \widehat{u}_k^T R_k \widehat{u}_k.$$

$r_k \in \mathbb{R}^l$ is the reference output and $\widehat{y}_k \in \mathbb{R}^l$ the k-step-ahead predicted output. The input and output positive semidefinite weighting matrices $Q_k \in \mathbb{R}^{l \times l}$ and $R_k \in \mathbb{R}^{m \times m}$ are chosen by the user. The formal problem definition of this chapter is depicted in Figure 9.1.

We have seen in Chapter 6 that for linear systems, the classical way to solve the above problem is to split it up into a system identification, an observer (Kalman filter) design and an optimal state feedback design. Very few results are available where these three problems are treated as one. It comes as no surprise that, in the case of bilinear systems, even less results are available on this matter. The problem of designing a predictive controller for a known bilinear system has been treated in the literature by several authors. Most of these results however only consider special classes of bilinear systems. For instance [134] and [51] treat SISO systems that are described by the following input/output equation:

$$y_k = \sum_{i=1}^{p} \{a_i y_{k-i} + b_i y_{k-i} u_{k-i} + c_i u_{k-i}\} \qquad (9.5)$$

which is a more restricted class than the general MIMO state-space description (9.1)-(9.2). The results that are closest to the problem treated here are [113],

[100], [135], where an adaptive minimum variance controller ($N = 1$) for bilinear systems (9.5) is derived.

9.2 Model-free subspace-based predictive control for bilinear systems

In this section, a new solution for the problem defined in Figure 9.1 is presented. The method is closely related to the bilinear subspace identification theory of Chapters 3 and 4 on the one hand, and the linear model-free subspace-based LQG-theory of Chapter 6 on the other. Here, these results will be combined to design a predictive controller for a bilinear system directly from data, i.e. without prior knowledge of the system.

In matrix form, the performance criterion (9.4) can be written as:

$$J = (\widehat{y}_f^l)^T \bar{Q} \widehat{y}_f^l + (u_f^l)^T \bar{R} u_f^l$$

where

$$\widehat{y}_f^l \stackrel{\text{def}}{=} \begin{pmatrix} \widehat{y}_N \\ \widehat{y}_{N-1} \\ \vdots \\ \widehat{y}_1 \end{pmatrix} \quad , \quad u_f^l \stackrel{\text{def}}{=} \begin{pmatrix} u_N \\ u_{N-1} \\ \vdots \\ u_1 \end{pmatrix}$$

and the matrices $\bar{Q} \in \mathbb{R}^{Nl \times Nl}$ and $\bar{R} \in \mathbb{R}^{Nm \times Nm}$ are defined as:

$$\bar{Q} \stackrel{\text{def}}{=} \begin{pmatrix} Q_N & 0 & \dots & 0 \\ 0 & Q_{N-1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Q_1 \end{pmatrix} \quad , \quad \bar{R} \stackrel{\text{def}}{=} \begin{pmatrix} R_N & 0 & \dots & 0 \\ 0 & R_{N-1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R_1 \end{pmatrix}.$$

We have seen in Chapter 3 that the predicted output sequence $\widehat{y}_f^l$ can be written as (4.6):

$$\widehat{y}_f^l = L_w w_p + L_u u_f + L_{uw} u_f \otimes w_p. \tag{9.6}$$

where

$$w_p = \begin{pmatrix} y_p \\ u_p \end{pmatrix}$$

and the generalized past inputs u_p and outputs y_p and the generalized future inputs u_f are defined as in (3.25):

$$y_p = y_{-M+1|0} = \begin{pmatrix} y_0 \\ y_{-M+1|-1} \\ u_0 \otimes y_{-M+1|-1} \end{pmatrix}, \tag{9.7}$$

$$u_p = u_{-M+1|0} = \begin{pmatrix} u_0 \\ u_{-M+1|0} \\ u_0 \otimes u_{-M+1|-1} \end{pmatrix} \quad (9.8)$$

$$u_f = u_{1|N} = \begin{pmatrix} u_N \\ u_{1|N-1} \\ u_N \otimes u_{1|N-1} \end{pmatrix} \quad (9.9)$$

and the initial conditions $y_{-M+1|-M+1} = y_{-M+1}$, $u_{-M+1|-M+1} = u_{-M+1}$ and $u_{1|1} = u_1$. As we have seen in Chapter 4, the parameters L_w, L_u and L_{uw} can be easily found from the LQ-decomposition (4.7).

From the definitions of u_f and u^l_f, it can be directly seen that it is possible to split up u_f as:

$$u_f = \begin{pmatrix} u^l_f \\ u^b_f \end{pmatrix}$$

where u^b_f is a vector consisting of specific products between the elements of u^l_f. Therefore, there exists a function f that relates u_f and u^l_f in the following way:

$$u_f = f(u^l_f). \quad (9.10)$$

The predicted output sequence $\widehat{y}^l_f$ can now be expressed as:

$$\widehat{y}^l_f = L_w w_p + L_u f(u^l_f) + L_{uw} f(u^l_f) \otimes w_p.$$

By substituting this expression of $\widehat{y}^l_f$ in the cost function (9.4) we have:

$$\begin{aligned} J(u^l_f) = {} & (L_w w_p + L_u f(u^l_f) + L_{uw} f(u^l_f) \otimes w_p)^T \bar{Q} \\ & (L_w w_p + L_u f(u^l_f) + L_{uw} f(u^l_f) \otimes w_p) + (u^l_f)^T \bar{R} u^l_f. \end{aligned} \quad (9.11)$$

The solution $u^{l,*}_f$ of the bilinear predictive control problem is the input sequence that minimizes the above cost function:

$$u^{l,*}_f = \arg\min J(u^l_f).$$

A major difficulty we encounter is that, whereas in the linear subspace-based LQG-control problem J is a quadratic function, here J is a nonlinear and non-convex function of u^l_f. In order to solve this problem we propose an iterative Gauss-Newton optimization method which is guaranteed to converge to at least a local minimum. However, this requires the calculation of the first and second derivatives, i.e. the gradient G and the Hessian H respectively, of the cost function J with respect to the control input vector u^l_f. Although we could not find an analytical expression for G or H, it is fairly easy to calculate them exactly in Matlab. The final Gauss-Newton optimization algorithm is

Newton optimization of (9.4)

Initialization: Find an initial estimate $u_f^{l,1}$ of the optimal solution $u_f^{l,*}$. One could for instance use the solution of the linear predictive controller as described in Figure 6.5.

Iteration k:

Step 1: Calculate the gradient G and the Hessian H of the cost function J at $u_f^{l,k}$:

$$\begin{aligned} G &= \frac{\delta J(u_f^{l,k})}{\delta u_f^l}, \\ H &= \frac{\delta J^2(u_f^{l,k})}{\delta (u_f^l)^2}. \end{aligned}$$

Step 2: Find the new solution:

$$u_f^{l,k+1} = u_f^{l,k} - H^{-1}G.$$

Iterate until the solution has converged to a local optimum $u_f^{l,*}$.

Figure 9.2: **Newton algorithm for the optimization of (9.11). In order to start the iteration, the solution of a linear subspace predictive controller (Figure 6.5) is used as an initial estimate $u_f^{l,1}$ of the optimal solution $u_f^{l,*}$. First the gradient G and the Hessian H of the cost function (9.11) are calculated. Although an analytical expression of G and H could not be found, it is straightforward to calculate them based on the definition (9.9) of u_f and the expression of the cost function (9.11). In a second step G and H are used to calculate the new solution $u_f^{l,k}$.**

summarized in Figure 9.2. Although the algorithm is guaranteed to converge to at least a local minimum, if the initial solution $u_f^{l,1}$ is too far away from the optimal solution $u_f^{l,*}$, this local minimum is not necessarily equal to the global minimum. Multiple simulations showed that if the input weight matrix R is too small, which implies an aggressive control, the algorithm gives a biased solution.

9.3 Algorithm

The final bilinear subspace predictive control algorithm is given in Figure 9.3.

Model-free bilinear subspace-based predictive control algorithm

LQ-decomposition: Construct the data block-Hankel matrices $Y_f^l, U_f, W_p = \binom{Y_p}{U_p}$ and $U_f \odot W_p$ from the data and determine L_w, L_u and L_{uw} from the the following projection (preferably with an LQ-decomposition):

$$Y_f / \begin{pmatrix} W_p \\ U_f \\ U_f \odot W_p \end{pmatrix} \stackrel{\text{LQ}}{=} L_w W_p + L_u U_f + L_{uw} U_f \odot W_p.$$

SV-decomposition: Approximate L_w by a rank-deficient matrix with the following singular value decomposition:

$$L_w \stackrel{\text{SVD}}{=} U_1 S_1 V_1^T,$$

where the rank of L_w is chosen by inspecting the number of dominant singular values.

Controller: Calculate the optimal control sequence $u_f^{l,*}$ with the Newton algorithm of Figure 9.2 and implement the first input u_1. To implement the following control step construct the new control input vector w_p with the newly measured system output y_1 and repeat this step.

Figure 9.3: **Subspace-based algorithm for the predictive control of unknown bilinear systems. The first 2 steps, from which the controller parameters L_w, L_u and L_{uw} are derived, only have to be performed once. The last step represents the implementation of the controller and has to be calculated at every time-step.**

9.4 Example

The aim of this example is to compare the bilinear subspace predictive control algorithm of Figure 9.3 and the linear algorithm of Figure 6.5. The compari-

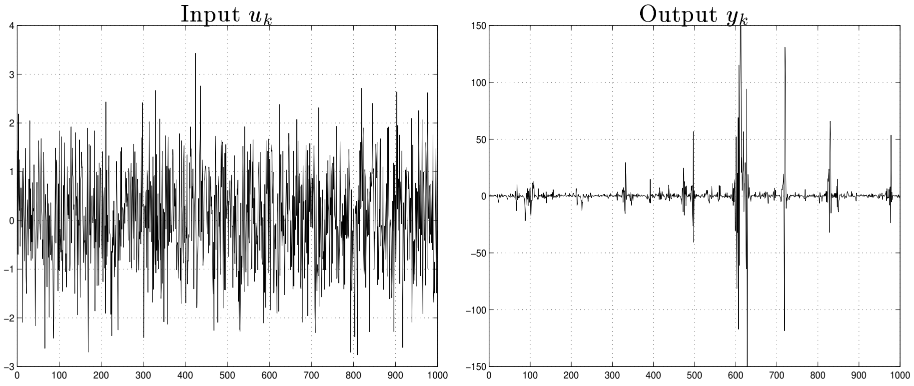


Figure 9.4: **The available input/output data set. The input is a white, Gaussian, zero-mean signal with variance 1.**

son is made by applying both algorithms on the same input/output data set. Moreover, the same input and output weighting matrices Q_i and R_i are used in the performance criterion (9.4).

The system we consider is a deterministic SISO second order bilinear system of the form (3.4)-(3.5) with the following state-space matrices:

$$\begin{aligned} A &= \begin{pmatrix} 0.5019 & -0.3240 \\ -0.3240 & 0.0786 \end{pmatrix}, \quad B = \begin{pmatrix} 0.4442 \\ 0.6206 \end{pmatrix}, \quad D = 0.2473, \\ N &= \begin{pmatrix} -0.7164 & -0.3857 \\ -1.8002 & -0.0355 \end{pmatrix}, \quad C = \begin{pmatrix} 0.9517 & 0.6400 \end{pmatrix}. \end{aligned}$$

The available input/output data set is given in Figure 9.4. This data set is then used in the linear algorithm of Figure 6.5 and the bilinear algorithm of Figure 9.3 to design a linear and a bilinear predictive controller respectively. The input and output matrices Q_i and R_i where chosen the same for both controllers, whereas the forward and the backward horizon for the linear controller are chosen larger ($M = 10, N = 10$) than for the bilinear controller ($M = 3, N = 2$). This comes down to choosing a linear controller of order 10 and a bilinear controller of order 3. Figure 9.5 shows the response of the closed-loop system, controlled with both the linear and the bilinear controller. Up to time 0 the system is still operating in open-loop, after which the controller is implemented. One can clearly see that the response of the system controlled with the bilinear controller is considerably better than that with the linear controller.

