Katholieke Universiteit Leuven

Departement Elektrotechniek

ESAT-SISTA/TR 1999-50

Space-time processing algorithms for smart antennas in wireless communication networks [1]

Piet Vandaele [2]

PhD thesis
November 1999

[1]This report is available by anonymous ftp from *ftp.esat.kuleuven.ac.be* in the directory *pub/SISTA/vandaele/reports/99-78.ps.Z*

[2]ESAT (SISTA) - Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee), Belgium, Tel. 32/16/321799, Fax 32/16/321970, WWW: *http://www.esat.kuleuven.ac.be/sista*. E-mail: *piet.vandaele@esat.kuleuven.ac.be*. Piet Vandaele is a Research Assistant supported by the Flemish I.W.T. (the Flemish Institute for the Advancement of Scientific-Technological Research in Industry). This research work was carried out at the ESAT laboratory of the Katholieke Universiteit Leuven, in the framework of a Concerted Action Project of the Flemish Community, entitled *Model-based Information Processing Systems* (GOA/MIPS/95/99/3) as well as the IT-program of the I.W.T., *Integrating Signal Processing Systems* (ITA/GBO/T23) and was partially sponsored by IMEC (Flemish Interuniversity Microelectronics Center) and IUAP P4-02 (1997-2001): Modeling, Identification, Simulation and Control of Complex Systems. The scientific responsibility is assumed by its authors.



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

SPACE-TIME PROCESSING ALGORITHMS FOR SMART ANTENNAS IN WIRELESS COMMUNICATION NETWORKS

Promotor:
Prof. Dr. ir. M. MOONEN

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

door

Piet VANDAELE

November 1999



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

SPACE-TIME PROCESSING ALGORITHMS FOR SMART ANTENNAS IN WIRELESS COMMUNICATION NETWORKS

Jury:
Prof. Dr. ir. J. Berlamont, voorzitter
Prof. Dr. ir. M. Moonen, promotor
Prof. Dr. ir. J. Vandewalle
Prof. Dr. ir. B. De Moor
Prof. Dr. ir. S. Van Huffel
Prof. Dr. ir. A. Bultheel
Prof. Dr. ir. E. Deprettere (T.U. Delft)
Dr. ir. I. Proudler (DERA, U.K.)

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

door

Piet VANDAELE

U.D.C. 621.396.6 November 1999

© Katholieke Universiteit Leuven – Faculteit Toegepaste Wetenschappen
Arenbergkasteel, B-3001 Heverlee (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotocopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm or any other means without written permission from the publisher.

D/1999/7515/46

ISBN 90-5682-217-9

Voorwoord

Bij het einde van mijn doctoraat wil ik met plezier een aantal mensen bedanken. Mijn oprechte dank gaat uit naar mijn promotor Prof. Marc Moonen die mij overtuigde om een doctoraat te beginnen. Zijn inzicht en opbouwende kritiek vormden een welgekomen steun gedurende de afgelopen jaren en zijn gedrevenheid heeft steeds een grote indruk op mij gemaakt. Ik dank Marc voor de kansen die ik gekregen heb en kan eerlijk zeggen dat ik mezelf geen betere promotor kon wensen.

Prof. Bart De Moor wist me door zijn enthousiasme te strikken voor een eindwerk bij SISTA en stond zo mee aan de basis van dit doctoraat. Zijn "Let's do it" spirit is iets dat me altijd zal bijblijven.

Met Prof. Joos Vandewalle had ik steeds een vlotte samenwerking voor zijn vak netwerkanalyse. Zijn aandacht voor de kwaliteit van het onderwijs en zijn ruime wetenschappelijke belangstelling hebben ertoe bijgedragen dat SISTA is uitgegroeid tot een florerende onderzoeksgroep.

Ik dank Prof. Sabine Vanhuffel voor de interesse die ze steeds heeft getoond voor mijn werk. Verder wil ik alle leden van het leescommissie danken voor het zorgvuldig nalezen van de eerste versie van deze thesis.

I would like to thank dr. Ian Proudler for accepting to participate as a member of the jury. He visited Leuven several times during the last four years and his original points of view always formed the starting point for interesting discussions.

Ik wil Prof. Ed Deprettere en Prof. Adhemar Bultheel danken voor hun onmiddellijke bereidheid om in de jury te zetelen, ondanks hun drukke agenda.

Ik dank Prof. Jean Berlamont als voorzitter van de jury.

Ik dank alle leden van SISTA voor de aangename sfeer binnen en buiten de werkuren. De stimulerende werksfeer heeft ervoor gezorgd dat de laatste vier jaar zijn voorbijgevlogen. Ik denk met plezier terug aan de verrijkende samen-

werking met Patrick Schaumont en Geert Leus. Aan mijn bureaugenoten Bart, Willem, Axel, Frank en Koenraad bewaar ik goede herinneringen. Bart De Schutter en Geert Rombouts wil ik speciaal bedanken om hun nooit versagende bereidheid om UNIX problemen allerhande op te lossen en Bart Motmans dank ik voor de administratieve bijstand. Ik wil ook alle secretaresses bedanken, waarbij ik speciaal denk aan Ida en Rita. Verder dank ik mijn familie, vrienden en huisgenoten van de laatste jaren, gewoon omdat ze er waren.

Tenslotte dank ik het I.W.T. voor de financiële steun.

Abstract

Antenna arrays sample propagating waves simultaneously at different places. Space-time processing refers to digital signal processing techniques that employ antenna arrays to increase both the capacity and performance of wireless systems. Blind channel equalization/estimation algorithms based on single input multiple output data models (one user transmitting to an antenna array) or multiple input multiple output data models (multiple users transmitting to an antenna array) form a specific class of space-time algorithms that generalize results both from array signal processing and classical equalization. They estimate the channels between one or more transmitters and a receiver using the received signals only and without relying on training sequences.

This thesis is concerned with the development of deterministic subspace algorithms for blind channel equalization/estimation and focuses on two aspects of such algorithms. First, subspace algorithms are mostly based on computationally intensive matrix decompositions, while data rates in wireless systems are high. Therefore we have special attention for *computational complexity*. We show that complexity can be decreased via the use of adaptive signal processing techniques and via specific multi-user coding schemes. The second theme is the *robustness* of subspace algorithms. We demonstrate that coding provides robustness against order detection problems and additionally develop algorithms that are robust against an unknown spatial color of the additive noise.

Abstract

Antenneroosters bemonsteren propagerende golven tegelijkertijd op verschillende plaatsen. Ruimte-tijd digitale signaalverwerkingstechnieken gebaseerd op antenneroosters zijn in staat om zowel de capaciteit als de performantie van draadloze systemen op te drijven. Blinde kanaalschatting/egalisatie gebaseerd op één ingang meerdere uitgangen datamodellen (één gebruiker communiceert met een antennerooster) of meerdere ingangen meerdere uitgangen datamodellen (meerdere gebruikers communiceren met een antennerooster) vormen een specifieke klasse van ruimte-tijd algoritmen die de resultaten van zowel roostersignaalverwerking als klassieke egalisatie veralgemenen. Ze schatten de kanalen tussen één of meerdere zenders en een ontvanger met behulp van de uitgangen aan de ontvanger maar zonder trainingssequenties.

Deze thesis ontwikkelt deterministische deelruimte-algoritmen voor blinde kanaalschatting/egalisatie en concentreert zich op twee aspecten van deelruimte-algoritmen. Ten eerste steunen deelruimte-algoritmen meestal op rekenintensieve matrixontbindingen terwijl de transmissiesnelheden in draadloze communicatiesystemen hoog zijn. Daarom hebben we speciale aandacht voor de *complexiteit* van de algoritmen. We tonen aan hoe de complexiteit kan verlaagd worden door het gebruik van adaptieve signaalverwerkingstechnieken en via specifieke multi-gebruiker coderingsschema's. Het tweede thema is de *robuustheid* van deelruimte-algoritmen. We tonen aan dat codering de robuustheid tegen orde-detectie-problemen verhoogt en bijkomend ontwikkelen we deelruimte-algoritmen die robuust zijn tegen een (ongekende) spatiale kleur van de additieve ruis.

Glossary

Symbols

notation	meaning
$A(k,l)$	element on the kth row and lth column of matrix A
$A(k:l,:)$	rows k up to l of matrix A
$A(:,k:l)$	columns k up to l of matrix A
$A^\dagger$	Moore-Penrose pseudo-inverse of matrix A
A^{-1}	inverse of matrix A
A^T	transpose of matrix A
A^{-T}	inverse transpose of matrix A
A^H	Hermitian transpose of matrix A
$A^{1/2}$	Cholesky factor of square matrix A
$\hat{A}$	estimate of matrix A
$\|A\|_F$	Frobenius norm of matrix A
$\mathrm{vec}(A)$	column vector obtained by stacking columns of matrix A
$\mathrm{col}(A)$	column space of matrix A
$\mathrm{row}(A)$	row space of matrix A
A/B	orthogonal projection of matrix A onto the row space of matrix B
$A/_B\ C$	oblique projection of matrix A along the row space of matrix B onto the row space of matrix C
$0_{k\times l}$	a $k \times l$ matrix filled with zero entries
I_k	identity matrix of size $k \times k$
$\mathbf{a}$	vector $\mathbf{a}$
$\dim(\mathcal{S})$	dimension of subspace $\mathcal{S}$
$E\{x\}$	expectation of random variable x
x^*	complex conjugate of x
$\Re(x)$	real part of a complex number x
$\Im(x)$	imaginary part of a complex number x
$\|x\|$	absolute value of $x \in \mathbb{R}$, modulus of $x \in \mathbb{C}$
$\mathrm{erfc}(x)$	complementary error function evaluated at x

$\mathcal{D}(x)$	map x on the closest constellation symbol
$\delta_{i,j}$	Kronecker-δ, $\delta_{i,j} = [i = j]$
$\#(\Omega)$	number of elements in the set Ω
$\star$	don't care value
$\odot$	convolution
$\mathbb{C}$	set of complex numbers
$\mathbb{N}$	set of whole numbers
$\mathbb{R}$	set of real numbers

Acronyms

AIC	Aikaikes Information Criterion
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
CCI	CoChannel Interference
CDMA	Code Division Multiple Access
CMA	Constant Modulus Algorithm
DBPSK	Differential Binary Phase Shift Keying
DML	Deterministic Maximum Likelihood
DOA	Direction Of Arrival
DS CDMA	Direct Sequence Code Division Multiple Access
DSP	Digital Signal Processor
FDMA	Frequency Division Multiple Access
FIR	Finite Impulse Response
GF	Galois Field
GMSK	Gaussian Minimum Shift Keying
GSM	Global System for Mobile communications
HDTV	High Definition TeleVision
HF	High Frequency
HOS	Higher Order Statistics
IIR	Infinite Impulse Response
ILSE	Iterative Least Squares with Enumeration
ILSP	Iterative Least Squares with Projection
IQML	Iterative Quadratic Maximum Likelihood
ISI	Inter Symbol Interference
LMS	Least Mean Squares
LS	Least Squares
MAI	Multiple Access Interference
MDL	Minimum Description Length
MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MMSE	Minimum Mean Squared Error

MSE	Mean Squared Error
PAM	Pulse Amplitude Modulation
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
QRD	QR Decomposition
RC	Raised Cosine
RLS	Recursive Least Squares
RRC	Root Raised Cosine
RTLS	Recursive Total Least Squares
RTLS-D	RTLS for Deconvolution problems
RTLS-I	RTLS based on Inverse updating
RTLS-QR	RTLS based on QR updating
SDMA	Spatial Division Multiple Access
SIMO	Single Input Multiple Output
SISO	Single Input Single Output
SML	Stochastic Maximum Likelihood
SNR	Signal to Noise Ratio
SVD	Singular Value Decomposition
TDMA	Time Division Multiple Access
TLS	Total Least Squares