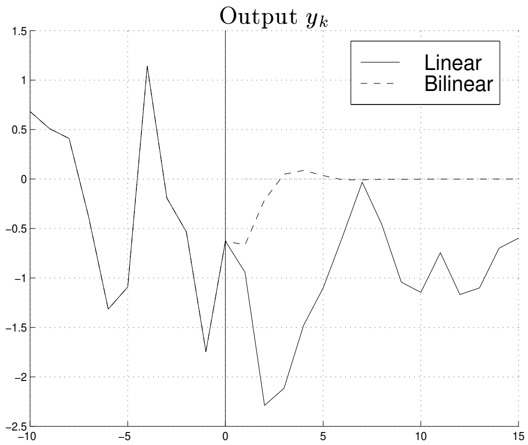


Figure 9.5: **Closed-loop response of the system controlled with a linear subspace-based LQG-controller (full line) and with a bilinear subspace predictive controller (dashed line). Up to time 0 the system is operating in open-loop. The input and output data up to that time-step are used to calculate the controllers. From time-step 1 on (vertical line) the loop is closed and the system evolves to the origin ($r_k = 0$). The user-defined weighting matrices Q_i and R_i were chosen the same in both controllers. One can clearly see that, although the order of the linear controller is considerably higher than for the bilinear controller (10 vs. 3), the bilinear controller gives a better closed-loop behavior than the linear controller.**

In this chapter we have combined the bilinear subspace identification theory of Chapters 3 and 4 on the one hand, and the linear model-free subspace-based LQG-algorithm of Chapter 6 on the other hand. The result is a new class of algorithms for the design of predictive controllers, directly from measurements of the inputs and the outputs of unknown bilinear systems.

Chapter 10

Conclusions and open problems

10.1 Conclusions

Part I: In the first part of this thesis, a new class of subspace system identification algorithms has been introduced. They are a direct generalization of the, by now, well-known linear subspace identification algorithms and can be used for the identification of bilinear systems. In the case the inputs of the identification data set are white, it was proven that most of the results of linear subspace identification also hold for bilinear systems. For instance one can also interpret the states estimated with the subspace algorithm as Kalman filter states of the bilinear system. If on the other hand the inputs are not white, the algorithm only gives consistent results if the number of block rows in the generalized data Hankel matrices is infinite. It was shown, by the hand of multiple simulation examples, that bilinear subspace identification can bring a significant improvement when compared to linear subspace identification.

Part II: In Part II of the thesis subspace identification algorithms have been analyzed from a control point of view. It was shown that there are a number of interesting analogies between subspace system identification and model-based optimal control. For instance, the concept of past and future is clearly present in subspace identification as well as in optimal control. Also the role of the Kalman filter state is of primary importance for both methods. These analogies have been analyzed and combined in such a way that a new algorithm was found for an efficient implementation of LQG-controllers. The algorithms are referred to as model-free subspace-based methods; model-free on the one hand because at no point in the algorithm an explicit model of the system to be controlled is calculated and subspace-based on the other hand because, just as for subspace system identification, the algorithm is also based on LQ- and SV-decompositions. The resulting algorithms allow for the calculation of an optimal controller of an unknown linear time-invariant system, directly from a set of open-loop measurements of that system. The algorithms were also extended to the case where the measurements were obtained under time-invariant feedback.

Also the problem of controller reduction has been analyzed in a subspace identification framework. It was shown that, by choosing certain specific weights in the SV-decomposition step of the algorithm, the resulting controller is directly in balanced form. According to the choice of the weights, the controller will be in open- or closed-loop balanced form.

10.2 Open problems

Although quite some efforts have been made to make the picture of this thesis as complete as possible, there are of course still a lot of open issues for future research.

Part I: Linear: Whereas in prediction error system identification there is a wealth of results on the statistical properties of the model parameters, very few results are available in the field of subspace identification [131], [101], [132], [15]. One of the main reasons why it is so difficult to analyze the statistical properties of subspace methods is that they do not minimize a specific error criterion. Hence, the interesting question of the optimality of subspace methods is also still open.

Bilinear: In the case the inputs are not necessarily white, the subspace algorithm of Chapter 4 gives biased results. The fundamental reason for this is that, contrary to linear systems, for bilinear systems we could not find a matrix, constructed out of inputs and the output measurements only, of which the rows span the state sequence of the system. Based on the results of Chapters 3 and 4 a new algorithm was recently proposed [23] that tries to solve this problem. It is only valid for bilinear systems with more outputs than states however, which is a significant restriction.

Part II: Linear:

- In Chapter 6, a subspace-based algorithm was presented for the calculation of LQG-controllers directly from open-loop data. Chapter 7 in turn focused on the extension to the case where the inputs and outputs were measured under closed-loop conditions. Although this algorithm can handle closed-loop data, there is still the restriction that the data must be measured under *time-invariant* feedback. This means that the problem of finding a fully adaptive subspace-based LQG-control algorithm, i.e. where the controller changes at every time-step, is still open.
- Although very important, the problems of stability and robustness of the subspace-based controllers are still open.
- As we mentioned in Chapter 6, there are strong links between MPC and the subspace-based LQG-controller of Figure 6.5. The main merit of MPC was to stress the importance and analyze the influence of input and/or output inequality constraints on the controller. Constraints are indeed very important from a engineering point of view, since they are directly linked to the safety and performance of a process. The advent of powerful methods in the field of linear matrix inequalities (LMI) and the subspace-based approach of the LQG-control problem presented in this thesis, create new perspectives for the constrained optimal control problem.

Bilinear: The bilinear subspace predictive control algorithm of Chapter 9 is of course very rudimentary. Although one of the purposes of the chapter was to complete the picture of this thesis, i.e. linear/bilinear - control/identification, it gives a new angle to attack the problem of identifying and controlling bilinear systems. It is clear that the algorithm needs to be analyzed more thoroughly. For instance, from a practical point of view, the convergence properties of the Newton optimization are very important. Also the stability issue is completely unsolved.

Bibliography

[1] Z. Aganovic and Z. Gajic. The succesive approximation procedure for finite-time optimal control of bilinear systems. *IEEE Trans. Automat. Control*, 39(9):1932–1935, 1994.

[2] M.S. Ahmed. Parameter estimation in bilinear systems by instrumental variable method. *Internat. J. Control*, 44(4):1177–1183, 1984.

[3] P.L. Ainsleigh. Observations on oblique projectors and pseudo-inverses. *IEEE Trans. Automat. Control*, 45(7):1886–1888, 1997.

[4] H. Akaike. Stochastic theory of minimal realization. *IEEE Trans. Automat. Control*, 19:667–674, 1974.

[5] H. Akaike. Markovian representation of stochastic processes by canonical variables. *SIAM J. Control Optim.*, 13(1):162–173, 1975.

[6] B.D.O. Anderson and Y. Liu. Controller reduction: concepts and approaches. *IEEE Trans. Automat. Control*, 34(8):802–812, 1989.

[7] B.D.O. Anderson and J.B. Moore. *Linear optimal control.* Prentice-Hall, Englewood Cliffs NJ, USA, 1971.

[8] K. Åström and P. Eykhoff. System identification - A survey. *Automatica*, 7:123–167, 1971.

[9] K.J. Åström. *Introduction to stochastic control theory.* Academic Press, New York, USA, 1970.

[10] K.J. Åström and B. Wittenmark. On self-tuning regulators. *Automatica*, 9:982–987, 1973.

[11] R.S. Baheti. Second-order correlation method for bilinear system identification. *IEEE Trans. Automat. Control*, 25(6):1141–1146, 1980.

[12] J.K. Baksalary and R. Kala. Two relations between oblique and Λ-orthogonal projections. *Linear Algebra and its Applications*, 24:99–103, 1979.

[13] A.R. Barron. Universal approximation bounds for superposition of a sigmoidal function. *IEEE Trans. on Information Theory*, 39(3):930–945, 1993.

[14] T. Bastogne, H. Noura, A. Richard, and J.M. Hittinger. Application of subspace methods to the identification of a winding process. In *Proc. of the 4th European Control Conference, ECC 97, July 1-4, Brussels, Belgium*, page ECC560, 1997.

[15] D. Bauer, M. Deistler, and W. Scherrer. The analysis of the asymptotic variance of subspace algorithms. In *Proc. of the 11th IFAC Symposium on System Identification, SYSID 97, July 8-11, Kitakyushu, Japan*, volume 3, pages 1087–1091, 1997.

[16] A. Benallou, D.A. Mellichamp, and D.E. Seborg. Optimal stabilizing controllers for bilinear systems. *Internat. J. Control*, 48(4):1487–1501, 1988.

[17] R. Bitmead, M. Gevers, and V. Wertz. *Adaptive Optimal Control - The Thinking Man's GPC*. Prentice Hall International, 1990.

[18] G.E. Box and G.M. Jenkins. *Time series analysis, forecasting and control.* Holden-Day series in time series analysis and digital processing. Holden-Day, Oackland, Revised edition, 1976.

[19] S. Boyd, Y.S. Tang, and L.O. Chua. Measuring Volterra kernels. *IEEE Trans. Circuits and Systems*, 30(8):571–577, 1983.

[20] R.W. Brockett. Volterra series and geometric control theory. *Automatica*, 12:167–176, 1976.

[21] C. Bruni, G. Di Pillo, and G. Koch. Bilinear systems: an appealing class of nearly linear systems. *IEEE Trans. Automat. Control*, 19:334–348, 1974.

[22] C. Ceton, P.M.R. Wortelboer, and O.H. Bosgra. Frequency-weighted closed-loop balanced reduction. In *Proc. of the 2nd European Control Conference, ECC 93, Groningen, The Netherlands*, pages 697–701, 1993.

[23] H. Chen and J. Maciejowski. Subspace identification of deterministic bilinear systems. Technical report, Departement of Engineering, University of Cambridge, UK, 1999. Submitted to ACC2000.

[24] B. Cheng and N-S. Hsu. Analysis and parameter estimation of bilinear systems via block-pulse functions. *Internat. J. Control*, 36(1):53–56, 1982.

[25] J-H. Chou and I-R. Horng. Shifted-Chebyshev-series analysis and identification of time-varying bilinear systems. *Internat. J. Control*, 43(1):129–137, 1986.

[26] D.W. Clarke, C. Mohtadi, and P.S. Tuffs. Generalized Predictive Control: Parts I and II. *Automatica*, 23:137–160, 1987.

[27] C.R. Cutler and B.L. Ramaker. Dynamic Matrix Control - A computer control algorithm. In *AIChE National Mtg, Houston, Texas*, 1979.

[28] B. De Moor. *Mathematical concepts and techniques for modeling of static and dynamic systems*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium, 1988.

[29] B. De Moor. Numerical algorithms for state space subspace system identification. *Academia Analecta, Klasse der Wetenschappen, Koninklijke Akademie voor Wetenschappen, jaargang 55*, 5, 1994.

[30] B. De Moor, P. De Gersem, B. De Schutter, and W. Favoreel. DAISY: A database for identification of systems. *Journal A, Special Issue on CACSD (Computer Aided Control Systems Design)*, 38(3):4–5, Sep. 1997. `http://www.esat.kuleuven.ac.be/sista/daisy`.

[31] B. De Moor and P. Van Overschee. *Numerical Algorithms for Subspace State Space System Identification*, pages 385–422. A. Isidori, Trends in Control. A European Perspective. Springer, Italy, 1995.

[32] B. De Moor, P. Van Overschee, and W. Favoreel. *Numerical algorithms for subspace state space system identification - An overview*, volume 1 of *Applied and Computational Control, Signals and Circuits*, chapter 6. Birkhauser, Boston, 1998.

[33] B. De Moor and J. Vandewalle. A geometrical strategy for the identification of state space models of linear multivariable systems with singular value decomposition. In *Proc. of the 3rd International Symposium on Applications of Multivariable System Techniques*, pages 59–69, Plymouth, UK, April 13-15 1987.

[34] I. Derese and E. Noldus. Design of linear feedback laws for bilinear systems. *Internat. J. Control*, 31(2):219–237, 1980.

[35] U.B. Desai. Realization of bilinear stochastic systems. *IEEE Trans. Automat. Control*, 31(2):189–192, 1986.

[36] M. España and I.D. Landau. Reduced order models of distillation columns. *Automatica*, 14:345–357, 1978.

[37] P. Eykhoff. *System identification*. Wiley, London, 1974.

[38] P. Faurre. Stochastic realization algorithms. In Lainiotis D. (Eds.) Mehra R., editor, *System Identification: Advances and case studies*. Academic Press, 1976.

[39] W. Favoreel, B. De Moor, and M. Gevers. SPC: subspace predictive control. In *Proc. of the 14th IFAC World Congress, Beijing, P.R. China*, pages 235–240, 1999.

[40] W. Favoreel, B. De Moor, M. Gevers, and P. Van Overschee. Model-free subspace-based LQG-design. Technical Report 98-34 (Available by ftp - submitted), Katholieke Universiteit Leuven - ESAT/SISTA, 1998.

[41] W. Favoreel, B. De Moor, M. Gevers, and P. Van Overschee. Closed-loop model-free subspace-based LQG-design. In *Proc. of the 7th IEEE Mediterranean Conference on Control and Automation, June 28-30, Haifa, Israel*, 1999.

[42] W. Favoreel, B. De Moor, M. Gevers, and P. Van Overschee. Model-free subspace-based LQG-design. In *Proc. of the American Control Conference, ACC 99, June 2-4, San Diego, CA, US*, 1999.

[43] W. Favoreel, B. De Moor, S. Van Huffel, V. Sima, and M. Verhaegen. Benchmark for subspace system identification algorithms. Technical Report 98-14, Katholieke Universiteit Leuven - ESAT/SISTA, 1998.

[44] W. Favoreel, B. De Moor, S. Van Huffel, V. Sima, and M. Verhaegen. Comparative study between three subspace identification methods. In *Proc. of the 5th European Control Conference, ECC 99, August 31 - September 3, Karlsruhe, Germany*, 1999.

[45] W. Favoreel, B. De Moor, and P. Van Overschee. A bilinear extension of subspace identification for systems subject to white inputs. In *Proc. of the American Control Conference, ACC 97, June 4-6, Albuquerque, NM, US*, 1997. WM07-3.

[46] W. Favoreel, B. De Moor, and P. Van Overschee. Subspace identification of balanced deterministic bilinear systems subject to white inputs. In *Proc. of the 4th European Control Conference, ECC 97, July 1-4, Brussels, Belgium*, 1997. ECC585.

[47] W. Favoreel, B. De Moor, and P. Van Overschee. Subspace identification of bilinear systems. In *Proc. of the 13th Conference on Mathematical Theory of Networks and Systems, MTNS98, July 6-10, Padova, Italy*, pages 787–790, 1998.

[48] W. Favoreel, B. De Moor, and P. Van Overschee. Subspace state space system identification for industrial processes. Technical report, Katholieke Universiteit Leuven - ESAT/SISTA, 1998. Accepted in Journal of Process Control.

[49] W. Favoreel, B. De Moor, and P. Van Overschee. Subspace state space system identification for industrial processes. In *Proc. of the 5th IFAC*

Symposium on Dynamics and Control of Process Systems, June 8-10, Corfu, Greece, pages 322–330, 1998.

[50] W. Favoreel, B. De Moor, and P. Van Overschee. Subspace identification of bilinear systems subject to white inputs. *IEEE Trans. Automat. Control*, 44(6):1157–1165, 1999.

[51] E. Feng, J. Yu, and W. Jiang. New method for predictive controller design for bilinear systems. *Internat. J. Control*, 53:97–111, 1991.

[52] F. Fnaiech and L. Ljung. Recursive identification of bilinear systems. *Internat. J. Control*, 45(2):453–470, 1987.

[53] U. Forsell and L. Ljung. Closed-loop identification revisited. *Automatica*, 35:1215–1241, 1999.

[54] K. Furuta and M. Wongsaisuwan. Discrete-time LQG dynamic controller design using plant markov parameters. *Automatica*, 31(9):1317–1324, 1995.

[55] K. Furuta, M. Wongsaisuwan, S. Umeki, O. Nishimura, and H. Werner. From noise contaminated pulse responses to design of LQG controller. In *Proc. of the 11th IFAC Symposium on System Identification, SYSID 97, July 8-11, Kitakyushu, Japan*, volume 4, pages 1723–1728, 1997.

[56] M.M. Gabr. A recursive (on-line) identification of bilinear systems. *Internat. J. Control*, 44(4):911–917, 1984.

[57] M.M. Gabr and T. Subba Rao. On the identification of bilinear systems from operating records. *Internat. J. Control*, 40(1):121–128, 1984.

[58] C.E. Garcia and M. Morari. Internal model control: 1. A unifying review and some new results. *Ind. Eng. Chem. Process. Des. Dev.*, 21:308–323, 1982.

[59] M. Gevers. Identification for control. In H.L. Trentelman and J.C. Willems, editors, *Essays on control: perspectives in the theory and its applications*, pages 111–151. 1993.

[60] M. Gevers. Identification for control. In *Proc. of the 5th Symposium on Adaptive Control and Signal Processing*, pages 1–12, 1995.

[61] K. Glover. All optimal Hankel-norm approximations of linear multivariable systems and their l_∞-error bounds. *Internat. J. Control*, 39:1115–1193, 1984.

[62] O.M. Grasselli, A. Isidori, and F. Nicolò. Dead-beat control of discrete-time bilinear systems. *Internat. J. Control*, 32(1):21–39, 1980.

[63] M.J. Grimble. Implicit and explicit LQG self-tuning controllers. *Automatica*, 20(5):661–669, 1984.

[64] T. Gustafsson. System identification using subspace-based instrumental variable methods. In *Proc. of the 11th IFAC Symposium on System Identification, SYSID 97, July 8-11, Kitakyushu, Japan*, volume 3, pages 1119–1124, 1997.

[65] R. Hakvoort. Approximate identification in the controller design problem. Master's thesis, Delft University of Technology, The Netherlands, Measurement and Control Theory Section, Mech. Eng., A-538, 1990.

[66] R. Hakvoort, R. Schrama, and P. Van Den Hof. Approximate identification with closed-loop performance criterion and application to LQG feedback design. *Automatica*, 30(4):679–690, 1994.

[67] B. Haverkamp and M. Verhaegen. *SMI Toolbox: State space Model Identification software for multivariable dynamical systems.* Technical University Delft, 1997. http://lcewww.et.tudelft.nl/ haver/smi.html.

[68] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin. Iterative Feedback Tuning: theory and applications. *IEEE Control Systems Magazine*, 18:26–41, 1998.

[69] H. Hjalmarsson, S. Gunnarsson, and M. Gevers. A convergent iterative restricted complexity control design scheme. In *Proc. of the 33th Conference on Decision and Control, CDC 94, Orlando, Florida, US*, pages 1735–1740, 1994.

[70] B.L. Ho and R.E. Kalman. Efficient construction of linear state space models from input/output functions. *Regelungstechnik*, 14:545–548, 1966.

[71] C. Hwang and M-Y. Chen. Analysis and parameter identification of bilinear systems via shifted Legendre polynomials. *Internat. J. Control*, 44(2):351–362, 1984.

[72] M. Inagaki and H. Mochizuki. Bilinear system identification by Volterra kernels estimation. *IEEE Trans. Automat. Control*, 29(8):746–749, 1984.

[73] A. Isidori. Direct construction of minimal bilinear realizations from nonlinear input-output maps. *IEEE Trans. Automat. Control*, 18(6):626–631, 1973.

[74] A. Isidori and A. Ruberti. *Realization theory of bilinear systems*, pages 83–130. Mayne D.Q. and Brockett R.W., Dordrecht, The Netherlands: Reidel, 1973.

[75] M. Jansson and B. Wahlberg. On consistency of subspace based system identification methods. In *Proc. of the 13th IFAC World Congress, San Francisco, CA, US*, pages 181–186, 1996.

[76] E. Jonckheere and L. Silverman. A new set of invariants for linear systems – Application to reduced order compensator design. *IEEE Trans. Automat. Control*, 28(10):953–964, 1983.

[77] T. Kailath. *Linear Systems.* Prentice Hall, Englewood Cliffs, New Jersey, 1980.

[78] R.E. Kalman. Contributions to the theory of optimal control. *Bull. Soc. Math. Eng. Mex.*, 5:102–119, 1960.

[79] V.R. Karanam, P.A. Frick, and R.R. Mohler. Bilinear system identification by Walsh functions. *IEEE Trans. Automat. Control*, 23(4):709–713, 1978.

[80] Y. Kawamura. Direct synthesis of LQ regulator from inner products of response signals. In *Proc. of the 11th IFAC Symposium on System Identification, SYSID 97, July 8-11, Kitakyushu, Japan*, pages 1717–1722, 1997.

[81] Y. Kawamura. Direct construction of LQ regulator based on orthogonalization of signals: dynamical output feedback. *Systems Control Lett.*, 34:1–9, 1998.

[82] A.Y. Khapalov and R.R. Mohler. Reachable sets and controllability of bilinear time-invariant systems: a qualitative approach. *IEEE Trans. Automat. Control*, 1996.

[83] C.G. Khatri and C.R. Rao. Solutions to some function equations and their applications to characterization of probability distributions. *The Indian J. Stat.*, 30:167–180, 1968.

[84] D.E. Koditschek and K.S. Narendra. The controllability of planar bilinear systems. *IEEE Trans. Automat. Control*, 30(1):87–89, 1985.

[85] S.Y. Kung. A new identification method and model reduction algorithm via singular value decomposition. In *12th Asilomar Conf. on Circuits, Systems and Comp.*, pages 705–714, 1978.

[86] V. Kučera. *Discrete linear control: The polynomial equation approach.* Wiley, New York, 1979.

[87] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems.* Wiley-Interscience, 1972.

[88] W.E. Larimore. System identification, reduced order filtering and modeling via canonical variate analysis. In *Proc. of the American Control Conference, ACC 83, San Francisco, CA, US*, 1983.

[89] W.E. Larimore. Identification of nonlinear systems using canonical variate analysis. In *Proc. of the 26th Conference on Decision and Control, CDC 87, December, Los Angeles, CA, US*, pages 1694–1699, 1987.

[90] W.E. Larimore. Canonical variate analysis in identification, filtering and adaptive control. In *Proc. of the 29th Conference on Decision and Control, CDC 90, Hawaii, US*, pages 596–604, 1990.

[91] W.S. Lee, B.D.O. Anderson, R.L. Kosut, and I.M.Y. Mareels. A new approach to adaptive robust control. *Int. Journal of Adaptive Control and Signal Processing*, 7:183–211, 1993.

[92] L. Ljung. *System identification - Theory for the User*. Prentice Hall, Englewood Cliffs, NJ, 1987.

[93] L. Ljung and T. McKelvey. Subspace identification from closed loop data. *Signal Processing, Special Issue on Subspace Methods, Part II: System Identification*, 52(2):209–216, 1996.

[94] K. Lo, E.S. Yoon, Y.K. Yeo, and H.K. Song. Application of a bilinear long-range predictive control method in a distillation process. In *IFAC Dynamics and Control of Chemical Reactors (DYCORD+'92), Maryland, USA*, pages 249–254, 1992.

[95] R.R. Mohler. *Nonlinear systems: V.2 Applications to bilinear systems.* Englewood Cliffs, NJ: Prentice-Hall, 1991.

[96] M. Moonen, B. De Moor, L. Vandenberghe, and J. Vandewalle. On and off-line identification of linear state space models. *Internat. J. Control*, 49(1):219–232, 1989.

[97] M. Moonen and J. Ramos. A subspace algorithm for balanced state space system identification. *IEEE Trans. Automat. Control*, 38:1727–1729, 1993.

[98] B.C. Moore. Principal component analysis in linear systems: controllability, observability, and model reduction. *IEEE Trans. Automat. Control*, 26(1):17–31, 1981.

[99] D.N.P. Murthy. Controllability of a discrete-time bilinear system. *IEEE Trans. Automat. Control*, 24(6):974–975, 1979.

[100] F. Ohkawa and Y. Yonezawa. A model reference adaptive control system for discrete bilinear systems. *Internat. J. Control*, 37(5):1095–1101, 1983.

[101] B. Ottersten and M. Viberg. A subspace based instrumental variable method for state space system identification. In *Proc. of the 10th IFAC Symposium on System Identification, SYSID 94, July 4-6, Copenhagen, Denmark*, volume 2, pages 139–144, 1994.

[102] V. Peterka. Predictor-based self-tuning control. *Automatica*, 20:39–50, 1984.

[103] K. Peternell, W. Scherrer, and M. Deistler. Statistical analysis of novel subspace identification methods. *Signal Processing*, 52:161–177, 1996.

[104] V. Ranganathan, A.N. Jha, and V.S. Rajamani. Recursive parameter-estimation algorithms for bilinear and non-linear systems using Laguerre-polynomial approach. *Internat. J. Control*, 44(2):419–426, 1986.

[105] J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14:413–428, 1978.

[106] R.J.P. Schrama and P.M.J. Van den Hof. Iterative identification and control design: A three step procedure with robustness analysis. In *Proc. of the 2nd European Control Conference, ECC 93, Groningen, The Netherlands*, pages 237–241, 1993.

[107] D.S. Shook, C. Mothadi, and S.H. Shah. Identification for long-range predictive control. *Automatica*, 1990.

[108] V. Sima. Algorithms and LAPACK-based software for subspace identification. In *Proc. of The 1996 IEEE International Symposium on Computer-Aided Control System Design, September 15–18, Dearborn, MI, US*, pages 182–187, 1996.