Contents

Voorwoord i

Abstract iii

Glossary v

Contents viii

Nederlandse Samenvatting xiii

1 Introduction 1

1.1 Problem statement 1
1.2 Blind equalization 5
1.3 Linear algebra tools and algorithms 6
1.4 General survey 6
1.5 Chapter by chapter overview 7

2 Concepts and tools 13

2.1 Baseband model 13
2.2 Digital modulation formats 15
2.3 SIMO data model 17
2.4 Identifiability conditions 21
2.4.1 Definition 21
2.4.2 Identifiability conditions 22
2.4.3 The input 22
2.4.4 The channel 23
2.4.5 Physical channels 24
2.5 Overview of SISO and SIMO algorithms 25

2.6 Related topics . . . 30
2.7 Multi-user identification . . . 30
2.8 Algebraic tools . . . 33
2.8.1 QR decomposition . . . 34
2.8.2 Singular value decomposition . . . 34
2.9 Conclusions . . . 35

I Adaptive single user blind symbol estimation algorithms 37

3 Viterbi and Kalman filter algorithms 41
3.1 Subspace decomposition . . . 42
3.2 Viterbi algorithm . . . 47
3.2.1 Viterbi algorithm . . . 47
3.2.2 Further discussion . . . 51
3.3 Kalman filter algorithm . . . 53
3.3.1 State space data model . . . 54
3.3.2 Kalman filter algorithm . . . 54
3.4 Conclusions . . . 57

4 Recursive total least squares algorithms 59
4.1 The total least squares problem . . . 60
4.2 RTLS based on QR updating (RTLS-QR) . . . 62
4.2.1 RTLS-QR: version 1 . . . 63
4.2.2 RTLS-QR: version 2 . . . 66
4.3 RTLS based on inverse updating (RTLS-I) . . . 71
4.3.1 RTLS-I: version 1 . . . 73
4.3.2 RTLS-I: version 2 . . . 74
4.4 RTLS for deconvolution problems (RTLS-D) . . . 79
4.4.1 Data model . . . 79
4.4.2 RTLS-D: version 1 . . . 80
4.4.3 RTLS-D: version 2 . . . 82
4.5 Conclusions . . . 85

5 Performance analysis and DSP implementation 87
5.1 Block Processing Algorithm [80] . . . 88
5.1.1 Algorithm . . . 88

5.1.2 Computational complexity 89
5.2 Complexity comparison 90
5.3 Simulation results 93
5.4 RTLS-D algorithm: complexity and performance 99
5.5 DSP implementation 101
5.5.1 Transmitter and receiver 103
5.5.2 DSP implementation 106
5.5.3 Simulation example 106
5.6 Conclusions 107

II Multi-user blind channel equalization algorithms 109

6 A multi-user subspace + Viterbi algorithm 115
6.1 Viterbi algorithm 116
6.2 Unequal channel lengths 124
6.3 Conclusions 128

7 Linear coding based blind signal separation 129
7.1 Data model 130
7.2 Algorithm C1 131
7.2.1 Algorithm C1 131
7.2.2 Code selection 134
7.3 Algorithm C2 137
7.3.1 Algorithm C2 137
7.3.2 Code selection 138
7.3.3 Further discussion 142
7.4 Algorithm C3 145
7.4.1 Data model 145
7.4.2 Algorithm C3 146
7.4.3 Code selection 148
7.5 Algorithm C1, C2 and C3 with ISI 149
7.6 Adaptive algorithms based on C2 and C3 153
7.7 Conclusions 154

8 Simulation results 155
8.1 ILSP algorithm [115] 155
8.2 Simulation results 157

8.3 Conclusions 169

III Subspace algorithms for blind channel estimation in noise fields with unknown spatial color 171

9 A stochastic algorithm based on orthogonal projections 175
9.1 Data model 176
9.1.1 Forward state space model 176
9.1.2 Backward state space model 177
9.1.3 Hankel matrix notation 178
9.2 Model assumptions 179
9.3 Stochastic subspace algorithm 179
9.3.1 Step 1: Orthogonal projections 180
9.3.2 Step 2: Channel identification 183
9.4 Conclusions 184

10 Two deterministic algorithms based on oblique projections 187
10.1 The oblique projection 188
10.2 Deterministic subspace algorithms 190
10.2.1 Deterministic subspace algorithm 1 192
10.2.2 Further discussion 194
10.2.3 Deterministic subspace algorithm 2 196
10.3 Channel order estimation 199
10.4 Conclusions 200

11 Performance analysis 201
11.1 Moulines et al. [97] 201
11.2 Tong and Zhao [121] 202
11.3 Abed-Meraim et al. [2] 203
11.4 Algorithm comparison 204
11.5 Simulation results 205
11.6 Conclusions 209

12 Conclusions and open problems 211
12.1 Contributions 211
12.2 Suggestions for further research 213

Ruimte-tijd algoritmen voor "smart antennas" in draadloze communicatie-netwerken

Nederlandse samenvatting

Hoofdstuk 1. Inleiding

De laatste jaren is de markt voor draadloze telecommunicatie-systemen enorm gegroeid. Dit heeft geleid tot een intens onderzoek naar digitale signaalverwerkingstechnieken die de capaciteit en performantie van draadloze systemen kunnen opdrijven. Bijzonder aantrekkelijk zijn zogenaamde ruimte-tijd algoritmen die facetten van roostersignaalverwerking en klassieke egalisatie combineren. Meerdere-kanaals blinde egalisatie/estimatie-algoritmen zijn ruimte-tijd algoritmen die de kanalen tussen zender en ontvanger schatten enkel gebaseerd op de uitgangen aan de ontvanger. Het centrale onderwerp van deze thesis is de ontwikkeling van dergelijke blinde algoritmen.

In een cellulair communicatiesysteem (zoals het GSM systeem) wordt het te bestrijken gebied opgedeeld in cellen. In elke cel bevindt zich een basisstation dat de communicatie verzorgt met alle mobiele gebruikers aanwezig in die cel. Periodisch wordt aan elke gebruiker een tijdsvak en een frequentieband toegekend voor communicatie met het basisstation. Omdat signalen gedempt worden bij propagatie kunnen frequenties hergebruikt worden in verschillende cellen op voorwaarde dat twee cellen die dezelfde frequentieband delen zich op voldoende grote afstand van elkaar bevinden. Deze afstand wordt de *frequentie hergebruik afstand* genoemd.

Aanvankelijk waren er weinig gebruikers in een draadloos netwerk. Operatoren waren er dan ook om bezorgd om een zo groot mogelijke dekkingsgraad te halen met zo weinig mogelijk basisstations. De kwaliteit van het ontvangen signaal wordt dan in de eerste plaats bepaald door de propagatieomgeving. Wanneer opeenvolgend doorgestuurde symbolen met elkaar interfereren, ten gevolge van een propagatievertraging veroorzaakt door de multi-pad omgeving, geeft dit aanleiding tot *inter-symbool interferentie* (ISI). Het ontvangen signaal is dan een lineair gefilterde versie van het doorgestuurde signaal. Om performantieverlies aan de ontvangstzijde te vermijden, wordt het kanaal tussen zender en ontvanger geschat (de *tijdsstructuur* van het kanaal wordt geïdentificeerd) met behulp van trainingssequenties. Vervolgens worden de kanaaleffecten gecompenseerd via kanaalegalisatie. Nadeel is wel dat trainingssequenties een aanzienlijke hoeveelheid transmissiebandbreedte opslorpen (voor het GSM systeem bedraagt dit ongeveer 18%).

De snelle groei van het aantal mobiele gebruikers heeft geleid tot kleinere frequentie hergebruik afstanden. Dit heeft echter tot gevolg dat signalen van nabij-gelegen cellen die dezelfde frequentieband aanwenden zullen interfereren met intra-cel signalen. Co-kanaal signalen komen ook voor in SDMA toepassingen. SDMA beoogt een capaciteitstoename door frequentie-hergebruik binnen een cel. Verschillende gebruikers binnen één cel delen dezelfde frequentieband en worden gescheiden via spatiale informatie. Dit fenomeen van co-kanaalsinterferentie wordt ook aangeduid als *multi-access interferentie* (MAI). Antenneroosters zijn een mogelijke oplossing [8, 9, 50, 51, 103]. Antenneroosters bestaan uit verschillende antennes die tegelijkertijd het golffront bemonsteren. Signaalverwerking met behulp van roosterantennes maakt doorgaans gebruik van richtingsschatting en bundelsturingstechnieken wat kan aanzien worden als *ruimtelijke processing* van het ontvangen signaal. Richtingsschatting en bundelsturing vereisen een accurate kennis van de roosterkarakteristiek (de antenne-transfertfunctie als functie van de invalshoek van het signaal). Dit maakt een dure calibratie noodzakelijk en wordt bovendien bemoeilijkt door de snel variërende multi-pad structuur van het draadloze kanaal.

Blinde egalisatietechnieken kunnen de transmissie-efficiëntie van bestaande systemen opvoeren door het elimineren van trainingssequenties: het kanaal wordt geschat door enkel gebruik te maken van het uitgangssignaal. Wanneer MAI aanwezig is, neemt het basissstation verschillende co-kanaal signalen waar. Zoals voor het ISI probleem kan MAI geschat en verwijderd worden via ofwel trainingssequenties of met behulp van blinde methodes. Opnieuw verdienen blinde methodes de voorkeur vanuit het standpunt van transmissie-efficiëntie.

Doorgaans zijn blinde algoritmen gebaseerd op hogere orde statistiek en maken ze gebruik van één-ingang één-uitgang datamodellen (bv. één gebruiker communiceert met één antenne in het basisstation). Meer recent werd echter duidelijk dat blinde identificatie gebaseerd op tweede orde statistiek mogelijk wordt via één-ingang meerdere-uitgangen datamodellen. Deze da-

tamodellen ontstaan wanneer één gebruiker communiceert met een basisstation uitgerust met een antennerooster. Wanneer co-kanaal signalen aanwezig zijn verkrijgen we meerdere-ingangen meerdere-uitgangen datamodellen. Blinde meerdere-gebruikers algoritmen reduceren zowel inter-symbool interferentie als co-kanaalsinterferentie en vormen een specifieke klasse van ruimte-tijd algoritmen. Ze kunnen beschouwd worden als een uitbreiding van roostersignaalverwerkingstechnieken (spatiaal filteren) en klassieke egalisatietechnieken (temporaal filteren).

Vanuit wiskundig oogpunt is lineaire algebra de aangewezen techniek voor ruimte-tijd signaalverwerking. De signalen die op elk tijdstip gemeten worden aan de antenne-uitgangen vormen een vector. Wanneer deze observaties verzameld worden over een tijdsspanne kunnen ze opgeslagen worden in matrices. De algoritmen die in deze thesis bestudeerd worden zijn *deterministische deelruimte-algoritmen*. Ze steunen op geometrische eigenschappen van matrixgebaseerde datamodellen. Matrixontbindingen zoals de QR-decompositie en singuliere waarden ontbinding spelen dan ook een erg belangrijke rol.