[109] V. Sima. High-performance numerical software for control systems analysis and design, and subspace-based system identification. Technical report, Katholieke Universiteit Leuven - ESAT/SISTA, Leuven, Belgium, March 1997. `ftp://wgs.esat.kuleuven.ac.be`, directory `/pub/WGS/REPORTS/rep97-2.ps.Z.`, 1997.

[110] R.E. Skelton and G. Shi. The data-based LQG control problem. In *Proc. of the 33th Conference on Decision and Control, CDC 94, Orlando, Florida, US*, volume 2, pages 1447–1452, 1994.

[111] T. Söderström and P. Stoica. *System Identification*. Prentice Hall International Series in Systems and Control Engineering, Prentice Hall, New York, 1989.

[112] S. Svoronos, G. Stephanopoulos, and R. Aris. Bilinear approximation of general non-linear dynamic systems with linear inputs. *Internat. J. Control*, 31(1):109–126, 1980.

[113] S. Svoronos, G. Stephanopoulos, and R. Aris. On bilinear estimation and control. *Internat. J. Control*, 34(4):651–684, 1981.

[114] K.N. Swamy and T.J. Tarn. *Optimal control of discrete bilinear systems*, pages 275–284. Mayne D.Q. and Brockett R.W., Dordrecht, The Netherlands: Reidel, 1973.

[115] T.J. Tarn, D.L. Elliot, and T. Goka. Controllability of discrete bilinear systems with bounded control. *IEEE Trans. Automat. Control*, pages 298–301, 1973.

[116] P.M.J. Van den Hof and R.J.P. Schrama. Identification and control - closed-loop issues. *Automatica*, 31:1751–1770, 1995.

[117] A. Van Der Veen, E.F. Deprettere, and A.L. Swindlehurst. Subspace-based signal analysis using singular value decompositions. *Proceedings of the IEEE*, 81(9):1277–1308, 1993.

[118] P. Van Overschee and B. De Moor. Subspace algorithms for the stochastic identification problem. *Automatica*, 29(3):649–660, 1993.

[119] P. Van Overschee and B. De Moor. N4SID: subspace algorithms for the identification of combined deterministic and stochastic systems. *Automatica, Special Issue on Statistical Signal Processing and Control*, 30(1):75–93, 1994.

[120] P. Van Overschee and B. De Moor. Choice of state space basis in combined deterministic-stochastic subspace identification. *Automatica*, 31(12):1877–1883, 1995.

[121] P. Van Overschee and B. De Moor. A unifying theorem for three subspace system identification algorithms. *Automatica, Special Issue on Trends in System Identification*, 31(12):1853–1864, 1995.

[122] P. Van Overschee and B. De Moor. Closed loop subspace system identification. Technical report, Katholieke Universiteit Leuven - ESAT/SISTA, 1996. Accepted for publication in Automatica.

[123] P. Van Overschee and B. De Moor. *Subspace identification for linear systems: theory, implementation, applications*. Kluwer Academic Publishers, Dordrecht, 1996.

[124] P. Van Overschee and B. De Moor. Closed loop subspace system identification. In *Proc. of the 36th Conference on Decision and Control, CDC 97, San Diego, California, US*, pages 1848–1853, 1997.

[125] M. Verhaegen. Application of a subspace model identification technique to identify LTI systems operating in closed-loop. *Automatica*, 29(4):1027–1040, 1993.

[126] M. Verhaegen. Identification of the deterministic part of MIMO state space models given in innovations form from input-output data. *Automatica, Special Issue on Statistical Signal Processing and Control*, 30(1):61–74, 1994.

[127] M. Verhaegen and P. Dewilde. Subspace identification, part I: The output-error state space model identification class of algorithms. *Internat. J. Control*, 56:1187–1210, 1992.

[128] E. Verriest. Suboptimal LQG-design via balanced realizations. In *Proc. of the 20th Conference on Decision and Control, CDC 81, December, San Diego, CA, US*, pages 686–687, 1981.

[129] M. Viberg. Subspace methods in system identification. In *Proc. of the 10th IFAC Symposium on System Identification, SYSID 94, July 4-6, Copenhagen, Denmark*, volume 1, pages 1–12, 1994.

[130] M. Viberg. Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, 31(12):1835–1852, 1995.

[131] M. Viberg, B. Ottersten, B. Wahlberg, and L. Ljung. Performance of subspace based state space system identification methods. In *Proc. of the 12th IFAC World Congress, Sydney, Australia*, volume 7, pages 369–372, 1993.

[132] M. Viberg, B. Wahlberg, and B. Ottersten. Analysis of state space system identification methods based on instrumental variables and subspace fitting. *Automatica*, 33(9):1603–1616, 1997.

[133] M-L. Wang, R-Y. Chang, and S-Y. Yang. Analysis and parameter estimation of bilinear systems via generalized orthogonal polynomials. *Internat. J. Control*, 4(2):719–729, 1987.

[134] Y.D. Yeo and D.C. Williams. Bilinear model predictive control. *Ind. Eng. Chem. Res.*, 26:2267–2274, 1987.

[135] Y.K. Yeo and D.C. Williams. An adaptation algorithm for parallel model reference adaptive bilinear systems. *IEEE Trans. Automat. Control*, 31(11):1071–1073, 1986.

[136] H. Zeiger and A. McEwen. Approximate linear realizations of given dimension via Ho's algorithm. *IEEE Trans. Automat. Control*, 19:153, 1974.

Appendix A

Appendix

A.1 Proof of Theorem 3.2.3

Let us first define the estimate error $\widetilde{x}_k^s$, the error covariance matrix $\widetilde{\Sigma}_k^s$ and the covariance matrix Σ_k^s of the state estimate.

$$\begin{aligned}
\widetilde{x}_k^s &\stackrel{\text{def}}{=} x_k^s - \hat{x}_k^s, \\
\widetilde{\Sigma}_k^s &\stackrel{\text{def}}{=} \mathbf{E}[\widetilde{x}_k^s(\widetilde{x}_k^s)^T], \\
\Sigma_k^s &\stackrel{\text{def}}{=} \mathbf{E}[\hat{x}_k^s(\hat{x}_k^s)^T].
\end{aligned}$$

From the filter equation (3.13) and the state-space description (3.6)-(3.8) we find that:

$$\widetilde{x}_{k+1}^s = A\widetilde{x}_k^s + Nu_k \otimes \widetilde{x}_k^s + K_k C\widetilde{x}_k^s + K_k v_k - w_k.$$

The trace of $\widetilde{\Sigma}_{k+1}^s$ then becomes:

$$\begin{aligned}
\text{tr}\,(\widetilde{\Sigma}_{k+1}^s) &= \text{tr}\,\mathbf{E}[\widetilde{x}_{k+1}^s(\widetilde{x}_{k+1}^s)^T] \\
&= \text{tr}\,\mathbf{E}[(A\widetilde{x}_k^s + Nu_k \otimes \widetilde{x}_k^s + K_k C\widetilde{x}_k^s + K_k v_k - w_k) \\
&\qquad (A\widetilde{x}_k^s + Nu_k \otimes \widetilde{x}_k^s + K_k C\widetilde{x}_k^s + K_k v_k - w_k)^T] \\
&= \text{tr}\,\Big\{A\widetilde{\Sigma}_k^s A^T + \sum_{p=1}^{m} N_p \widetilde{\Sigma}_k^s N_p^T + Q - SK_k^T - K_k S^T - K_k C\widetilde{\Sigma}_k^s A^T \\
&\qquad + K_k C\widetilde{\Sigma}_k^s C^T K_k^T + K_k R K_k^T - A\widetilde{\Sigma}_k^s C^T K_k^T\Big\} \\
&= \text{tr}\,\Big\{[K_k(R + C\widetilde{\Sigma}_k^s C^T) - A\widetilde{\Sigma}_k^s C^T - S](R + C\widetilde{\Sigma}_k^s C^T)^{-1} \\
&\qquad [K_k(R + C\widetilde{\Sigma}_k^s C^T) - A\widetilde{\Sigma}_k^s C^T - S]^T\Big\}
\end{aligned}$$

$$
\begin{aligned}
&+\operatorname{tr}\{Q + A\widetilde{\Sigma}^s_k A^T + \sum_{p=1}^{m} N_p \widetilde{\Sigma}^s_k N_p^T \\
&-A\widetilde{\Sigma}^s_k C^T (R + C\widetilde{\Sigma}^s_k C^T)^{-1} C\widetilde{\Sigma}^s_k A^T \\
&\quad -A\widetilde{\Sigma}^s_k C^T (R + C\widetilde{\Sigma}^s_k C^T)^{-1} S^T - S(R + C\widetilde{\Sigma}^s_k C^T)^{-1} C\widetilde{\Sigma^s}_k A^T - \\
&\quad S(R + C\widetilde{\Sigma}^s_k C^T)^{-1} S^T\}.
\end{aligned}
$$

This last equation consists of two terms of which the second one is independent of K_k. Therefore the criterion $\operatorname{tr}(\widetilde{\Sigma}^s_k)$ will be minimized when the first term becomes zero i.e.:

$$
K_k = (A\widetilde{\Sigma}^s_k C^T + S)(R + C\widetilde{\Sigma}^s_k C^T)^{-1}.
$$

Using the definitions of Λ_0, G, Σ^s_k and the fact that $\widetilde{\Sigma}^s_k = \Sigma^s - \Sigma^s_k$ we finally find that the nonsteady-state Kalman filter equations of the bilinear system (3.6)-(3.8):

$$
\begin{aligned}
K_k &= (G - A\Sigma^s_k C^T)(\Lambda_0 - C\Sigma^s_k C^T)^{-1}, \\
\Sigma^s_k &= A\Sigma^s_{k-1} A^T + \sum_{p=1}^{m} N_p \Sigma^s_{k-1} N_p^T + \\
&\quad (G - A\Sigma^s_{k-1} C^T)(\Lambda_0 - C\Sigma^s_{k-1} C^T)^{-1} (G - A\Sigma^s_{k-1} C^T)^T.
\end{aligned}
$$

It follows that the steady-state ($k = \infty$) Kalman filter equations are:

$$
\begin{aligned}
K &= (G^s - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1}, \\
\Sigma^s &= A\Sigma^s A^T + \sum_{p=1}^{m} N_p \Sigma^s N_p^T + \\
&\quad (G - A\Sigma^s C^T)(\Lambda_0 - C\Sigma^s C^T)^{-1} (G - A\Sigma^s C^T)^T.
\end{aligned}
$$

These equations are very similar to the linear Kalman filter equations. The only difference is the additional N-term in the state covariance equation.

A.2 Proof of Theorem 3.4.1

In a first step we prove that the theorem holds for $k = 1$. The proof is completed by showing that, when it holds for $k = p$, it is also true for $k = p + 1$.

$\underline{k = 1}$

With (3.40), (3.12) and (3.41) we have:

$$
\Omega_1 = (\Delta^c_1 - A_1 \Sigma^1_0 \Gamma^T_1)(L_1 - \Gamma_1 \Sigma^1_0 \Gamma^T_1)^{-1}
$$

$$
\begin{aligned}
&= (G - A\Sigma_0^s C^T)(\Lambda_0 - C\Sigma_0^s C^T)^{-1} \\
&= K_0.
\end{aligned}
$$

From (3.35) we have:

$$
\begin{aligned}
\widehat{x}_1 &= A\widehat{x}_0 + Nu_0 \otimes \widehat{x}_0 + Bu_0 + K_0(y_0 - C\widehat{x}_0 - Du_0) \\
&= \left([A \;\; N_1 \;\; \ldots \;\; N_m] \ldots - K_0[C \;\; 0 \;\; \ldots \;\; 0] \;\middle|\; B - K_0 D \;\middle|\; K_0 \right) \\
&\qquad \left(\begin{array}{c} \binom{1}{u_0} \otimes \widehat{x}_0 \\ \hline u_0 \\ \hline y_0 \end{array} \right) \\
&= \left(A_1 - \Omega_1 \Gamma_1^T \;\middle|\; \Delta_1^d - \Omega_1 H_1^d \;\middle|\; \Omega_1 \right) \left(\begin{array}{c} u_{0|0}^P \otimes \widehat{x}_0 \\ \hline u_0 \\ \hline y_0 \end{array} \right)
\end{aligned}
$$

which is equal to (3.38) for $k = 1$.

$$
\begin{aligned}
\Sigma_1^s &= A\Sigma_0^s A^T + \sum_{i=1}^{m} N_i \Sigma_0^s N_i^T + (G - A\Sigma_0^s C^T) \\
&\quad (\Lambda_0 - C\Sigma_0^s C^T)(G - A\Sigma_0^s C^T)^T \\
&= A_1 \Sigma_0^1 \bar{A}_1^T + (\Delta_1^c - A_1 \Sigma_0^1)(L_1 - \Gamma_1 \Sigma_0^1 \Gamma_1^T)^{-1} (\Delta_1^c - A_1 \Sigma_0^1 \Gamma_1^T)^T.
\end{aligned}
$$

$\underline{k = p \;\Rightarrow\; k = p + 1}$

We assume that (3.38) and (3.39) hold for $k = p$ and prove that this is also true for $k = p + 1$. Adopting the following short-hand notations:

$$
\begin{aligned}
\alpha_p &\stackrel{\text{def}}{=} \Delta_p^c - A_p \Sigma_0^p \Gamma_p^T, \\
\beta_p &\stackrel{\text{def}}{=} L_p - \Gamma_p \Sigma_0^p \Gamma_1^T, \\
\gamma_p &\stackrel{\text{def}}{=} G - AA_p \Sigma_0^p A_p^T C^T
\end{aligned}
$$

we have for (3.38) and (3.39):

$$
\widehat{x}_p = \left(A_p - \Omega_p \Gamma_p \;\middle|\; \Delta_p^d - \Omega_p H_p^d \;\middle|\; \Omega_p \right) \left(\begin{array}{c} u_{0|p-1}^P \otimes \widehat{x}_0 \\ \hline u_{0|p-1} \\ \hline y_{0|p-1} \end{array} \right) \tag{A.1}
$$

$$
\Sigma_p^s = A_p \Sigma_0^p A_p + \alpha_p \beta_p^{-1} \alpha_p^T. \tag{A.2}
$$

Before proceeding to the proof of (3.38) and (3.39) for $p+1$ we first prove that:

$$
\Omega_{p+1} = \left(K_p \;\middle|\; (A + K_p C^T)\Omega_p \;\middle|\; N_1 \Omega_p \;\;\ldots\;\; N_m \Omega_p \right).
$$

Since $\Omega_{p+1} = \alpha_{p+1} \beta_{p+1}^{-1}$ we start by analyzing α_{p+1} and β_{p+1}:

$$
\alpha_{p+1} = \Delta_{p+1}^c - A_{p+1} \Sigma_0^{p+1} \Gamma_{p+1}^T
$$

$$
\begin{aligned}
&= \left(\begin{array}{c|c|ccc} G & A\Delta_p^c & N_1\Delta_p^c & \dots & N_m\Delta_p^c \end{array} \right) - \\
&\quad \begin{pmatrix} AA_p & N_1A_p & \dots & N_mA_p \end{pmatrix} \begin{pmatrix} \Sigma_0^p & 0 & \dots & 0 \\ 0 & \Sigma_0^p & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Sigma_0^p \end{pmatrix} \\
&\quad \begin{pmatrix} A_p^TC^T & \Gamma_p^T & 0 & \dots & 0 \\ 0 & 0 & \Gamma_p^T & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \Gamma_p^T \end{pmatrix} \\
&= \Big(G - AA_p\Sigma_0^pA_p^TC^T \;\Big|\; A(\Delta_p^c - A_p\Sigma_0^p\Gamma_p^T) \;\Big| \\
&\quad N_1(\Delta_p^c - A_p\Sigma_0^p\Gamma_p^T) \quad \dots \quad N_m(\Delta_p^c - A_p\Sigma_0^p\Gamma_p^T) \Big) \\
&= \left(\begin{array}{c|c|ccc} \gamma_p & A\alpha_p & N_1\alpha_p & \dots & N_m\alpha_p \end{array} \right).
\end{aligned}
$$

$$
\begin{aligned}
\beta_{p+1} &= L_{p+1} - \Gamma_{p+1}\Sigma_0^{p+1}\Gamma_{p+1}^T \\
&= \left(\begin{array}{c|c|ccc} \Lambda_0 & C\Delta_p^c & 0 & \dots & 0 \\ \hline (\Delta_p^c)^TC^T & L_p & 0 & \dots & 0 \\ \hline 0 & 0 & L_p & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & L_p \end{array} \right) - \begin{pmatrix} CA_p & 0 & \dots & 0 \\ \Gamma_p^T & 0 & \dots & 0 \\ 0 & \Gamma_p^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Gamma_p^T \end{pmatrix} \\
&\quad \begin{pmatrix} \Sigma_0^p & 0 & \dots & 0 \\ 0 & \Sigma_0^p & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Sigma_0^p \end{pmatrix} \begin{pmatrix} A_p^TC^T & \Gamma_p^T & 0 & \dots & 0 \\ 0 & 0 & \Gamma_p^T & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \Gamma_p^T \end{pmatrix} \\
&= \left(\begin{array}{c|c|ccc} \Lambda_0 - CA_p\Sigma_0^pA_p^TC^T & C(\Delta_p^c - A_p\Sigma_0^p\Gamma_p^T) & 0 & \dots & 0 \\ \hline ((\Delta_p^c)^T - \Gamma_p\Sigma_0^pA_p^T)C^T & L_p - \Gamma_p\Sigma_0^p\Gamma_p^T & 0 & \dots & 0 \\ \hline 0 & 0 & L_p - \Gamma_p\Sigma_0^p\Gamma_p^T & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \dots & L_p - \Gamma_p\Sigma_0^p\Gamma_p^T \end{array} \right) \\
&= \left(\begin{array}{c|c|ccc} \Lambda_0 - CA_p\Sigma_0^pA_p^TC^T & C\alpha_p & 0 & \dots & 0 \\ \hline \alpha_p^TC^T & \beta_p & 0 & \dots & 0 \\ \hline 0 & 0 & \beta_p & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \dots & \beta_p \end{array} \right).
\end{aligned}
$$

The inverse of β_{p+1} can be calculated with the matrix inversions lemma of [77]:

$$
\begin{aligned}
\beta_{p+1}^{-1} &= \left(\begin{array}{c|c|ccc}
\Delta^{-1} & -\Delta^{-1}C\alpha_p\beta_p^{-1} & 0 & \dots & 0 \\ \hline
-\beta_p^{-1}\alpha_p^T C^T \Delta^{-1} & \left(\begin{smallmatrix}\beta_p^{-1}+\beta_p^{-1}\alpha_p^T C^T \\ \Delta^{-1}C\alpha_p\beta_p^{-1}\end{smallmatrix}\right) & 0 & \dots & 0 \\ \hline
0 & 0 & \beta_p^{-1} & \dots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \dots & \beta_p^{-1}
\end{array}\right) \\
&= \left(\begin{array}{c|c|ccc}
\Delta^{-1} & -\Delta^{-1}C\Omega_p & 0 & \dots & 0 \\ \hline
-\Omega_p^T \Delta^{-1} & \beta_p^{-1} + \Omega_p^T C^T \Delta^{-1} C \Omega_p^{-1} & 0 & \dots & 0 \\ \hline
0 & 0 & \beta_p^{-1} & \dots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \dots & \beta_p^{-1}
\end{array}\right)
\end{aligned}
$$

where

$$
\begin{aligned}
\Delta^{-1} &= \Lambda_0 - C(A_p\Sigma_0^p A_p^T + \alpha_p\beta_p^{-1}\alpha_p^T)C^T \\
&= \Lambda_0 - C\Sigma_p^s C^T.
\end{aligned}
$$

Now we have that Ω_{p+1} can be written as:

$$
\begin{aligned}
\Omega_{p+1} &= \alpha_{p+1}\beta_{p+1}^{-1} \\
&= \left(\begin{array}{c|c|ccc} \gamma_p & A\alpha_p & N_1\alpha_p & \dots & N_m\alpha_p \end{array}\right) \\
&\quad \left(\begin{array}{c|c|ccc}
\Delta^{-1} & -\Delta^{-1}C\Omega_p & 0 & \dots & 0 \\ \hline
-\Omega_p^T C^T \Delta^{-1} & \beta_p^{-1} + \Omega_p^T C^T \Delta^{-1} C \Omega_p^{-1} & 0 & \dots & 0 \\ \hline
0 & 0 & \beta_p^{-1} & \dots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \dots & \beta_p^{-1}
\end{array}\right) \\
&= \Big(\; (\gamma_p - A\Omega_p\alpha_p^T)\Delta^{-1} \; \Big| \; A\Omega_p + A\Omega_p\alpha_p^T C^T \Delta^{-1} C\Omega_p - \gamma_p\Delta^{-1}C\Omega_p \; \Big| \\
&\qquad N_1\Omega_p \quad \dots \quad N_m\Omega_p \; \Big) \\
&= \Big(\; (G - A\Sigma_p^s C^T)\Delta^{-1} \; \Big| \; (A + (G - A\Sigma_p^s C^T)\Delta^{-1}C)\Omega_p \; \Big| \\
&\qquad N_1\Omega_p \quad \dots \quad N_m\Omega_p \; \Big) \\
&= \Big(\; K_p \; \Big| \; (A - K_pC)\Omega_p \; \Big| \; N_1\Omega_p \quad \dots \quad N_m\Omega_p \; \Big).
\end{aligned}
$$

Finally the expressions for Σ_{p+1}^s and $\widehat{x}_{p+1}$ become:

$$
\begin{aligned}
\Sigma_{p+1}^s &= A_{p+1}\Sigma_0^{p+1}A_{p+1}^T + \Omega_{p+1}\alpha_{p+1}^T \\
&= AA_p\Sigma_0^p A_p^T A^T + \sum_{i=1}^m N_i A_p \Sigma_0^p A_p^T N_i^T + \\
&\quad \Big(\; K_p \; \Big| \; (A - K_pC)\Omega_p \; \Big| \; N_1\Omega_p \quad \dots \quad N_m\Omega_p \; \Big)
\end{aligned}
$$

$$
\begin{aligned}
& \left(\begin{array}{c|c|ccc} \gamma_p & A\alpha_p & N_1\alpha_p & \dots & N_m\alpha_p \end{array}\right)^T \\
= \; & A(A_p\Sigma_0^p A_p^T + \Omega_p\alpha_p^T)A^T + \sum_{i=1}^{m} N_i(A_p\Sigma_0^p A_p^T + \Omega_p\alpha_p^T)N_i^T \\
& K_p\left(G^T - C(A_p\Sigma_0^p A_p^T + \Omega_p\alpha_p^T)A^T\right) \\
= \; & A\Sigma_p^s A^T + \sum_{i=1}^{m} N_i\Sigma_p^s N_i^T + \\
& (G - A\Sigma_p^s C^T)(\Lambda_0 - C\Sigma_p^s C^T)^{-1}(G - A\Sigma_p^s C^T)^T.
\end{aligned}
$$

$$
\begin{aligned}
\widehat{x}_{p+1} \; = \; & (A_{p+1} - \Omega_{p+1}\Gamma_{p+1})u_{0|p}^P \otimes \widehat{x}_0 + (\Delta_{p+1}^d - \Omega_{p+1}H_{p+1}^d)u_{0|p} + \Omega_p y_{0|p} \\
= \; & \Big[\left(\begin{array}{cccc} AA_p & N_1A_p & \dots & N_mA_p \end{array}\right) - \\
& \left(\begin{array}{c|c|ccc} K_p & (A-K_pC)\Omega_p & N_1\Omega_p & \dots & N_m\Omega_p \end{array}\right) \\
& \left(\begin{array}{cccc} CA_p & 0 & \dots & 0 \\ \Gamma_p & 0 & \dots & 0 \\ 0 & \Gamma_p & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Gamma_p \end{array}\right)\Big] \left(\begin{array}{c} u_{0|p-1}^P \\ u_{p,1}u_{0|p-1}^P \\ \vdots \\ u_{p,m}u_{0|p-1}^P \end{array}\right) \otimes \widehat{x}_0 \\
& +\Big[\left(\begin{array}{c|c|ccc} G & A\Delta_p^d & N_1\Delta_p^d & \dots & N_m\Delta_p^d \end{array}\right) \\
& -\left(\begin{array}{c|c|ccc} K_p & (A-K_pC)\Omega_p & N_1\Omega_p & \dots & N_m\Omega_p \end{array}\right) \\
& \left(\begin{array}{c|c|ccc} D & C\Delta_p^d & 0 & \dots & 0 \\ \hline 0 & H_p^d & 0 & \dots & 0 \\ \hline 0 & 0 & H_p^d & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & H_p^d \end{array}\right)\Big] \left(\begin{array}{c} u_p \\ u_{0|p-1} \\ u_{p,1}u_{0|p-1} \\ \vdots \\ u_{p,m}u_{0|p-1} \end{array}\right) \\
& +\left(\begin{array}{c|c|ccc} K_p & (A-K_pC)\Omega_p & N_1\Omega_p & \dots & N_m\Omega_p \end{array}\right) \left(\begin{array}{c} y_p \\ y_{0|p-1} \\ u_{p,1}y_{0|p-1} \\ \vdots \\ u_{p,m}y_{0|p-1} \end{array}\right) \\
= \; & (A - K_pC)(A_p - \Omega_p\Gamma_p)u_{0|p-1}^P \otimes \widehat{x}_0 \\
& (A - K_pC)(\Delta_p^d - \Omega_pH_p^d)u_{0|p-1}^P + (A - K_pC)\Omega_p y_{0|p-1} \\
& +K_py_p + (B - K_pD)u_p \\
& \sum_{i=1}^{m} N_i u_{p,i}(A_p - \Omega_p\Gamma_p)u_{0|p-1}^P \otimes \widehat{x}_0
\end{aligned}
$$

$$
\begin{aligned}
& + \sum_{i=1}^{m} N_i u_{p,i} \left((\Delta_p^d - \Omega_p H_p^d) u_{0|p-1} + \Omega_p y_{0|p-1} \right) \\
= \; & A\widehat{x}_p + N u_p \otimes \widehat{x}_p + B u_p + K_p (y_p - C\widehat{x}_p - D u_p).
\end{aligned}
$$