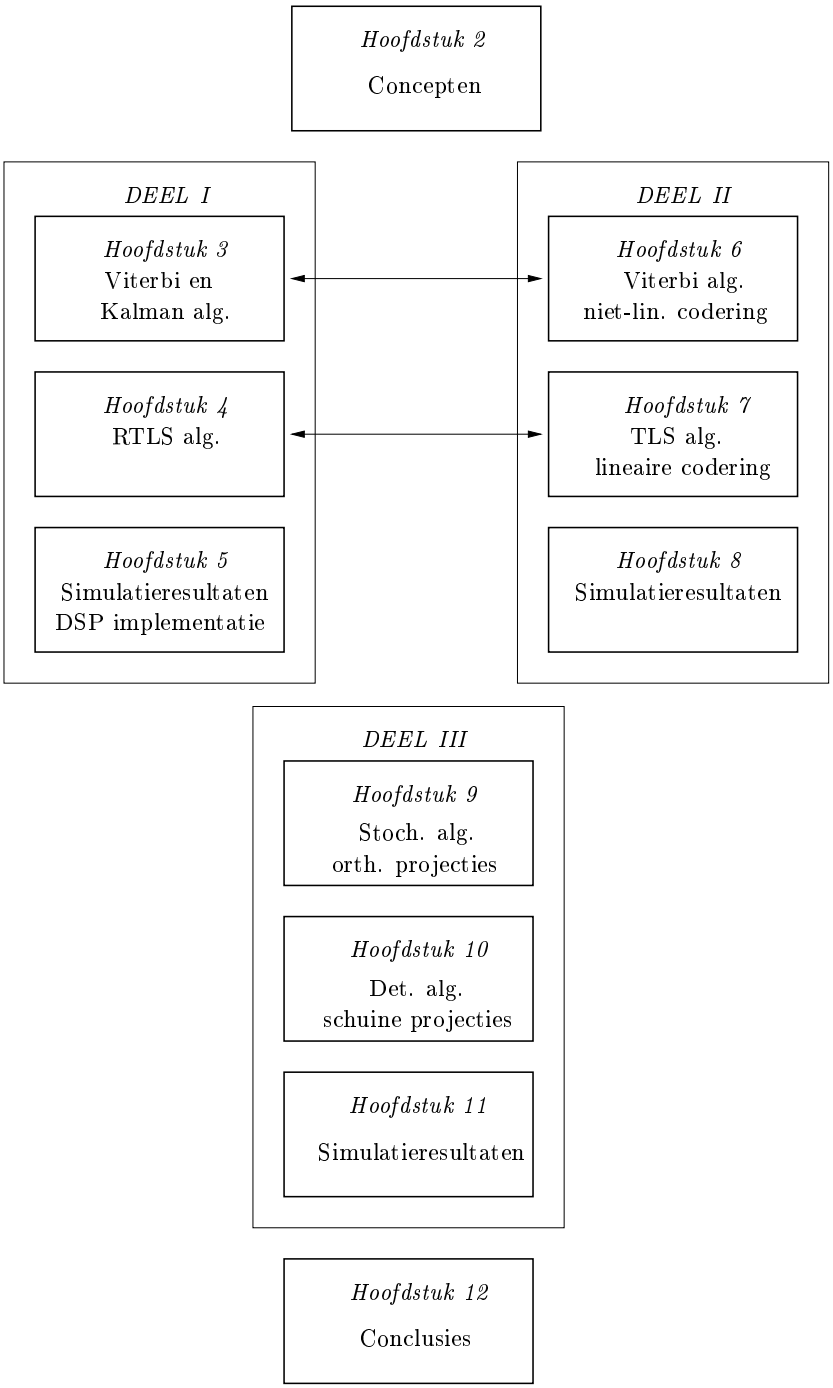
Figuur 0.1 weerspiegelt de structuur van deze thesis. De thesis bestaat uit drie delen die elk drie hoofdstukken omvatten. Deel 1 ontwikkelt adaptieve blinde deelruimte-algoritmen voor directe symboolschatting. Alhoewel erg krachtig zijn matrixdecomposities ook erg rekenintensief. De hoge transmissiesnelheden in draadloze verbindingen (GSM 270 kbit/sec) vereisen echter algoritmen met lage complexiteit. Adaptieve technieken kunnen de rekencomplexiteit (en geheugenvereisten) van deelruimte-algoritmen reduceren. In deel 2 stellen we verschillende meerdere-gebruikers blinde algoritmen voor. Het gemeenschappelijke kenmerk van de algoritmen is de toepassing van codering aan zenderzijde om zo het decoderingsproces aan ontvangstzijde te vergemakkelijken.

Naast complexiteit van deelruimte-algoritmen is hun robuustheid het tweede thema waarrond deze thesis is opgebouwd. In deel 2 tonen we aan dat codering deelruimte-algoritmen robuust maakt tegen orde-schattingsproblemen. Deel 3 legt uit hoe orthogonale en schuine projecties deelruimte-algoritmen robuust maken tegen een spatiale kleuring van de ruis.

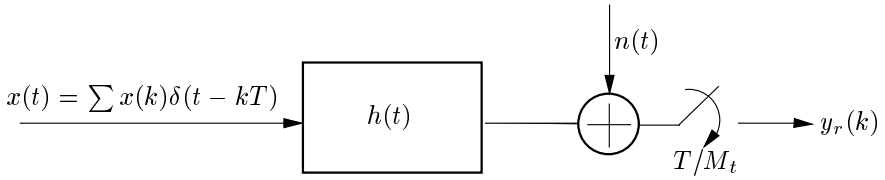
Hoofdstuk 2. Concepten en bouwblokken

Dit hoofdstuk bevat basismateriaal uit de domeinen van digitale communicatie, blinde (tweede orde) egalisatie en lineaire algebra. Met behulp van basisbandnotatie kan het ontvangen signaal voor lineaire digitale modulatie over een lineair kanaal met additieve ruis uitgedrukt worden als:

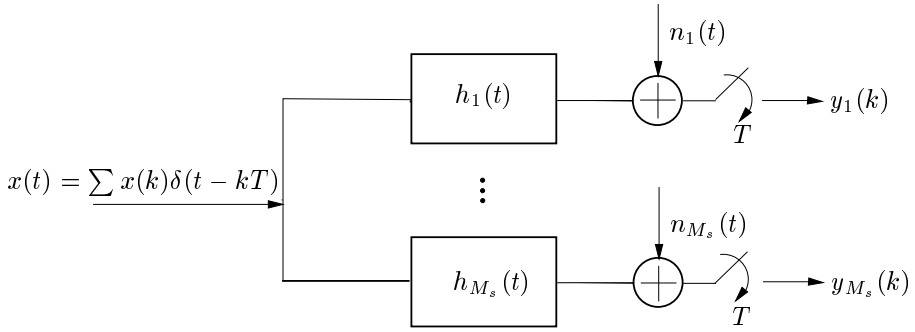
$$y(t) = \sum_k h(t - kT) \cdot x(k) + n(t) \tag{0.1}$$



Figuur 0.1: Schematisch overzicht van de thesis.



Figuur 0.2: Overbemonstering in de tijd voor een 1 ingang M_t uitgangen datamodel.



Figuur 0.3: Spatiale overbemonstering voor een 1 ingang M_s uitgangen datamodel.

waarbij $x(\cdot)$ de doorgestuurde symboolsequentie voorstelt, T is de symboolperiode en $n(t)$ de additieve ruis. $h(t)$ is de globale impulsresponsie die zowel het eigenlijke transmissiekanaal als zender- en ontvangerfilters omvat. Zoals al vermeld, steunen de algoritmen in deze thesis op één ingang meerdere uitgangen datamodellen. Meerdere uitgangen kunnen bekomen worden door ofwel het overbemonsteren in de tijd (het ontvangen signaal wordt bemonsterd aan een snelheid die M_t keer hoger ligt dan de baud rate) of door spatiaal te overbemonsteren (door het gebruik van een antennerooster bestaande uit M_s antennes aan ontvangstzijde), zie figuren 0.2 en 0.3.

Gecombineerd geven spatiale en temporale overbemonstering aanleiding tot het volgende datamodel met 1 ingang en $M = M_t \cdot M_s$ uitgangen. Definieer de uitgangs- en ruisvectoren op tijdstip k als:

$$\mathbf{y_k} \triangleq \begin{bmatrix} y_1(k) & \dots & y_M(k) \end{bmatrix}^T$$
$$\mathbf{n_k} \triangleq \begin{bmatrix} n_1(k) & \dots & n_M(k) \end{bmatrix}^T$$

dan

$$\mathbf{y_k} = \underbrace{\begin{bmatrix} h_1(L) & \dots & h_1(1) & h_1(0) \\ \vdots & & & \vdots \\ h_M(L) & \dots & h_M(1) & h_M(0) \end{bmatrix}}_{\boxed{H}} \begin{bmatrix} x(k-L) \\ \vdots \\ x(k) \end{bmatrix} + \mathbf{n_k}. \quad (0.2)$$

Met behulp van bovenstaande ingangs/uitgangs-vergelijking, kan nu als volgt een datamodel worden opgebouwd. Definieer

$$Y_{k|k+i-1} \triangleq \begin{bmatrix} \mathbf{y_k} & \mathbf{y_{k+1}} & \cdots & \mathbf{y_{k+j-1}} \\ \mathbf{y_{k+1}} & \mathbf{y_{k+2}} & \cdots & \mathbf{y_{k+j}} \\ \vdots & \vdots & & \vdots \\ \mathbf{y_{k+i-1}} & \mathbf{y_{k+i}} & \cdots & \mathbf{y_{k+i+j-2}} \end{bmatrix} \quad (0.3)$$

(het subscript verwijst naar de tijdsindices in de eerste kolom, het aantal kolommen is j), en met een gelijkaardige notatie,

$$X_{k-L|k+i-1} \triangleq \begin{bmatrix} x(k-L) & \dots & x(k-L+j-1) \\ x(k-L+1) & \dots & x(k-L+j) \\ \vdots & & \vdots \\ x(k+i-1) & \dots & x(k+i+j-2) \end{bmatrix} \quad (0.4)$$

dan,

$$Y_{k|k+i-1} = \underbrace{\begin{bmatrix} \boxed{H} & 0 & 0 & \dots \\ 0 & \boxed{H} & 0 & \dots \\ & & \ddots & \\ 0 & 0 & \dots & \boxed{H} \end{bmatrix}}_{\mathcal{H}_i} X_{k-L|k+i-1} + N_{k|k+i-1} \quad (0.5)$$

met een voor de hand liggende definitie voor de ruismatrix $N_{k|k+i-1}$. *De doelstelling van blinde egalisatie is nu om de doorgestuurde symboolsequentie te schatten enkel uitgaande van* $Y_{k|k+i-1}$. Cruciaal in het datamodel van vergelijking 0.5 zijn de blok-Toeplitz structuur van $\mathcal{H}_i$, de Hankel structuur van $X_{k-L|k+i-1}$ en de rangeigenschappen van $\mathcal{H}_i$ en $X_{k-L|k+i-1}$.

Hoofdstuk 2 bevat blinde identificeerbaarheidsvoorwaarden, analyseert de repercursies op praktische systemen en geeft een beknopt literatuuroverzicht. Het is hierbij belangrijk een onderscheid te maken tussen *stochastische* en *deterministische* algoritmen naargelang de algoritmen respectievelijk wel of geen gebruik maken van de stochastische eigenschappen van het ingangssignaal. Stochastische algoritmen vertonen een performantiedegradatie wanneer slechts een

beperkt aantal datamonsters beschikbaar zijn. Zelfs in de afwezigheid van ruis bestaat er een schattingsfout voor een eindig aantal datamonsters. Deterministische algoritmen zijn erg data-efficiënt. In de afwezigheid van ruis geven ze perfecte parameterschattingen met een eindig aantal data als de modelassumpties voldaan zijn. Deterministische algoritmen zijn echter gevoeliger voor een slechte conditie van het probleem.

Voor (deterministische) deelruimte-algoritmen [97, 1, 111, 80, 132, 82] zijn de volgende twee eigenschappen van het datamodel uit vergelijking 0.5 belangrijk.