A.3 Proof of Theorem 3.4.2

Following the definition orthogonal projections 2.3.1:

$$
\begin{aligned}
\mathcal{O}_i &= Y_f^l / \begin{pmatrix} Y_p \\ U_p \end{pmatrix} \\
&= \mathbf{E}_j [Y_f^l \begin{pmatrix} Y_p \\ U_p \end{pmatrix}^T] \mathbf{E}_j^{-1} [\begin{pmatrix} Y_p \\ U_p \end{pmatrix} \begin{pmatrix} Y_p \\ U_p \end{pmatrix}^T] \begin{pmatrix} Y_p \\ U_p \end{pmatrix}.
\end{aligned}
$$

Using the matrix input/output equations (3.29)-(3.34) and the fact that the inputs u_k are white and independent of the noise e_k the different covariances in this projection can be written as:

$$
\begin{aligned}
\mathbf{E}_j[U_p U_p^T] &= I, \\
\mathbf{E}_j[Y_p U_p^T] &= \mathbf{E}_j[(\Gamma_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + H_i^d U_p + H_i^s E_p) U_p^T] \\
&= H_i^d, \\
\mathbf{E}_j[Y_p Y_p^T] &= \mathbf{E}_j[(\Gamma_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + H_i^d U_p + H_i^s E_p) \\
& \quad (\Gamma_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + H_i^d U_p + H_i^s E_p)^T] \\
&= \Gamma_i \Sigma^{d,i} \Gamma_i^T + H_i^d (H_i^d)^T + L_i \\
\mathbf{E}_j[Y_f^l U_p^T] &= \mathbf{E}_j[(\Gamma_i^l X_f + \Gamma_i^b \bar{U}_f^P \odot X_f + H_i^{d,b} U_f + H_i^{s,b} E_f) U_p^T] \\
&= \mathbf{E}_j[(\Gamma_i^l A_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p \end{pmatrix} \odot X_p + \Gamma_i^l \Delta_i^d U_p + \Gamma_i^l \Delta_i^s E_p) U_p^T] \\
&= \Gamma_i^l \Delta_i^d \\
\mathbf{E}_j[Y_f^l Y_p^T] &= \mathbf{E}_j[(\Gamma_i^l X_f + \Gamma_i^b \bar{U}_f^P \odot X_f + H_i^{d,b} U_f + H_i^{s,b} E_f) \\
& \quad (\Gamma_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + H_i^d U_p + H_i^s E_p)^T] \\
&= \mathbf{E}_j[(\Gamma_i^l A_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + \Gamma_i^l \Delta_i^d U_p + \Gamma_i^l \Delta_i^s E_p) \\
& \quad (\Gamma_i \begin{pmatrix} I_{1\times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + H_i^d U_p + H_i^s E_p)^T]
\end{aligned}
$$

$$= \quad \Gamma_i^l (A_i \Sigma^{d,i} \Gamma_i^T + \Delta_i^d (H_i^d)^T + \Delta_i^c)$$

With the matrix inversion theorem of [77] and with $\Delta = L_i + \Gamma_i \Sigma^{d,i} \Gamma_i^T$ we have:

$$\begin{aligned}
\mathcal{O}_i &= \Gamma_i^l \begin{pmatrix} A_i \Sigma^{d,i} \Gamma_i^T + \Delta_i^d (H_i^d)^T + \Delta_i^c & \Delta_i^d \end{pmatrix} \\
&\quad \begin{pmatrix} L_i + \Gamma \Sigma^{d,i} \Gamma_i^T + H_i^d (H_i^d)^T & H_i^d \\ (H_i^d)^T & I \end{pmatrix}^{-1} \begin{pmatrix} Y_{0|i-1} \\ U_{0|i-1} \end{pmatrix} \\
&= \Gamma_i^l \begin{pmatrix} \Delta_i^c + A_i \Sigma^{d,i} \Gamma_i^T + \Delta_i^d (H_i^d)^T & \Delta_i^d \end{pmatrix} \\
&\quad \begin{pmatrix} \Delta^{-1} & -\Delta^{-1} H_i^d \\ -(H_i^d)^T \Delta^{-1} & I + (H_i^d)^T \Delta^{-1} (H_i^d)^T \end{pmatrix} \begin{pmatrix} Y_{0|i-1} \\ U_{0|i-1} \end{pmatrix} \\
&= \Gamma_i^l \Big\{ \Delta_i^d U_{0|i-1} + (A_i \Sigma^{d,i} \Gamma_i^T + \Delta_i^c) \\
&\quad (L_i + \Gamma_i \Sigma^{d,i} \Gamma_i^T)^{-1} (Y_{0|i-1} - H_i^d U_{0|i-1}) \Big\} \\
&= \Gamma \widehat{X}_i
\end{aligned}$$

where $\widehat{X}_i$ is the nonsteady-state Kalman filter state sequence as defined in Theorem 3.4.1 with $\widehat{X}_0 = 0$ and $\Sigma_0^s = -\Sigma^d$. This proves the first part of the theorem while the second and third part directly follow from the properties of the singular value decomposition.

A.4 Proof of Theorem 4.2.2

From the matrix input/output equations (3.29) and (3.31) it can be seen that, under the assumptions of Theorem 4.2.2, the state sequence X_f is contained in the row space of W_p. By substituting E_p of (3.29) into X_f of (3.31) and under the condition that the last two assumptions are satisfied, we get the following expression for the state sequence:

$$\begin{aligned}
X_f &= (A_i - \Delta_i^s H_i^{-s} \Gamma_i) \begin{pmatrix} I_{1 \times j} \\ \bar{U}_p^P \end{pmatrix} \odot X_p + \\
&\quad \begin{pmatrix} \Delta_i^s H_i^{-s} \mid \Delta_i^d - \Delta_i^d H_i^{-s} H_i^d \end{pmatrix} \begin{pmatrix} Y_p \\ U_p \end{pmatrix}.
\end{aligned}$$

The first term of the right hand side is zero since we assumed that $i = \infty$ and the matrix $A + N u_k \otimes x_k - KC$ is stable for all u_k.

Using the matrix input/output equation (3.33):

$$Y_f^l \quad = \quad \Gamma_i^l X_f + \Gamma_i^b \bar{U}_f^P \odot X_f + H_i^{d,b} U_f + H_i^{s,b} E_f,$$

the oblique projection (4.5) can be written as:

$$\begin{aligned} Y_f^l /_{\binom{U_f}{U_f \odot W_p}} W_p &= \Gamma_i^l X_f /_{\binom{U_f}{U_f \odot W_p}} W_p + \Gamma_i^b \underbrace{(\bar{U}_f^P \odot X_f) /_{\binom{U_f}{U_f \odot W_p}} W_p}_{=0} \\ &\quad + H_i^{d,b} \underbrace{U_f /_{\binom{U_f}{U_f \odot W_p}} W_p}_{=0} + H_i^{s,b} \underbrace{E_f /_{\binom{U_f}{U_f \odot W_p}} W_p}_{=0} \\ &= \Gamma_i^l \widehat{X}_f . \end{aligned}$$

Since X_f is spanned by the row space of W_p, the second projection of the right hand side is zero. The last two projections are always zero. The rank of X_f should also be maintained after projection:

$$\operatorname{rank} X_f \quad = \quad \operatorname{rank} X_f /_{\binom{U_f}{U_f \odot W_p}} W_p$$

which will be satisfied if the input is persistently exciting (Definition 4.2.1).

A.5 Proof of Theorem 6.5.1

Riccati equation (6.10)

First we prove that

$$P_N \quad = \quad \Gamma_N^T \left(\bar{Q} - \bar{Q} H_N (\bar{R} + H_N^T \bar{Q} H_N)^{-1} H_N^T \bar{Q} \right) \Gamma_N \qquad \text{(A.3)}$$

equals the LQR Riccati equation (6.10):

$$\begin{aligned} P_N &= A^T P_{N-1} A + C^T Q_1 C - (C^T Q_1 D + A^T P_{N-1} B) \\ &\quad (R_1 + D^T Q_1 D + B^T P_{N-1} B)^{-1} (C^T Q_1 D + A^T P_{N-1} B)^T . \end{aligned}$$

The proof is one by induction: first we prove that (6.10) equals (A.3) for $N = 1$. In as second step we assume that they are equal for $N - 1$, and show that this is also true for N, which then completes the proof.

$\underline{N=1}$:

For $N = 1$ (A.3) is:

$$\begin{aligned} P_1 &= \Gamma_1^T \left(Q_N - Q_N H_1^d (R_N + (H_1^d)^T Q_N H_1^d)^{-1} (H_1^d)^T Q_N \right) \Gamma_1 \\ &= C^T Q_N C - C^T Q_N D (R_N + D^T Q_N D)^{-1} D^T Q_N C \end{aligned}$$

which corresponds to (6.10) with $N = 1$ and $P_0 = 0$.

$\underline{N-1 \to N}$:

Now we assume that (6.10) equals (A.3) for $N-1$

$$\begin{aligned} P_{N-1} &= \Gamma_{N-1}^T \left(Q_{2:N} - Q_{2:N} H_{N-1}^d (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \right. \\ & \qquad \left. (H_{N-1}^d)^T Q_{2:N}\right) \Gamma_{N-1}, \end{aligned} \tag{A.4}$$

and we prove that this also holds for N. Before proceeding to this proof, we first use the matrix inversion theorem to calculate the inverse of $\bar{R} + (H_N^d)^T \bar{Q} H_N^d$. With the Schur complement Δ:

$$\begin{aligned} \Delta &\stackrel{\text{def}}{=} R_1 + D^T Q_1 D + B^T \Gamma_{N-1}^T Q_{2:N} \Gamma_{N-1} B - B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \cdot \\ & \qquad (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} (H_{N-1}^d)^T Q_{2:N} \Gamma_{N-1} B \\ &= R_1 + D^T Q_1 D + B^T P_{N-1} B, \end{aligned}$$

we have that:

$$(\bar{R} + (H_N^d)^T \bar{Q} H_N^d)^{-1} =$$

$$\begin{pmatrix} \Delta^{-1} & \begin{matrix} -\Delta^{-1} B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \cdot \\ (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \end{matrix} \\ \begin{matrix} -(R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \cdot \\ (H_{N-1}^d)^T Q_{2:N} \Gamma_{N-1} B \Delta^{-1} \end{matrix} & \begin{matrix} (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \\ \big(I + (H_{N-1}^d)^T Q_{2:N} \Gamma_{N-1} B \Delta^{-1} \cdot \\ B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \\ (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1}\big) \end{matrix} \end{pmatrix} \tag{A.5}$$