- Als $X_{k-L|k+i-1}$ volle rijrang heeft, dan $\mathrm{col}(Y_{k|k+i-1}) = \mathrm{col}(\mathcal{H}_i)$.
- Als $\mathcal{H}_i$ volle kolomrang heeft, dan $\mathrm{row}(Y_{k|k+i-1}) = \mathrm{row}(X_{k-L|k+i-1})$.

De eerste eigenschap zegt dat de kolomruimte van de kanaalmatrix kan bepaald worden uit de kolomruimte van de uitgangsmatrix $Y_{k|k+i-1}$, wanneer de ingangsmatrix volle rijrang heeft. Het deelruimte-algoritme [97] (dat samengevat wordt in sectie 11.1) toont aan dat het mogelijk is om de kanaalparameters te identificeren vanuit deze kolomruimte. In het derde deel van deze thesis ontwikkelen we deelruimte-gebaseerde kanaalschattingsalgoritmen die eveneens gebruik maken van deze eigenschap en die bovendien robuust zijn tegen een spatiale ruiskleuring.

De tweede observatie leidt tot algoritmen die directe symboolschatting beogen [80, 81, 133]. In deze algoritmen worden de doorgestuurde symboolsequenties rechtstreeks geïdentificeerd, zonder eerst het kanaal te schatten. Eenmaal de rijruimte van $X_{k-L|k+i-1}$ bepaald is, kan de ingangssequentie geschat worden door de Hankelstructuur van $X_{k-L|k+i-1}$ uit te buiten. Het eerste deel van deze thesis ontwikkelt enkele adaptieve deelruimte-gebaseerde algoritmen voor directe symboolschatting.

Hoofdstuk 2 breidt het datamodel van vergelijking 0.5 verder uit naar meerdere gebruikers en frist de QR-decompositie en de singuliere waarden ontbinding (SWO) op samen met hun belangrijkste eigenschappen.

Hoofdstuk 3. Viterbi en Kalman filter algoritmen

Deelruimte-algoritmen vormen een belangrijke groep van deterministische blinde algoritmen die in de afgelopen jaren werden ontwikkeld. Meestal zijn deelruimte-algoritmen [97, 133, 80] gebaseerd op blokverwerking (alle ontvangen datamonsters worden in één stap verwerkt) en doorgaans hebben ze een hoge complexiteit. Adaptieve technieken kunnen de complexiteit en geheugenvereisten

van deze algoritmen reduceren. Bovendien zijn adaptieve algoritmen in staat om tijdsvariërende kanalen te volgen. In hoofdstukken 3 en 4 stellen we drie adaptieve twee-staps blinde directe symboolschattingsalgoritmen voor. De drie algoritmen hebben een identieke eerste stap, die bestaat uit een deelruimte-decompositie. Deze laat toe om een stelsel van homogene vergelijkingen in de doorgestuurde symboolsequentie te construeren. De tweede stap schat de doorgestuurde symboolsequentie door een niet-trivialiteitsbeperking toe te passen. In hoofdstuk 3 passen we een eindige alfabet beperking en een monische beperking toe. In adaptieve vorm leidt dit tot respectievelijk een Viterbi en een Kalman filter algoritme.

In de deelruimte-decompositie-stap wordt in elke iteratie k een SWO berekend van $Y_{k|k+i-1}$ (zoals gedefinieerd in vergelijking 0.5). De parameter j, het aantal kolommen van $Y_{k|k+i-1}$, is veel kleiner dan de burst lengte N. Een grotere j zorgt weliswaar voor een betere ruisuitmiddeling, maar anderzijds is voor sterk tijdsvariërende kanalen uitmiddelen over lange datasequenties niet altijd zinvol. De deelruimte-decompositie laat toe om een stelsel van homogene vergelijkingen te construeren in een segment van de doorgestuurde symboolsequentie:

$$W_k \cdot \mathbf{x}(k : k + j - 1) = 0. \tag{0.6}$$

Hierbij wordt W_k opgebouwd uit nulruimten berekend in opeenvolgende iteratiestappen. $\mathbf{x}$ is een kolomvector van lengte N die alle onbekende symbolen bevat. Met ruis proberen we in elke tijdsstap de volgende kostfunctie te minimaliseren:

$$\min_{\hat{\mathbf{x}}(k:k+j-1)} \|W_k \cdot \hat{\mathbf{x}}(k : k + j - 1)\| \qquad k = 1, \cdots, N - j + 1. \tag{0.7}$$

Deze vergelijking geeft (bij benadering) een verband tussen de (gezochte) ingangen en een 'virtueel' kanaal. Om de ingang te schatten hebben we een *niet-trivialiteitsbeperking* nodig, die dan ook automatisch toelaat om de ruis in rekening te brengen. Afhankelijk van de niet-trivialiteitsbeperking krijgt men een ander algoritme.

Het digitale ingangssignaal behoort tot een eindig alfabet: het kan op elk tijdstip slechts een eindig aantal waarden aannemen. Indien we de kostfunctie van vergelijking 0.7 minimaliseren over dit eindig alfabet krijgen we een Viterbi algoritme. Het Viterbi algoritme heeft een complexiteit die lineair is in de burstlengte N maar exponentieel in de venstergrootte j. Het algoritme heeft een hoge performantie, maar wegens de grote complexiteit is het interessant om te kijken naar minder complexe alternatieven.

Een monische beperking leidt tot een Kalman filter type algoritme dat vertrekt vanuit een stochastisch toestandsruimtemodel. De ongekende ingang wordt dan gemodelleerd als een random witte ruis term. In elke iteratiestap k veronderstellen we dat een betrouwbare schatting van $\mathbf{x}(k-1)$: $\hat{\mathbf{x}}(k-1)$ beschikbaar

is, wat toelaat om het stelsel van homogene vergelijkingen om te vormen naar een stelsel van niet-homogene vergelijkingen.

Hoofdstuk 4. Recursieve totale kleinste kwadraten algoritmen

In dit hoofdstuk passen we een kwadratische niet-trivialiteitsbeperking toe. Dit leidt tot verschillende vormen van een efficiënt recursief totaal kleinste kwadraten algoritme.

In iteratie k wordt een nieuwe W_k berekend. Dit levert een set van homogene vergelijkingen in het datasegment $\mathbf{x}(k : k+j-1)$. Op dat ogenblik kunnen alle vorige vergelijkingen in W_1, W_2,..., W_{k-1} samen met de nieuwe vergelijkingen in W_k worden samengebracht in één matrix om één groot overgedetermineerd stelsel van homogene vergelijkingen te vormen in $\mathbf{x}(1 : k + j - 1)$:

$$\underbrace{\begin{bmatrix} \boxed{W_1} & 0\ 0\ \dots \\ 0 & \boxed{W_2} \quad 0\ \dots \\ & \ddots \\ 0\ 0\ \dots & \boxed{W_k} \end{bmatrix}}_{\mathcal{W}_k} \cdot \mathbf{x}(1 : k + j - 1) \approx 0. \tag{0.8}$$

Met een kwadratische niet-trivialiteitsbeperking bekomen we:

$$\min_{\hat{\mathbf{x}}(1:k+j-1),\ \|\hat{\mathbf{x}}(1:k+j-1)\|_2=1} \|\mathcal{W}_k \cdot \hat{\mathbf{x}}(1 : k + j - 1)\|_2. \tag{0.9}$$

De oplossing voor dit probleem wordt gegeven door de rechter singuliere vector $\mathbf{v_k}$ overeenkomstig de kleinste singuliere waarde van $\mathcal{W}_k$. Zonder ruis zal $\mathbf{v_k}$ gelijk zijn aan de doorgestuurde symboolsequentie $\mathbf{x}(1 : k + j - 1)$, op een scaleringsfactor na.

We berekenen deze singuliere vector nu recursief: in iteratiestap k vertrekken we vanuit de resultaten bekomen in iteratie $k - 1$. Hierbij maken we gebruik van de *spaarsheid van* $\mathcal{W}_k$. Elk van de hierna volgende algoritmen bestaat uit twee stappen.

In het RTLS-QR algoritme bestaat de eerste stap uit QR-updating waarin de R-factor van de QR-decompositie van $\mathcal{W}_k$ wordt gevolgd. Deze stap kan aanzien worden als een datacompressie stap. In een tweede stap wordt de eigenlijke singuliere vector dan berekend via inverse iteratie gebruik makende van de R factor berekend in de eerste stap.

De eerste stap in het RTLS-I algoritme volgt de factor R^{-1} (i.p.v. de factor R), de tweede stap bestaat eveneens uit inverse iteratie. De performantie van het RTLS-I algoritme is identiek aan dat van het RTLS-QR algoritme.

In het RTLS-D algoritme wordt aangetoond hoe het RTLS-QR algoritme kan veralgemeend worden om andere deconvolutieproblemen op te lossen waarbij een stelsel van niet-homogene vergelijkingen voorkomt.

Hoofdstuk 5. Performantie-analyse en DSP-implementatie

In hoofdstuk 5 worden de complexiteit en performantie van de algoritmen uit voorgaande hoofdstukken besproken. Hieruit blijkt dat het Viterbi algoritme de hoogste complexiteit heeft, maar tevens de beste performantie biedt. Het RTLS-QR algoritme en het Kalman filter algoritme hebben een lagere, bijna identieke complexiteit. Het RTLS-QR algoritme heeft echter een betere performantie. In volgende hoofdstukken wordt daarom de monische beperking achterwege gelaten. Verder tonen simulatieresultaten aan dat de adaptieve algoritmen in staat zijn om een tijdsvariërend kanaal te volgen.

Simulatieresultaten geven aan dat de performantie van het RTLS-D algoritme dat van het standaard blok TLS algoritme benadert bij voldoende hoge SNR en dit met een sterk gereduceerde complexiteit.

Tenslotte bespreekt hoofdstuk 5 de DSP-implementatie van het RTLS-QR algoritme. Hierbij werd een volledige zender en ontvanger op DSP geïmplementeerd en wordt het blinde egalisatie-algoritme in een complete modemstructuur geïntegreerd.

Hoofdstuk 6. Een 'deelruimte + Viterbi' algoritme voor meerdere-gebruikers blinde egalisatie

In vorige hoofdstukken stelden we een aantal adaptieve één-gebruiker blinde algoritmen voor. In hoofdstuk 6 en hoofdstuk 7 tonen we aan hoe deze ideeën kunnen uitgebreid worden naar meerdere-gebruikers blinde egalisatie. Meerdere-gebruikers blinde egalisatie veralgemeent het één-gebruiker scenario in de zin dat nu twee problemen worden opgelost. Net als voor het één-gebruiker geval verstoort *inter-symbool interferentie* (ISI) de ontvangen signalen. Wanneer verschillende gebruikers tegelijkertijd dezelfde frequentieband bezetten kan bovendien slechts een lineair mengsel van deze signalen onderscheiden

worden aan de ontvanger. Dit wordt *multi-access interferentie* (MAI) of co-kanaalsinterferentie genoemd.

Blinde multi-gebruiker egalisatie verwijst naar digitale signaalverwerkingsalgoritmen die zowel ISI als MAI verwijderen. *Blinde signaalscheiding* verwijst naar signaalverwerkingstechnieken die MAI verwijderen van een ogenblikkelijk mengsel van signalen. Beide technieken steunen enkel op het uitgangssignaal mogelijk gecombineerd met een determinstische of stochastische eigenschap van het ingangssignaal, maar gebruiken geen trainingssequenties.

De originele bijdrage van de algoritmen uit hoofdstuk 6 en 7 bestaat uit het feit dat ze *meerdere-gebruikers codering* toepassen aan zendzijde om zo eenvoudige deterministische decoderingstechnieken te ontwikkelen aan ontvangstzijde. De codering verwijdert automatisch MAI en laat zo meteen toe om d 1-dimensionale problemen in parallel op te lossen (voor d gebruikers) i.p.v. (iteratief) een d-dimensionaal probleem op te lossen. Dit leidt tot eenvoudige algoritmen.