We know that with:

$$\Delta = R_1 + D^T Q_1 D + B^T P_{N-1} B,$$

P_N becomes:

$$\begin{aligned} P_N &= \Gamma_N^T \left(\bar{Q} - \bar{Q} H_N (\bar{R} + H_N^T \bar{Q} H_N)^{-1} H_N^T \bar{Q}\right) \Gamma_N \\ &= \begin{pmatrix} C \\ \Gamma_{N-1} A \end{pmatrix}^T \left[\begin{pmatrix} Q_1 & 0 \\ 0 & Q_{2:N} \end{pmatrix} - \begin{pmatrix} Q_1 & 0 \\ 0 & Q_{2:N} \end{pmatrix} \begin{pmatrix} D & 0 \\ \Gamma_{N-1} B & H_{N-1}^d \end{pmatrix} \cdot \right. \\ & \quad \begin{pmatrix} \Delta^{-1} & \begin{matrix} -\Delta^{-1} B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \cdot \\ (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \end{matrix} \\ \begin{matrix} -(R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \cdot \\ (H_{N-1}^d)^T Q_{2:N} \Gamma_{N-1} B \Delta^{-1} \end{matrix} & \begin{matrix} (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \\ \big(I + (H_{N-1}^d)^T Q_{2:N} \Gamma_{N-1} B \Delta^{-1} \\ B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \\ (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1}\big) \end{matrix} \end{pmatrix} \cdot \\ & \quad \left. \begin{pmatrix} D & 0 \\ \Gamma_{N-1} B & H_{N-1}^d \end{pmatrix}^T \begin{pmatrix} Q_1 & 0 \\ 0 & Q_{2:N} \end{pmatrix} \right] \begin{pmatrix} C \\ \Gamma_{N-1} A \end{pmatrix} \\ &= -C^T Q_1 D \Delta^{-1} D^T Q_1 C^T + C^T Q_1 C \\ & \quad -A^T \Gamma_{N-1}^T Q_{2:N} \Gamma_{N-1} B \Delta^{-1} D^T Q_1 C \end{aligned}$$

$$
\begin{aligned}
&+A^T\Gamma_{N-1}^T Q_{2:N} H_{N-1}^d (R_{2:N}+H_{N-1}^d Q_{2:N} H_{N-1}^d)^{-1} H_{N-1}^d Q_{2:N}\Gamma_{N-1} B\Delta^{-1} D^T Q_1 C \\
&+C^T Q_1 D\Delta^{-1} B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d (R_{2:N}+H_{N-1}^d Q_{2:N} H_{N-1}^d)^{-1} B^T\Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} A \\
&-C^T Q_1 D\Delta^{-1} B^T \Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} A \\
&-A^T\Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} B\Delta^{-1} B^T\Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} A \\
&+A^T\Gamma_{N-1}^T Q_{2:N} H_{N-1}^d (R_{2:N}+H_{N-1}^d Q_{2:N} \\
&\qquad H_{N-1}^d)^{-1} (H_{N-1}^d)^T Q_{2:N}\Gamma_{N-1} B\Delta^{-1} B^T\Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} A \\
&+A^T\Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} B\Delta^{-1} B^T\Gamma_{N-1}^T Q_{2:N} \\
&\qquad H_{N-1}^d (R_{2:N}+H_{N-1}^d Q_{2:N} H_{N-1}^d)^{-1} (H_{N-1}^d)^T Q_{2:N}\Gamma_{N-1} A \\
&-A^T\Gamma_{N-1}^T Q_{2:N} H_{N-1}^d (R_{2:N}+H_{N-1}^d Q_{2:N} H_{N-1}^d)^{-1}. \\
&\qquad \big[I+(H_{N-1}^d)^T Q_{2:N}\Gamma_{N-1} B\Delta^{-1} B^T\Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \\
&\qquad (R_{2:N}+H_{N-1}^d Q_{2:N} H_{N-1}^d)^{-1}\big] (H_{N-1}^d)^T Q_{2:N}\Gamma_{N-1} A \\
&+A^T\Gamma_{N-1}^T Q_{2:N}\Gamma_{N-1} A.
\end{aligned}
$$

Since we assumed that (A.4) holds, we find the final expression for P_N:

$$
\begin{aligned}
P_N &= A^T P_{N-1} A + C^T Q_1 C - (C^T Q_1 D + A^T P_{N-1} B). \\
&\quad (R_1 + D^T Q_1 D + B^T P_{N-1} B)^{-1} (C^T Q_1 D + A^T P_{N-1} B)^T.
\end{aligned}
$$

Feedback gain equation (6.9)

We show that the first element u_1 of

$$
u_f = -(\bar{R} + (H_N^d)^T \bar{Q} H_N^d)^{-1} (H_N^d)^T \bar{Q} \Gamma_N \widehat{x}_1
$$

corresponds to the solution of the LQR equations (6.8)-(6.10).

Let us focus on the term $\bar{R} + (H_N^d)^T \bar{Q} H_N^d$. Since $\bar{R}$ is a block diagonal matrix, it can be split up into the first block element R_1 and the remaining part $R_{2:N}$. The same holds for $\bar{Q}$. Exploiting the structure of H_N^d we can write that:

$$
\begin{aligned}
\bar{R} + (H_N^d)^T \bar{Q} H_N^d &= \begin{pmatrix} R_1 & 0 \\ 0 & R_{2:N} \end{pmatrix} + \begin{pmatrix} D & 0 \\ \Gamma_{N-1} B & H_{N-1}^d \end{pmatrix}^T \begin{pmatrix} Q_1 & 0 \\ 0 & Q_{2:N} \end{pmatrix}. \\
&\qquad \begin{pmatrix} D & 0 \\ \Gamma_{N-1} B & H_{N-1}^d \end{pmatrix} \\
&= \begin{pmatrix} R_1 + D^T Q_1 D + B^T \Gamma_{N-1}^T Q_{2:N} \Gamma_{N-1} B & B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d \\ (H_{N-1}^d)^T Q_{2:N} \Gamma_{N-1} B & R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d \end{pmatrix}.
\end{aligned}
$$

We now see that the first element u_1 of u_f can be written as:

$$
\begin{aligned}
u_1 &= -\begin{pmatrix} \Delta^{-1} & -\Delta^{-1} B^T \Gamma_{N-1}^T Q_{2:N} H_{N-1}^d (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} \end{pmatrix}. \\
&\quad \begin{pmatrix} D & 0 \\ \Gamma_{N-1} B & H_{N-1}^d \end{pmatrix}^T \begin{pmatrix} Q_1 & 0 \\ 0 & Q_{2:N} \end{pmatrix} \begin{pmatrix} C \\ \Gamma_{N-1} B \end{pmatrix} \widehat{x}_1
\end{aligned}
$$

$$
\begin{aligned}
&= -\Delta^{-1}\left(D^T Q_1 C + B^T \Gamma_{N-1}^T Q_{2:N} \Gamma_{N-1} B - B^T \Gamma_{N-1}^T Q_{2:N}\right. \\
&\qquad \left. H_{N-1}^d (R_{2:N} + (H_{N-1}^d)^T Q_{2:N} H_{N-1}^d)^{-1} H_{N-1}^d Q_{2:N} \Gamma_{N-1} B\right) \widehat{x}_1 .
\end{aligned}
$$

Since we proved that P_N can be written as:

$$
P_N = \Gamma_N^T \left(\bar{Q} - \bar{Q} H_N (R + H_N^T \bar{Q} H_N)^{-1} H_N^T \bar{Q}\right) \Gamma_N
$$

we then have that Δ becomes:

$$
\Delta = R_1 + D^T Q_1 D + B^T P_{N-1} B.
$$

Finally for the control input u_1 we have

$$
\begin{aligned}
u_1 &= -\left(R_1 + D^T Q_1 D + B^T P_{N-1} B\right)^{-1} \left(D^T Q_1 C + B^T P_{N-1} A\right) \widehat{x}_1 \\
&= L_1 \widehat{x}_1 .
\end{aligned}
$$

This proves the feedback gain equation of the LQ-controller.

A.6 Proof of Theorem 8.3.1

From the subspace identification algorithm (See Section 2.4) it is known that the weighed singular value decomposition of L_w:

$$
W_1 L_w W_2 = U_1 S_1 V_1^T
$$

can be used to find an estimate of the extended observability matrix Γ_i and the Kalman filter state sequence $\widehat{X}_f$:

$$
\begin{aligned}
\Gamma_i &= W_1^{-1} U_1^T S_1^{1/2}, \\
\widehat{X}_f &= S_1^{1/2} V_1^T W_2^{-1} W_p .
\end{aligned}
$$

This Kalman filter state sequence can also be expressed as in (2.29):

$$
\widehat{X}_f = A^i X_p + \Delta_i^d U_p + \Delta_i^s E_p
$$

where Δ_i^d and Δ_i^s are the reversed extended controllability matrices (2.15) and (2.16) of the deterministic and the stochastic part of the system in innovation form (2.9)-(2.10). Substituting E_p from (2.27) in the above expression of $\widehat{X}_f$ we have:

$$
\widehat{X}_f = (A^i - \Delta_i^s H_i^{-s} \Gamma_i) \widehat{X}_p + \Delta_i^s H_i^{-s} Y_p + (\Delta_i^d - \Delta_i^s H_i^{-s} H_i) U_p
$$

where the matrix $A^i - \Delta_i^s H_i^{-s} \Gamma_i$ is in fact nothing else than $(A - KC)^i$ with $A - KC$ the dynamic matrix of the Kalman filter, which is always stable in the case of linear time-invariant systems. Since we assumed that the number of block rows in the data block-Hankel matrices $i = \infty$, the steady-state Kalman filter state sequence then becomes (dropping the index $i \to \infty$ everywhere):

$$\widehat{X}_f = \begin{pmatrix} \Delta^d & \Delta^s \end{pmatrix} \begin{pmatrix} 0 & I \\ H^{-s} & -H^{-s}H^d \end{pmatrix} \begin{pmatrix} Y_p \\ U_p \end{pmatrix} .$$

The observability and controllability Grammians $\mathcal{O}$ and $\mathcal{C}$ of the plant, solution of the following Lyapunov equations:

$$\begin{aligned} \mathcal{O} &= A^T \mathcal{O} A + C^T C, \\ \mathcal{C} &= A\mathcal{C}A^T + BB^T + KK^T, \end{aligned}$$

can also be written as:

$$\begin{aligned} \mathcal{O} &= \Gamma^T \Gamma, \\ \mathcal{C} &= \Delta^d (\Delta^d)^T + \Delta^s (\Delta^s)^T . \end{aligned}$$

We will now show that, if the weighting matrices W_1 and W_2 are chosen as in Theorem 8.3.1, the observability and the controllability Grammians are diagonal and equal to $\mathcal{O} = \mathcal{C} = S_1$, which indicates that the model is balanced.

Observability Grammian: With $W_1 = I$ we directly have for the observability Grammian $\mathcal{O}$:

$$\begin{aligned} \mathcal{O} &= \Gamma^T \Gamma \\ &= S_1^{1/2} U_1^T U_1 S_1^{1/2} \\ &= S_1 . \end{aligned}$$

Controllability Grammian: Comparing the expression of the Kalman filter state sequence $\widehat{X}_f$ estimated with the singular value decomposition:

$$\widehat{X}_f = S_1^{1/2} V_1^T W_2^{-1} W_p$$

and the expression of $\widehat{X}_f$ derived above:

$$\widehat{X}_f = \begin{pmatrix} \Delta^d & \Delta^s \end{pmatrix} \begin{pmatrix} 0 & I \\ H^{-s} & -H^{-s}H^d \end{pmatrix} W_p ,$$

we directly conclude that:

$$\begin{pmatrix} \Delta^d & \Delta^s \end{pmatrix} = S_1^{1/2} V_1^T W_2^{-1} \begin{pmatrix} 0 & I \\ H^{-s} & -H^{-s}H^d \end{pmatrix}^{-1} .$$

Finally, if W_2 is chosen equal to:

$$W_2 = \begin{pmatrix} H^d & H^s \\ I & 0 \end{pmatrix}$$

the controllability Grammian $\mathcal{C}$ becomes:

$$\begin{aligned}\mathcal{C} &= \Delta^d(\Delta^d)^T + \Delta^s(\Delta^s)^T \\ &= S_1^{1/2}V_1^T V_1 S_1^{1/2} \\ &= S_1.\end{aligned}$$

As a result, if W_1 and W_2 are chosen as in Theorem 8.4.1, the observability and controllability Grammians of the plant are diagonal and equal to S_1:

$$\mathcal{O} = \mathcal{C} = S_1$$

which proves that the model is balanced.