In hoofdstuk 6 wordt aangetoond hoe het één-gebruiker deelruimte + Viterbi algoritme uit hoofdstuk 3 met behulp van blokcodering kan uitgebreid worden naar meerdere-gebruikers blinde egalisatie. We ontwikkelen deterministische coderingsvoorwaarden en bespreken de signaalscheidingseigenschappen van de codering. We geven ook aan hoe codering kan beschermen tegen ISI en tegen ordeschattingsproblemen op voorwaarde dat de vensterlengte voldoende groot is en voldoende codering wordt toegepast. De complexiteit van het Viterbi algoritme is echter exponentieel in de vensterlengte. De hoeveelheid codering en/of de complexiteit van het algoritme worden aldus de beperkende factor. De meerdere-gebruikers algoritmen van hoofstuk 7 kunnen een oplossing bieden voor dit probleem.

Hoofdstuk 7. Blinde signaalscheiding gebaseerd op lineaire codering

Dit hoofdstuk introduceert drie lineaire coderingsschema's voor signaalscheiding met behulp van een kwadratische niet-trivialiteitsbeperking. De eerste stap in de algoritmen is een deelruimte-decompositie. In een tweede stap wordt code-informatie uitgebuit om een d-dimensionaal probleem (voor d gebruikers) op te splitsen in d één-dimensionale problemen.

Het eerste algoritme is gebaseerd op blokcodering en is toepasbaar voor elk modulatieschema. Het algoritme vereist echter een beperkte transmissie-overhead. De hoeveelheid codering kan afgewogen worden tegen de signaalscheidingseigenschappen van het algoritme. De minimale overhead stijgt lineair met het

aantal gebruikers.

Het tweede coderingsschema is enkel toepasbaar wanneer de datasymbolen tot een reëel alfabet behoren. Het coderingsschema behoudt de informatie-snelheid en vereist geen extra bandbreedte. Ten opzichte van het eerste coderingsschema is de beperkende factor niet langer de overhead maar het modulatieschema.

In het derde coderingsschema buiten we zender-diversiteit uit: we veronderstellen dat elke zender met ten minste twee antennes is uitgerust. Opnieuw blijft de informatiesnelheid ongewijzigd en vereist het algoritme geen extra bandbreedte. Bovendien wordt er geen beperking opgelegd aan het modulatieschema. Het aantal beschikbare antennes aan zend- en ontvangstzijde vormt hier de beperkende factor.

Eerst worden de algoritmen voorgesteld in een signaalscheidingscontext. Nadat de hoofdlijnen zijn uitgezet, worden de resultaten veralgemeend tot ISI verwijdering en tonen we aan hoe we adaptieve algoritmen kunnen bekomen. Alhoewel de nadruk opnieuw ligt op algoritmen die onmiddellijk de doorgestuurde symboolsequenties schatten, geven we ook aan hoe deze algoritmen kunnen omgevormd worden tot algoritmen waarin een egalisator wordt geschat. De bemonsterde uitgangen worden dan gefilterd door de egalisator en vormen zo een schatting van het gewenste ingangssignaal. We tonen eveneens aan hoe codering de deelruimte-gebaseerde algoritmen robuust maakt tegen ordeschattingsproblemen. Bovendien laat codering toe om de symboolsequenties van de verschillende gebruikers te schatten zonder de kanaallengtes van de individuele gebruikers te kennen.

Hoofdstuk 8. Simulatieresultaten

Dit hoofdstuk bespreekt simulatieresultaten voor de meerdere-gebruikers algoritmen van hoofstuk 6 en 7. In de afwezigheid van ISI is de performantie van de voorgestelde algoritmen vergelijkbaar met die van reeds bestaande signaalscheidingsalgoritmen. De kracht van de nieuwe algoritmen ligt vooral in hun eenvoud: alle algoritmen zijn niet-iteratief, creëren één-dimensionale problemen, ze zijn deterministisch en inherent parallel. Het Viterbi algoritme heeft de beste performantie maar de algoritmen gebaseerd op het tweede en derde coderingsschema uit hoofdstuk 7 vermijden een coderingsoverhead en zijn computationeel aantrekkelijker.

Wanneer de kanalen onderhevig zijn aan ISI hebben de nieuwe algoritmen enkele bijkomende interessante eigenschappen. Eerst onderzoeken we de invloed van de kennis van de kanaallengtes van de individuele gebruikers op de performantie. Simulatieresultaten tonen aan dat alle algoritmen een goede performantie hebben wanneer een accurate ondergrens voor de kanaallengtes

voor handen is, niet alle kanaallengtes van de individuele gebruikers moeten geïdentificeerd worden. Verder zijn de algoritmen gebaseerd op het tweede en derde coderingsschema robuust, zelfs wanneer helemaal niets over de kanaallengtes gekend is. Anderzijds kan de hoeveelheid codering die vereist wordt door het Viterbi algoritme een belemmerende factor vormen voor goede ISI resistentie. Ten tweede onderzoeken we de robuustheid van de algoritmen tegen foute ordeschatting in de deelruimte-decompositie stap. Simulatieresultaten geven aan dat in dit geval een blokverwerkingsaanpak effectiever kan zijn dan een adaptieve aanpak. De drie algoritmen van hoofdstuk 7 zijn robuust, zelfs wanneer de dimensie van de nulruimte sterk onderschat wordt. Tenslotte toont hoofdstuk 8 ook aan dat multi-gebruiker algoritmen bepaalde voordelen hebben ten opzichte van (optimale) één-gebruiker transmissie-schema's.

Hoofdstuk 9. Een stochastisch deelruimte-algoritme gebaseerd op orthogonale projecties

In vorige hoofdstukken bestudeerden we één- en meerdere-gebruikers blinde symboolschattingsalgoritmen. In hoofdstuk 9 en hoofdstuk 10 nemen we opnieuw het één-gebruiker systeem onder de loepe. De aanpak verschilt echter van die in vorige hoofdstukken. Ten eerste beogen we een schatting van het kanaal in plaats van directe symboolschatting. Nadat het kanaal is geïdentificeerd kunnen de symbolen geschat worden via bijvoorbeeld een Viterbi algoritme. Ten tweede veronderstellen de meeste blinde algoritmen die tot nog toe ontwikkeld werden dat de additieve ruis spatiaal wit is of dat de spatiale kleuring van de ruis gekend is. In hoofdstuk 9 en 10 stellen we een aantal algoritmen voor die robuust zijn tegen een (ongekende) spatiale ruiskleuring.

In hoofdstuk 9 leiden we een stochastisch algoritme voor blinde kanaalschatting af (het algoritme vormt de basis voor het tweede deterministische algoritme van hoofdstuk 10). Het buit de signaal- en ruiseigenschappen uit. Het algoritme geeft geen perfecte kanaalschattingen voor een eindig aantal observaties. Anderzijds biedt het perfecte kanaalschattingen voor een oneindig aantal data en dit onafhankelijk van het ruisniveau of de spatiale kleur van de ruis. Het algoritme is gebaseerd op een stochastisch toestandsruimtemodel en maakt gebruik van orthogonale projecties. Het is nauw verwant aan de theorie ontwikkeld in [135].

Hoofdstuk 10. Twee deterministische deelruimte-algoritmen gebaseerd op schuine projecties

In dit hoofdstuk beschouwen we het probleem van deterministische blinde kanaalschatting gebruik makende van *schuine projecties*. Schuine projecties [12, 135] laten toe om een matrix te ontbinden in twee niet-orthogonale componenten (de orthogonale projecties uit het vorige hoofdstuk ontbinden een matrix in twee orthogonale componenten). Gebruik makende van deze eigenschap ontwikkelen we twee nieuwe blinde kanaalschattingsalgoritmen. De algoritmen zijn robuust tegen een spatiale kleuring van de ruis en ze zijn deterministisch: in de afwezigheid van ruis laten ze een perfecte kanaalschatting toe met een eindig aantal data op voorwaarde dat de modelveronderstellingen voldaan zijn.

Het *eerste algoritme* gebruikt rijruimte-informatie om de kanaalparameters te schatten. De eerste stap in het algoritme is gebaseerd op twee schuine projecties. De schuine projecties ontbinden het kanaalschattingsprobleem in twee matrixvergelijkingen. In een tweede stap wordt de Hankelstructuur van de matrix met ingangssymbolen uitgebuit en worden de kanaalparameters geschat via een rang-één matrixbenadering.

Het *tweede algoritme* schat de kanaalparameters via kolomruimte-informatie, zoals de algoritmen [97, 2] en het stochastische algoritme uit het vorige hoofdstuk. De eerste stap van het algoritme is opnieuw gebaseerd op twee schuine projecties en kan geïnterpreteerd worden als een ruisverwijderingsstap. In een tweede stap worden de kanaalparameters geschat via een deelruimte-aanpak.

Hoofdstuk 11. Performantie-analyse

Dit hoofdstuk analyseert de performantie van de algoritmen uit hoofstuk 9 en 10. De performantie van het eerste deterministische algoritme is vergelijkbaar met dat van reeds bestaande kanaalschattingsalgoritmen wanneer de additieve ruis spatiaal wit is. Het algoritme vertoont echter een betere robuustheid tegen een ruiskleuring.

Het stochastische algoritme van hoofdstuk 9 en het tweede deterministische algoritme van hoofdstuk 10 zijn specifiek ontworpen voor situaties waar de ruis spatiaal gekleurd is. Ze hebben een betere performantie dan reeds bestaande deelruimte-algoritmen die eveneens bestand zijn tegen een spatiale ruiskleuring. Het determinstische algoritme presteert iets beter dan het stochastische algoritme en dit vooral wanneer het aantal uitgangsdata beperkt is.

Hoofdstuk 12. Conclusies en open problemen

Blinde egalisatie is de laatste jaren uitgegroeid tot een aantrekkelijk onderzoeksdomein wegens de mogelijke toepassing van blinde technieken in draadloze communicatiesystemen. Blinde algoritmen gebaseerd op tweede orde statistiek vormen een specifieke klasse van ruimte-tijd algoritmen die zowel verhoogde capaciteit als kwaliteit van draadloze systemen kunnen verwezenlijken.

In deze thesis hebben we een aantal deterministische blinde deelruimte-technieken voor symboolschatting, egalisator ontwerp en kanaalestimatie ontwikkeld. Alle algoritmen zijn erg data-efficiënt, ze geven excellente resultaten met een gering aantal datamonsters als de modelveronderstellingen voldaan zijn. We beschouwden zowel één- als meerdere-gebruikers blinde egalisatie. Twee thema's doorkruisten dit werk. We zochten naar algoritmen met een *lage complexiteit*. In het eerste deel ontwikkelden we adaptieve één-gebruiker algoritmen. In het tweede deel toonden we aan hoe codering aan zendzijde toelaat om aan ontvangstzijde decoderingsalgoritmen met lage complexiteit te ontwikkelen. Het tweede thema is de *robuustheid* van deelruimte-algoritmen. In het tweede deel van de thesis hebben we gezien dat het gebruik van codering de noodzaak tot identificatie van de kanaallengtes van individuele gebruikers omzeilt. Codering verhoogt eveneens de robuustheid tegen rangdetectieproblemen. Verder hebben we in het derde deel deelruimte-algoritmen ontwikkeld die robuust zijn tegen een spatiale kleuring van de ruis.

Tot nog toe zijn de meeste gepubliceerde onderzoeksresultaten echter gebaseerd op theoretische studies gekoppeld aan computersimulaties. Evaluaties op "real life" data komen zelden voor [54]. De DSP-implementatie voorgesteld in hoofdstuk 5 is nog in opbouw, maar vormt een eerste stap naar de integratie van blinde egalisatie-algoritmen in een complete modemstructuur. De opstelling zal in de nabije toekomst uitgebreid worden met antennes zodat meer realistische simulaties mogelijk worden.

In deze thesis hadden we enkel aandacht voor systemen gebaseerd op TDMA en FDMA (zoals het GSM systeem). Derde generatie draadloze systemen zullen naar alle waarschijnlijkheid gebaseerd zijn op breedbandige CDMA [24, 100]. In een CDMA systeem delen alle gebruikers dezelfde tijds- en frequentieband maar gebruiken verschillende orthogonale codes. Voor CDMA systemen kunnen gelijkaardige blinde egalisatie-technieken ontwikkeld worden als voor TDMA/FDMA gebaseerde systemen [79, 84, 75, 76].

Huidige standaardisatievoorstellen [24] voor derde generatie draadloze systemen voorzien in gebruiker-afhankelijke pilootsignalen voor kanaalschatting. Deze vergemakkelijken het gebruik van adaptieve antennetechnieken. In de context van blinde egalisatie is relatief weinig aandacht uitgegaan naar semi-blinde algoritmen. Deze algoritmen combineren elementen van trainingsgebaseerde en

blinde methodes en maken op een efficiënte manier gebruik van pilootsignalen. In het licht van het huidige standaardisatieproces kan het daarom nuttig zijn verder onderzoek te verrichten op dergelijke semi-blinde algoritmen.

Een ander interessant onderzoeksonderwerp is de toepassing van spatiale multiplexeringstechnieken in een WLAN (wireless local area network) omgeving, waar kanaalkarakteristieken traag tijdsvariërend zijn in vergelijking met klassieke mobiele communicatie. In deze context werden SDMA technieken reeds succesvol gecombineerd met OFDM (orthogonale frequentiemultiplexering) technieken [153, 154].

Chapter 1

Introduction

In recent years, the wireless communications market has expanded at a fast pace. This has spawned extensive research in the digital signal processing domain. The impetus behind this research is the potential applicability of digital signal processing techniques for performance and capacity improvements of wireless systems. Particularly promising are so-called space-time processing algorithms that combine features from array signal processing and classical equalization. Blind multi-channel estimation/equalization algorithms are space-time algorithms that estimate the channels between a transmitter and a receiver based only on the receiver outputs. This thesis is concerned with the development of such blind algorithms.

1.1 Problem statement

In a cellular communication system, like the GSM[1] system, an operator divides the covered area into distinct cells, see figure 1.1. In the center of each cell a base station establishes communication links with all mobile subscribers within that cell. It periodically assigns a frequency band and a time slot to each subscriber for communication with the base station. As signals are damped as they propagate, it is possible to reuse the frequency spectrum in different cells, provided that these cells are at a sufficiently large distance. The assignment of frequency bands to different cells is called cell planning. The distance between cells sharing the same frequency band is referred to as the *frequency reuse distance*.

Initially, wireless systems served a small number of users. The main aim of

[1] GSM: Global System for Mobile communication.

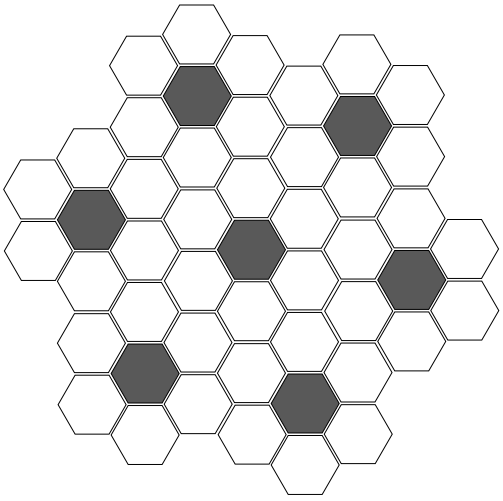


Figure 1.1: In a cellular communication system the area is divided in cells. Cells that are at a sufficient distance can reuse the same frequency. In the figure a frequency is reused in every black cell, i.e. in one out of seven cells.

operators was to cover the area with as few base stations as possible. The quality of the received signal is then primarily degraded by the propagation environment.

When subsequently transmitted symbols interfere with each other due to a propagation delay induced by the multi-path nature of the environment (see figure 1.2), a phenomenon known as *inter symbol interference* (ISI) results. The (digital) signal transmitted by the mobile user propagates through different paths towards the base station. The received signal thus consists of a superposition of copies of the same transmitted signal, possibly with a different gain, phase and delay and forms a linearly distorted version of the transmitted signal. In a hilly terrain multi-path components can have delays up to 15 μsec, in urban areas they are restricted to 10 μsec [22]. For the GSM system with a symbol period of 3.7 μsec (bit rate 270.3 kbit/s) this results in ISI over approximately 5 symbol periods. To avoid a serious performance degradation at the receiver, a compensation of the channel effects is mandatory. If the channel is known at the receiver side, it is possible to compensate for this ISI distortion via so-called *channel equalization*. However, usually the channel is unknown and estimated via training sequences [104]. The transmitter sends a symbol sequence that is known beforehand by the receiver. This allows to characterize the channel as a function of time at the receiver side: the *temporal structure* of the channel is identified. After the estimation of the channel, channel equalization is performed.

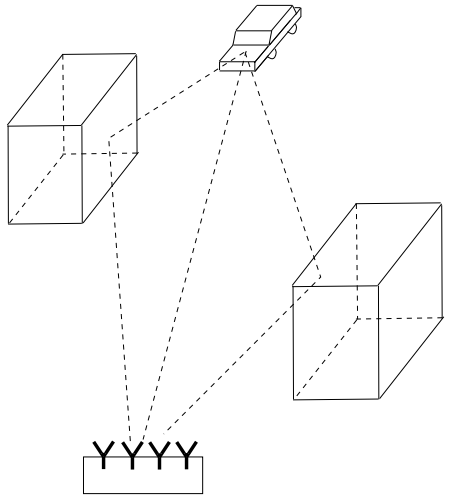


Figure 1.2: Multi-path environment in mobile telecommunications. The signal transmitted by the mobile subscriber propagates through different paths to the base station. Each path has a different delay, phase and amplitude.

Unfortunately, the wireless environment is highly time varying and the channel estimate has to be updated periodically. This leads to a considerable loss of capacity. In the GSM system [98], 26 training symbols occupy the central part of each 148 symbol periods long burst. This amounts to an 18 % capacity loss. In high-frequency (HF) communications, the time used to transmit training signals can amount to 50 % of the overall transmission [117].

The rapid growth of the number of mobile subscribers requires that the available frequency spectrum is utilized as efficiently as possible. Therefore it is important to keep the frequency reuse distance small. However, when the frequency reuse distance decreases, signals from neighboring cells using the same frequency will interfere with intra-cell signals. This phenomenon is known as *multiple access interference* (MAI) or *co-channel interference* (CCI). Multiple co-channel users also occur in SDMA[2] applications. SDMA aims at a capacity increase through frequency reuse within a cell. Several users within one cell share the same frequency band and are separated via spatial information. The installation of antenna arrays at the base station has been proposed to alleviate the problem of multiple access interference [8, 9, 50, 51, 103]. Antenna arrays consist of several antennas which sample the impinging wave front simultaneously at different places. Hence antenna arrays exploit the spatial dimension. Array signal processing research has its roots in radar signal processing where the elevation and direction of far field sources are estimated. In a wireless

[2]SDMA: spatial division multiple access.

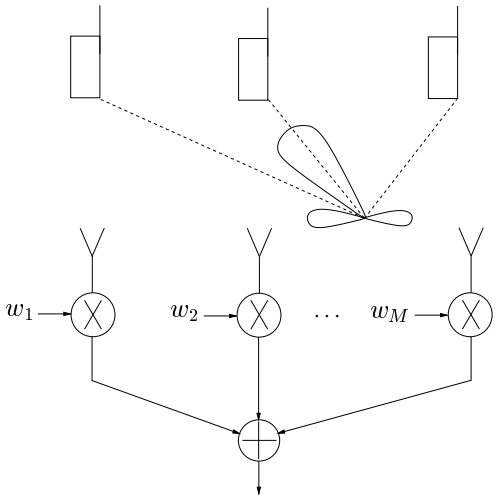


Figure 1.3: A base station exploits spatial diversity via an antenna array. The directions of arrival of the signals are estimated and beam forming is applied to enhance the signal of interest and to null the interference from co-channel users.

communications context, array signal processing has primarily been based on direction finding and beam forming and can be considered as *spatial processing* of the received signal [67], see figure 1.3. In a first step, the directions of arrival (DOA) of the impinging signals are estimated [67]. Once the directions of arrival are available, the second step involves interference reduction through adaptive beam forming (also known as "signal copy"). Consequently, systems equipped with antenna arrays are more robust against interference and smaller frequency reuse distances are allowed. However, DOA-based methods may suffer from several drawbacks [103]. First, DOA estimation requires an accurate knowledge of the array manifold (i.e. knowledge of the antenna transfer characteristics as a function of azimuth angle), which implies a calibrated array. This requires expensive calibration support and can be difficult to obtain due to to the rapidly varying multi-path structure of urban mobile communications environments. Next, DOA methods do not exploit the knowledge of the modulation format. Finally, DOA based techniques assume a well behaved channel consisting of a single or small number of paths. When multi-path propagation and delay spread are present, DOA based techniques can have poor performance.

1.2 Blind equalization

Training sequences consume a considerable part of transmission time in current wireless systems. Therefore, it is meaningful to investigate techniques which estimate the channel characteristics based only on the outputs, i.e. without employing training sequences. These techniques are known as *blind channel estimation /equalization* techniques. They rely only on the received signal and recover the input signal based on structural properties of the output signal. Eliminating training sequences has the obvious advantage of an increased transmission efficiency. Blind methods can further be useful in military interception problems where no training sequences are available. Results on the application of blind algorithms to the GSM system have been reported in [14, 32]. High-definition television (HDTV) broadcasting forms another potential application domain for blind equalization algorithms [43]. Outside a communications context, blind algorithms have been studied for solving seismic deconvolution problems [87] and for image restoration [158].

When MAI is present, the base station perceives multiple co-channel signals. MAI may arise from signals from nearby cells due to a limited frequency reuse distance or from intra-cell signals in an SDMA type of application.

As for ISI suppression, the multi-user channels can be estimated either via training signals or blindly. If training methods are used, the multiple training signals should be designed to have low cross-correlation properties so as to minimize cross coupling in the channel estimates. However, in the case of interference suppression of signals from neighboring cells, the base station may not always know the training sequence used by the interfering users. This calls again for blind separation.

Traditionally, blind algorithms have been based on the identification of single input single output (SISO) data models, e.g. one user transmitting to one antenna. Since second order statistics do not contain phase information, these algorithms exploit higher order statistics (HOS) for channel identification. However, more recently it has become clear that blind identification based on second order statistics is feasible if the underlying data model is single input multiple output (SIMO). SIMO data models naturally arise in a communications context when one user transmits to a base station equipped with an antenna array. With co-channel users present, blind identification is based on MIMO (multiple input multiple output) data models, i.e. multiple user transmitting to an antenna array at the base station. Blind multi-user algorithms reduce both inter symbol interference and co-channel interference and form a particular class of *space time processing* algorithms. They can be seen as a continuation of earlier array signal processing research where only spatial processing was employed. Similarly, they extend classical equalization techniques, where only time processing is used. Space time processing techniques fully ex-

ploit the rich structure of communication signals to improve the capacity and quality of cellular networks.