A.7 Proof of Theorem 8.4.1

The proof of this theorem is very similar to the one of Theorem 8.3.1. The observability and controllability Grammians of the controller (8.2)-(8.3), $\mathcal{O}_c$ and $\mathcal{C}_c$ respectively, are the solution of the following Lyapunov equations:

$$\begin{aligned}\mathcal{O}_c &= A_c^T \mathcal{O}_c A_c + C_c^T C_c, \\ \mathcal{C}_c &= A_c \mathcal{C}_c A_c^T + B_c^y (B_c^y)^T + B_c^r (B_c^r)^T.\end{aligned}$$

These equations can also be written as:

$$\begin{aligned}\mathcal{O}_c &= \Gamma_c^T \Gamma_c, \\ \mathcal{C}_c &= \Delta_c^y (\Delta_c^y)^T + \Delta_c^r (\Delta_c^r)^T\end{aligned}$$

where Γ_c is the extended observability matrix of the controller and Δ_c^y, Δ_c^r the reversed extended controllability matrices corresponding the inputs of the controller y_k and r_k. They have been defined previously in (7.11), (7.14) and (7.15). We will now show how these Grammians can be expressed as a function of the left and right singular vectors of the singular value decomposition (See Figure 8.5):

$$W_1 W_c L_w W_2 = U_1 S_1 V_1^T$$

where:

$$\begin{aligned}\Gamma_c &= W_1^{-1} U_1 S_1^{1/2}, \\ \widehat{X}_f &= S_1^{1/2} V_1^T W_2^{-1} W_p.\end{aligned}$$

Observability Grammian: With $W_1 = I$ we directly have for the observability Grammian $\mathcal{O}_c$:

$$\mathcal{O}_c = \Gamma_c^T \Gamma_c$$

$$\begin{aligned} &= S_1^{1/2} U_1^T U_1 S_1^{1/2} \\ &= S_1. \end{aligned}$$

Controllability Grammian: In the proof of Theorem 8.3.1 (See Appendix A.6) we have seen that the Kalman filter state sequence $\widehat{X}_f$ can be written as:

$$\begin{aligned} \widehat{X}_f &= (A^M - \Delta^s H^{-s}\Gamma)\widehat{X}_p + \\ &\quad \Delta^s H^{-s} Y_p + (\Delta^d - \Delta^s H^{-s} H^d) U_p. \end{aligned} \tag{A.6}$$

In a similar way as for the plant, one can also derive input/output equations (7.7)-(7.9) for the controller:

$$\begin{aligned} U_p &= \Gamma_c X_p^c + H_c^y Y_p + H_c^r R_p, \\ X_f^c &= A_c^M X_p^c + \Delta_c^y Y_p + \Delta_c^r R_p. \end{aligned}$$

When substituting R_p from the first equation in the second one we have:

$$\begin{aligned} X_f^c &= (A_c^M - \Delta_c^r H_c^{-r}\Gamma_c) X_p^c + \\ &\quad (\Delta_c^y - \Delta_c^r H_c^{-r} H_c^y) Y_p + \Delta_c^r H_c^{-r} U_p. \end{aligned} \tag{A.7}$$

Since the controller states are in fact the Kalman filter estimates of the plant we have that $X_f^c = \widehat{X}_f$. Comparing (A.6) with (A.7), and keeping in mind that $M = \infty$, it becomes clear that:

$$\begin{aligned} A_c^M - \Delta_c^r H_c^{-r}\Gamma_c &= A^M - \Delta^s H^{-s}\Gamma \\ &= (A - KC)^M \\ &\overset{M\to\infty}{=} 0. \end{aligned}$$

Finally the plant Kalman filter state sequence $\widehat{X}_f$ is:

$$\widehat{X}_f = \begin{pmatrix} \Delta_c^y & \Delta_c^r \end{pmatrix} \begin{pmatrix} I & 0 \\ -H_c^{-r} H_c^y & H_c^{-r} \end{pmatrix} W_p.$$

Comparing this with the expression of the state sequence $\widehat{X}_f$ estimated with the singular value decomposition:

$$\widehat{X}_f = S_1^{1/2} V_1^T W_2^{-1} W_p$$

we directly conclude that:

$$\begin{pmatrix} \Delta_c^y & \Delta_c^r \end{pmatrix} = S_1^{1/2} V_1^T W_2^{-1} \begin{pmatrix} I & 0 \\ -H_c^{-r} H_c^y & H_c^{-r} \end{pmatrix}^{-1}.$$

Finally, if W_2 is chosen equal to:

$$W_2 = \begin{pmatrix} I & 0 \\ H_c^y & H_c^r \end{pmatrix}$$

we have that:

$$\begin{pmatrix} \Delta_c^y & \Delta_c^r \end{pmatrix} = S_1^{1/2} V_1^T.$$

In that case, the controllability Grammian $\mathcal{C}_c$ of the controller becomes:

$$\begin{aligned} \mathcal{C}_c &= \Delta_c^y(\Delta_c^y)^T + \Delta_c^r(\Delta_c^r)^T \\ &= S_1^{1/2} V_1^T V_1 S_1^{1/2} \\ &= S_1. \end{aligned}$$

We may now conclude that, if the weighting matrices W_1 and W_2 are chosen as in Theorem 8.4.1, the observability and controllability Grammians are diagonal and equal:

$$\mathcal{O}_c = \mathcal{C}_c = S_1$$

which proves that the controller is open-loop balanced.

A.8 Proof of Theorem 8.4.2

From the closed-loop system (8.6)-(8.7) we derive the following matrix input/output equations:

$$\begin{aligned} \begin{pmatrix} X_f \\ X_f^c \end{pmatrix} &= \underbrace{\begin{pmatrix} (A_{cl}^M)^{pp} & (A_{cl}^M)^{pc} \\ (A_{cl}^M)^{cp} & (A_{cl}^M)^{cc} \end{pmatrix}}_{=A_{cl}^M} \begin{pmatrix} X_p \\ X_p^c \end{pmatrix} + \\ & \qquad \underbrace{\begin{pmatrix} \Delta_{cl}^{p,r} & \Delta_{cl}^{p,e} \\ \Delta_{cl}^{c,r} & \Delta_{cl}^{c,e} \end{pmatrix}}_{=\Delta_{cl}} \begin{pmatrix} R_p \\ E_p \end{pmatrix}, \end{aligned} \tag{A.8}$$

$$Y_p = \underbrace{\begin{pmatrix} \Gamma_{cl}^p & \Gamma_{cl}^c \end{pmatrix}}_{=\Gamma_{cl}} \begin{pmatrix} X_p \\ X_p^c \end{pmatrix} + \underbrace{\begin{pmatrix} H_{cl}^r & H_{cl}^e \end{pmatrix}}_{=H_{cl}} \begin{pmatrix} R_p \\ E_p \end{pmatrix}. \tag{A.9}$$

where the indices p, c and cl stand for "plant", "controller" and "closed-loop" respectively. The reversed extended controllability matrix of the closed-loop system Δ_{cl} is defined in a similar way as (2.15). The extended observability matrices Γ_{cl} and the block-Toeplitz matrix H_{cl} with the closed-loop impulse response coefficients are also defined in a similar way as in (2.14) and (2.17) respectively. If the number of block-rows M in the data block-Hankel matrices is infinite, the observability and controllability Grammians of the closed-loop system can be expressed as:

$$\mathcal{O}_{cl} = \begin{pmatrix} \mathcal{O}_{pp} & \mathcal{O}_{pc} \\ \mathcal{O}_{pc}^T & \mathcal{O}_{cc} \end{pmatrix} = \begin{pmatrix} (\Gamma_{cl}^p)^T \\ (\Gamma_{cl}^c)^T \end{pmatrix} \begin{pmatrix} \Gamma_{cl}^p & \Gamma_{cl}^c \end{pmatrix},$$

$$\mathcal{P}_{cl} = \begin{pmatrix} \mathcal{P}_{pp} & \mathcal{P}_{pc} \\ \mathcal{P}_{pc}^T & \mathcal{P}_{cc} \end{pmatrix} = \begin{pmatrix} \Delta_{cl}^{p,r} & \Delta_{cl}^{p,e} \\ \Delta_{cl}^{c,r} & \Delta_{cl}^{c,e} \end{pmatrix} \begin{pmatrix} \Delta_{cl}^{p,r} & \Delta_{cl}^{p,e} \\ \Delta_{cl}^{c,r} & \Delta_{cl}^{c,e} \end{pmatrix}^T .$$

The controller is closed-loop balanced when the part of the closed-loop Grammians corresponding to the controller are equal and diagonal:

$$\mathcal{O}_{cc} = \mathcal{C}_{cc} = \text{diagonal}.$$

We will now show, in a similar way as in Appendix A.6, that these Grammians can be expressed as a function of the left and right singular vectors of the singular value decomposition:

$$W_1 W_c L_w W_2 = U_1 S_1 V_1^T$$

where:

$$\begin{aligned} \Gamma_c &= W_1^{-1} U_1 S_1^{1/2}, \\ \widehat{X}_f &= S_1^{1/2} V_1^T W_2^{-1} W_p. \end{aligned}$$

Let us therefore analyze how these Grammians can be expressed:

Observability Grammian: Substituting U_f of (7.7) in the matrix input/output equation of the plant (2.27) we find:

$$Y_p = T_c \left(\Gamma \mid H^d \Gamma_c \right) \begin{pmatrix} X_p^p \\ X_p^c \end{pmatrix} + T_c \left(H^d H_c^r \mid H^s \right) \begin{pmatrix} R_p \\ E_p \end{pmatrix}$$

where $T_c = (I - H^d H_c^y)^{-1}$. Comparing this with (A.9) we have that:

$$\Gamma_{cl} = \begin{pmatrix} T_c \Gamma & T_c H^d \Gamma_c \end{pmatrix}.$$

Therefore, the part of the closed-loop observability Grammian corresponding to the controller can be written as:

$$\begin{aligned} \mathcal{O}_{cc} &= (\Gamma_{cl}^c)^T \Gamma_{cl}^c \\ &= \Gamma_c^T (H^d)^T T_c^T T_c H^d \Gamma_c \\ &= S_1^{1/2} U_1^T W_1^{-T} (H^d)^T T_c^T T_c H^d W_1^{-1} U_1 S_1^{1/2}. \end{aligned}$$

It is now easy to see that, if $W_1 = T_c H^d$, we have $\mathcal{O}_{cc} = S_1$.

Controllability Grammian: From (A.8) one can see that:

$$X_f^c = (A_{cl}^M)^{cp} X_p + (A_{cl}^M)^{cc} X_p^c + \Delta_{cl}^{c,r} R_p + \Delta_{cl}^{c,e} E_p.$$

Substituting E_p of (2.27) and R_p of (7.8) in the above equation we have:

$$X_f^c = ((A_{cl}^{cp})^M - \Delta_{cl}^{c,e} H^{-s} \Gamma) X_p + ((A_{cl}^c)^M - \Delta_{cl}^{c,r} H_c^{-r} \Gamma_c) X_p^c +$$

$$(\Delta_{cl}^{c,r} H_c^{-r} - \Delta_{cl}^{c,e} H^{-s} H^d) U_p + (\Delta_{cl}^{c,e} H^{-s} - \Delta_{cl}^{c,r} H_c^{-r} H_c^y) Y_p.$$

As M is assumed to be infinite and since the closed-loop system is assumed to be stable we have:

$$X_f^c = \begin{pmatrix} \Delta_{cl}^{c,r} & \Delta_{cl}^{c,e} \end{pmatrix} \begin{pmatrix} -H_c^{-r} H_c^y & H_c^{-r} \\ H^{-s} & -H^{-s} H^d \end{pmatrix} W_p.$$

Comparing this with the Kalman filter state sequence obtained from the singular value decomposition:

$$\widehat{X}_f = S_1^{1/2} V_1^T W_2^{-1} W_p$$

and knowing that the states of the controller are the plant Kalman filter states ($X_f^c = \widehat{X}_f$), we directly conclude that:

$$\begin{pmatrix} \Delta_{cl}^{c,r} & \Delta_{cl}^{c,e} \end{pmatrix} = S_1^{1/2} V_1^T W_2^{-1} \begin{pmatrix} -H_c^{-r} H_c^y & H_c^{-r} \\ H^{-s} & -H^{-s} H^d \end{pmatrix}^{-1}.$$

Finally, if W_2 is chosen equal to:

$$W_2 = \begin{pmatrix} -H_c^y & I \\ I & -H^d \end{pmatrix}^{-1} \begin{pmatrix} H_c^r & 0 \\ 0 & H^s \end{pmatrix}$$

we have that:

$$\begin{pmatrix} \Delta_{cl}^{c,r} & \Delta_{cl}^{c,e} \end{pmatrix} = S_1^{1/2} V_1^T.$$

In that case, the part of the closed-loop controllability Grammian related to the controller becomes:

$$\begin{aligned} \mathcal{C}_{cc} &= \Delta_{cl}^{c,r} (\Delta_{cl}^{c,r})^T + \Delta_{cl}^{c,e} (\Delta_{cl}^{c,e})^T \\ &= S_1^{1/2} V_1^T V_1 S_1^{1/2} \\ &= S_1. \end{aligned}$$

We may now conclude that, if the weighting matrices W_1 and W_2 are chosen as in Theorem 8.4.2, the observability and controllability Grammians are diagonal and equal:

$$\mathcal{O}_{cc} = \mathcal{C}_{cc} = S_1$$

and therefore the controller is closed-loop balanced.