1.3 Linear algebra tools and algorithms

From a mathematical point of view, *linear algebra* comes in as a natural tool for (blind) space time signal processing. The signals captured at the antenna array at any time instant form a vector. When collected over a time frame the array outputs can be stored in matrices. The algorithms studied in this thesis are so-called *subspace algorithms* which rely on the geometrical properties of such matrix based data models. Matrix decompositions such as the QR decomposition (QRD) and singular value decomposition (SVD) play an important role in this thesis.

Though very powerful, matrix decompositions are also computationally intensive. However, the high bit rates used in mobile communications (GSM 270.3 kbit/s) emphasize the need for low complexity algorithms. Therefore one of the major themes of this thesis is the reduction of the *computational complexity* of subspace methods. In the single user case, we reduce complexity (and memory requirements) with respect to existing subspace based symbol estimation algorithms through the use of adaptive techniques. In addition, adaptive techniques are the appropriate tool to track the highly time varying mobile communications environment. In the multi-user scenario we develop low complexity channel equalization algorithms by applying specific multi-user coding schemes at the transmitter side.

The second theme in this thesis is the *robustness* of subspace algorithms. We demonstrate that coding makes subspace algorithms robust against order estimation problems and channel length detection problems commonly encountered in subspace algorithms. Further we show how orthogonal and oblique projections render subspace algorithms robust against the spatial noise color.

1.4 General survey

This section gives a general overview of this thesis. Section 1.5 provides a more detailed chapter by chapter overview.

As is seen from figure 1.4, this thesis consists of three parts. Each part comprises three chapters: two chapters that present theoretical derivations and one chapter that provides a performance analysis through simulation examples.

Chapter 2 is an introductory chapter that covers the necessary background

material.

Part one develops *deterministic single user adaptive blind symbol estimation algorithms*. The algorithms directly estimate the transmitted symbol sequence without first identifying the channel. The algorithms treated in chapters 3 and 4 consist of a (common) subspace decomposition step and a (different) symbol recovery step. Chapter 5 contains simulation results, compares the computational complexity of the different approaches and presents a DSP implementation of one of the algorithms.

The **second part** is concerned with the development of *multi-user blind equalization algorithms*. The main theme through chapters 6 and 7 is the use of coding at the transmitter side to develop deterministic low complexity decoding algorithms at the receiver side. Depending on the coding scheme that is used, different algorithms result. The algorithms developed in chapters 6 and 7 are also linked to the adaptive algorithms of part one, i.e. coding based multi-user detection becomes an almost trivial extension of single user detection.

In the **third part**, we focus again on the single user scenario. However, the approach is different from the approach in part one in that we now aim at channel estimation rather than symbol estimation. Furthermore we assume that the additive noise is spatially colored. Chapters 9 and 10 present one stochastic and two deterministic algorithms robust against the spatial noise color[3]. Chapter 11 contains simulation examples.

Finally, chapter 12 recapitulates the main contributions and gives some suggestions for further research.

1.5 Chapter by chapter overview

Chapter 2 covers relevant background material from the fields of digital communications, (second order) blind equalization and linear algebra. From the telecommunications domain, we review the base band system representation and digital modulation formats. We introduce the single input multiple output data model underlying second order blind equalization. We present identifiability conditions and give a short literature overview. We also present extensions to multi-user blind equalization. Finally we review the QR decomposition and the singular value decomposition (SVD) which form the basis for the algorithmic developments presented in this thesis.

Part one is formed by chapters 3 to 5 and contains results on subspace based adaptive algorithms for single user blind symbol estimation. The algorithms

[3] The stochastic algorithm can be considered as a preliminary version of one of the deterministic algorithms.

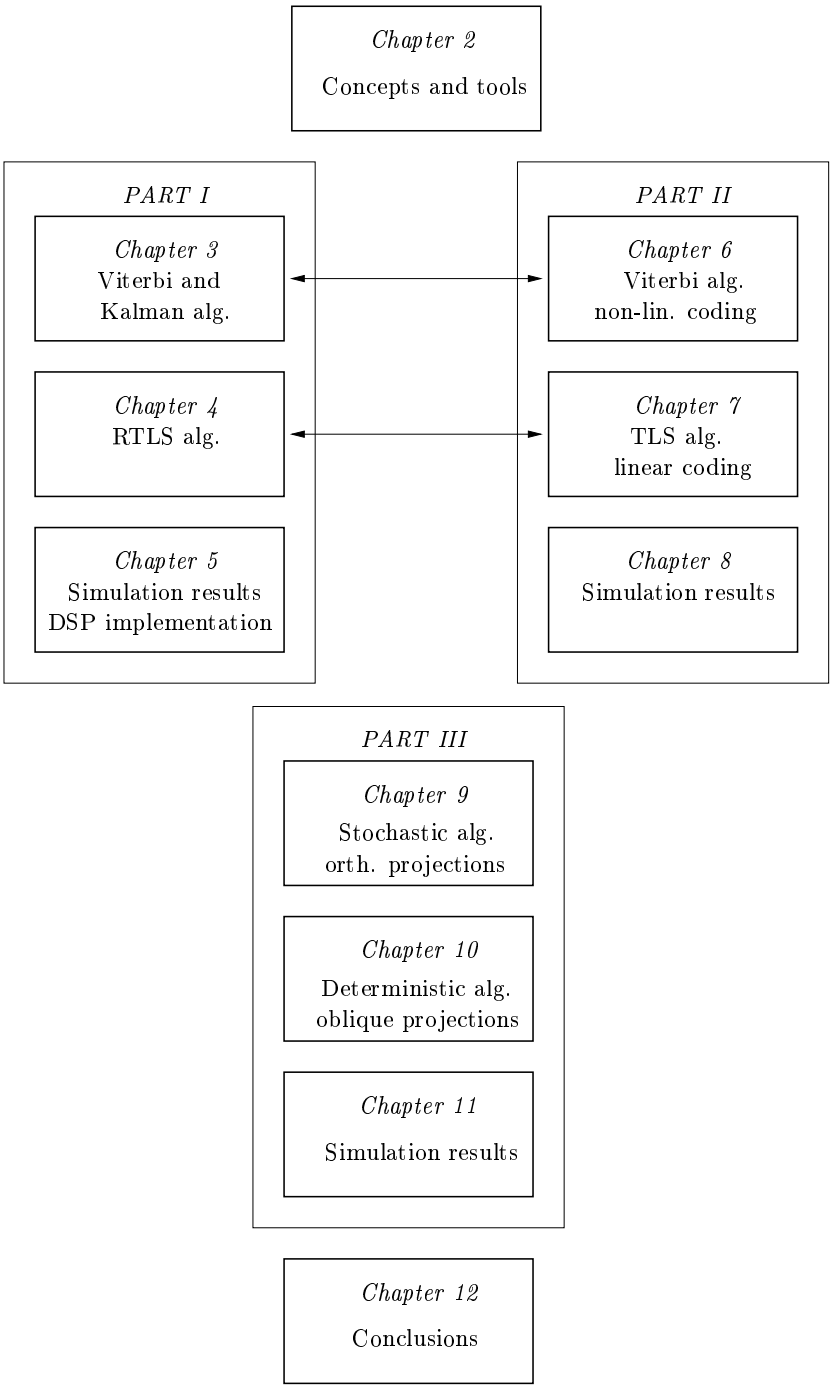


Figure 1.4: Schematic overview of the thesis.

directly estimate the transmitted symbol sequence without first identifying the channel. The aim is to develop deterministic algorithms that are able to reduce computational requirements with respect to existing deterministic subspace based direct symbol estimation algorithms. The algorithms that are developed consist of two steps. The first step is a subspace decomposition step and the second step is a symbol recovery step.

Chapter 3 commences with the subspace decomposition step that is common to all adaptive algorithms that are described. It allows to construct a set of homogeneous equations in the unknown input symbols. Depending on the non-triviality constraint that is used to solve this set of equations, different algorithms result. In chapter 3 we consider a finite alphabet and a monic constraint. In an adaptive problem formulation this leads to respectively a Viterbi and a Kalman filter algorithm.

The subspace + Viterbi algorithm was first presented in [139]. The Kalman filter approach was presented in [147] and [141].

Chapter 4 considers a quadratic non-triviality constraint for solving the set of homogeneous equations. This leads to a recursive total least squares solution. Solving a total least squares problem involves the calculation of the singular vector corresponding to the smallest singular value of a matrix. We propose two alternatives to compute this singular vector. In the first one, we use QR updating together with inverse iteration (RTLS-QR algorithm). In the second solution we use inverse updating (i.e. the factor R^{-1} of the QR decomposition is tracked) together with inverse iteration (RTLS-I algorithm). Finally, chapter 4 also explains how the RTLS-QR algorithm can be generalized to solve other deconvolution problems. The RTLS-D algorithm solves a set of non-homogeneous equations under a quadratic non-triviality constraint.

The RTLS-QR algorithm was presented in [147, 144], the RTLS-I in [140] and the RTLS-D algorithm in [145].

Chapter 5 contains simulation results. It compares the complexity and performance of the algorithms presented in chapters 3 and 4 with that of the non-adaptive subspace based symbol estimation algorithm of [80]. Chapter 5 also presents a DSP implementation of the RTLS-QR algorithm.

A simulation study was provided in [143] and the DSP implementation is described in [152].

The **second part** of this thesis discusses multi-user blind equalization algorithms. The theoretical derivations are made in chapters 6 and 7 and chapter 8 contains simulation results. In both chapters 6 and 7, coding is applied at the transmitter side to facilitate deterministic symbol recovery at the receiver. Although we propose different coding schemes, all techniques share some common features. Symbol detection is non-iterative and does not suffer from local

minima in the cost function (as is the case for some existing approaches). The algorithms are deterministic and inherently parallel. Instead of solving a d-dimensional problem for d users, the coding allows to split the problem into d one-dimensional problems which may be solved in parallel. The new idea of multi-user coding is also shown to overcome typical order detection and channel length estimation problems in a multi-user scenario.

In *chapter 6*, we show how the single user subspace + Viterbi algorithm developed in chapter 3 can be generalized to obtain a multi-user blind equalization scheme. Assigning a different coding scheme to each user at the transmitter side allows for a conceptually simple signal separation algorithm at the receiver. The Viterbi algorithm has the flexibility to allow both linear and non-linear coding (in the field of real or complex numbers).

A preliminary version of the multi-user subspace + Viterbi algorithm (without coding) was presented in [138]. Coding based versions are found in [142] and [148].

In *chapter 7*, we consider a quadratic non-triviality constraint coupled to linear coding schemes. First, the algorithms are presented in a block processing context for a blind source separation problem. After explaining the main ideas, the algorithms are extended to include multi-path effects and we discuss their adaptive implementation. We present three coding schemes. Algorithm C1 applies linear block coding and can be considered as the linear counterpart of the multi-user subspace + Viterbi algorithm of chapter 6. The algorithms C2 and C3 differ from the multi-user Viterbi algorithm and algorithm C1 in that they do not require coding overhead, i.e. there is no loss in transmission efficiency due to the presence of coding information. Algorithm C2 shows that for a one-dimensional symbol constellation (e.g. BPSK) it is possible to separate the different symbol sequences at the receiver side in a very simple way. Unfortunately, many commonly used constellations are complex two-dimensional. Algorithm C3 shows that by exploiting transmitter diversity, i.e. by using at least two antennas at each transmitter, it is possible to extend the ideas of coding scheme C2 to complex modulation formats. For the algorithms C1, C2 and C3 we also derive low complexity equalizer estimation versions.

Coding scheme C1 was analyzed in [149]. Coding scheme C2 was presented in [136] and coding scheme C3 in [137].

The simulation results of *chapter 8* compare the performance of the algorithms presented in chapters 6 and 7 with other signal separation algorithms [115, 128]. It is shown that the new algorithms do not suffer from initialization problems and provide robustness against order estimation problems. Chapter 8 also investigates the actual performance advantage of multi-user detection with respect to (optimum) single user detection.

The **third part** of this thesis again considers the single user problem. However, the problem is now approached from a different angle. First, we no longer aim at direct symbol recovery, but we develop blind channel estimation algorithms. Symbol recovery can then proceed via e.g. a Viterbi algorithm. Further, most blind channel estimation algorithms developed up till now assume that the additive noise is spatially white or else that its spatial color is known. Here, we explicitly develop algorithms that are robust against an (unknown) spatial color of the noise. Chapters 9 and 10 contain theoretical material and chapter 11 presents simulation results.

Chapter 9 introduces a stochastic algorithm for blind channel estimation. The algorithm does not provide perfect channel estimates for a finite number of data, but on the other hand it provides perfect channel estimates for a infinite number of data independent of the noise level or color. The algorithm starts from a stochastic state space description of the data model and is based on orthogonal projections.

The results of chapter 9 have been presented in [151, 146].

In *chapter 10* we again envisage a deterministic approach. We propose two blind channel estimation algorithms that provide robustness against the spatial noise color. The algorithms are based on oblique projections.

The results of chapter 10 have been presented in [150].

Chapter 11 provides simulation results for the algorithms of chapters 9 and 10 and compares their performance with other blind channel estimation algorithms [97, 121, 2].

Finally *chapter 12* contains the conclusions, it summarizes the main contributions and gives some suggestions for further research.

Chapter 2

Concepts and tools

This chapter provides background material from the areas of digital communications, (second order) blind equalization and linear algebra. From the digital communications domain, section 2.1 reviews the concept of baseband notation and section 2.2 presents the notion of digital modulation formats. The subsequent sections provide some background material on blind second order based equalization. Section 2.3 introduces the SIMO data model, section 2.4 reviews identifiability conditions and section 2.5 gives a succinct algorithmic overview. Section 2.6 discusses some extra algorithmic features while section 2.7 extends the SIMO equalization results to multi-user MIMO blind equalization. Section 2.8 discusses the QR and singular value decomposition which form the algorithmic backbone of this thesis. Finally, section 2.9 draws some conclusions.

2.1 Baseband model

Communication systems are usually narrow band: the bandwidth of the transmitted signals is much smaller than the carrier frequency, e.g. for GSM the signal bandwidth is 200 kHz while the carrier frequency is around 900 MHz. These passband systems are commonly studied by their baseband equivalent representation [104, Ch. 3].

A real-valued continuous-time signal $s(t)$ with a frequency content concentrated in a narrow band of frequencies in the vicinity of a carrier frequency f_c can be expressed as:

$$s(t) = a(t)\cos(2\pi f_c t + \phi(t))$$

where $a(t)$ denotes the amplitude and $\phi(t)$ denotes the phase of $s(t)$. If the

cosine is expanded an alternative expression for $s(t)$ results:

$$s(t) = s_i(t)\cos(2\pi f_c t) - s_q(t)(\sin 2\pi f_c t).$$

The components $s_i(t)$ and $s_q(t)$ are called the *inphase* and *quadrature* component of the signal. Alternatively $s(t)$ can also be represented by $s_b(t) = s_i(t) + is_q(t)$, such that:

$$s(t) \quad = \quad \Re\{s_b(t) \cdot e^{i2\pi f_c t}\}.$$

The signal $s_b(t)$ is then referred to as the baseband equivalent of $s(t)$.

Similarly, the baseband impulse response of the transmission channel $h(t)$ is related to the bandpass impulse response $f(t)$ by:

$$f(t) = 2\Re\{h(t) \cdot e^{i2\pi f_c t}\}.$$

In general the impulse response $h(t)$ of the equivalent baseband system is complex valued. Based on these definitions the response $r(t)$ of a bandpass system $f(t)$ to the bandpass signal $s(t)$:

$$r(t) = \int_{-\infty}^{+\infty} s(\tau) f(t-\tau) d\tau,$$

can be expressed in equivalent baseband notation as:

$$y(t) = \int_{-\infty}^{+\infty} s_b(\tau) h(t-\tau) d\tau,$$

with

$$r(t) = \Re\{y(t) e^{i2\pi f_c t}\}.$$

In the remainder of this thesis we will use the equivalent baseband data model.

Remark 2.1.1 White noise is a stochastic process having a constant power spectral density over the entire frequency range. Due to its wideband character, this type of noise cannot be expressed in terms of quadrature components. A convenient solution is to postulate that the signals and noise have passed through an ideal bandpass filter at the receiver, having a passband that includes the spectrum of the signals [104]. The resulting noise is called *bandpass white noise* and has an equivalent baseband representation. In this thesis, we will use the term white noise to refer to this baseband equivalent of bandpass white noise. We will assume baseband complex-valued zero mean "white" Gaussian noise $n(t)$ with independent real and imaginary parts, each having half of the variance of the complex noise.

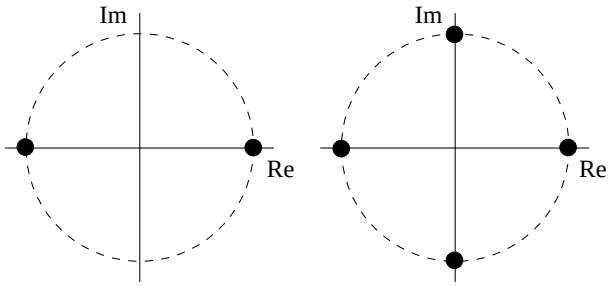


Figure 2.1: PSK constellation diagrams.

2.2 Digital modulation formats

In digital transmission, digital information is mapped on one out of η analog waveforms which are transmitted over the channel. The mapping is performed by taking blocks of $r = \log_2(\eta)$ binary digits at a time and selecting one of $\eta = 2^r$ deterministic, finite energy waveforms for transmission over the channel. In this thesis we will be concerned with modulation techniques which map a sequence of binary digits into a corresponding set of discrete phases or a set of discrete amplitudes.

Digital phase modulation results when binary digits from an information sequence are mapped into a set of discrete phases of the carrier. A η-phase-shift-keyed (PSK) signal is generated by mapping a block of $r = \log_2(\eta)$ binary digits into one of the η corresponding phases $\theta_m = 2\pi(m-1)/\eta, m = 1, 2, \ldots, \eta$. The resulting (baseband) waveforms are

$$s_b(t) = g(t)e^{i\theta_m} \tag{2.1}$$

$$= g(t)(\cos(\theta_m) + i\sin(\theta_m)) \tag{2.2}$$

where $g(t)$ is a pulse that serves the purpose of shaping the spectrum of the transmitted signal. The complex number $x = \cos(\theta_m) + i\sin(\theta_m)$ is called the transmitted *symbol*. For binary phase shift keying (BPSK) $\eta = 2$ and the possible transmitted symbols are $\Omega = \{-1, 1\}$. Ω is referred to as the *finite alphabet* for BPSK. For QPSK (quadrature phase shift keying) $\eta = 4$ and $\Omega = \{1, -1, i, -i\}$, see figure 2.1.

In *quadrature amplitude modulation* (QAM), symbol sequences are generated by encoding the input bit stream into both phase and amplitude information:

$$s_b(t) = g(t)(a_i + ia_q).$$

The complex number $x = a_i + ia_q$ is again called the transmitted *symbol*.

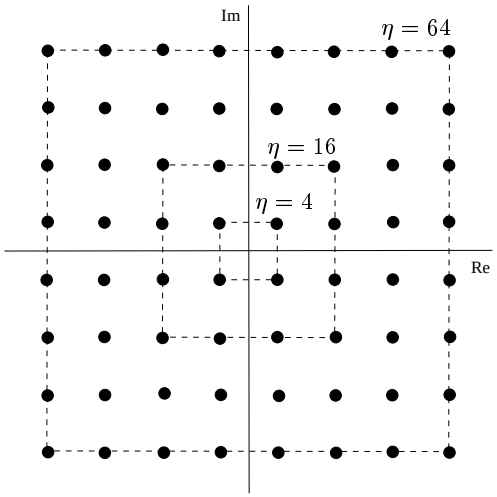


Figure 2.2: QAM constellation diagrams.

Examples of QAM constellation diagrams are given in figure 2.2.

Remark 2.2.1 The assignment of r information bits to one out of $\eta = 2^r$ possible symbols is usually done such that neighboring symbols in a constellation diagram differ only by one binary digit. This mapping is called *Gray encoding* [104, p 259]. When demodulating at the receiver, most symbol errors (incorrectly detected symbols) will then only result in a single bit error.

Remark 2.2.2 A common problem in digital communications is that of an unknown carrier phase at the receiver. The demodulated signal then only identifies the transmitted signal up to a phase factor. For a BPSK symbol sequence $\mathbf{x}$, the receiver cannot decide whether $\mathbf{x}$ or $-\mathbf{x}$ was transmitted. This problem can be overcome by encoding information in phase *differences* between successive transmitted symbols rather than by using absolute phase encoding. For example, in BPSK the information bit 1 may be transmitted by shifting the phase of the carrier by 180^o relative to the previous carrier phase, while the information bit 0 is transmitted by a zero phase shift relative to the phase in the previous signaling interval. In QPSK, the relative phase shifts between successive intervals are 0, 90^o, 180^o and -90^o corresponding to the information bits 00, 01, 11 and 10. The above process is called *differential encoding*.

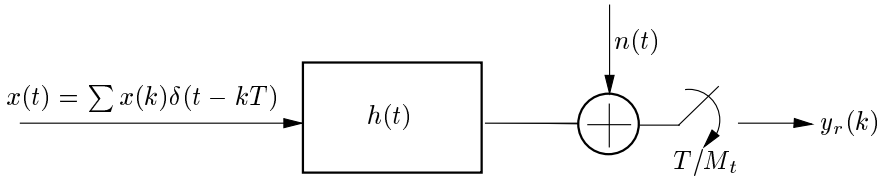


Figure 2.3: Temporal oversampling: the received signal is sampled at M_t times the symbol rate, resulting in a single input M_t output data model.

2.3 SIMO data model

Using the notion of baseband notation and digital modulation, we now introduce the data model underlying the algorithms developed in this thesis.

The received signal for linear digital modulation over a linear channel with additive noise, can be expressed as

$$y(t) = \sum_k h(t - kT) \cdot x(k) + n(t) \tag{2.3}$$

where $x(\cdot)$ represents the transmitted symbols, T is the symbol period and $n(t)$ is additive noise. $h(t)$ is the composite channel impulse response, which includes transmitter, channel and receiver filters (i.e. it includes the filter $g(t)$ of equation 2.1 and the corresponding filter at the receiver side. This definition is slightly different from the definition of $h(t)$ in section 2.1). The channel is assumed to be FIR with duration of approximately LT.

Remark 2.3.1 Although the GSM system uses a non linear phase modulation scheme GMSK (Gaussian minimum shift keying), it was shown in [70, 32, 131] that it is well approximated by a linear data model. Hence, the above data model also applies to the GSM system.

When sampled at the symbol rate, the data model of equation 2.3 represents a SISO (single input single output) system. As we will explain in section 2.5, "classical" blind equalization algorithms hinge upon this SISO data model. The algorithms developed in this thesis are based on single input, multiple output (SIMO) data models. Multiple outputs can be obtained either by temporal oversampling (by sampling the received signal faster than the symbol rate) or by spatial oversampling (by using more than one receiver antenna), see figures 2.3 and 2.4.

With a *temporal oversampling* factor M_t, the sampling instants for the received signal are $t_o + (k + \frac{r-1}{M_t}) \cdot T$ for integer k and $r = 1, 2, \ldots, M_t$. It is common

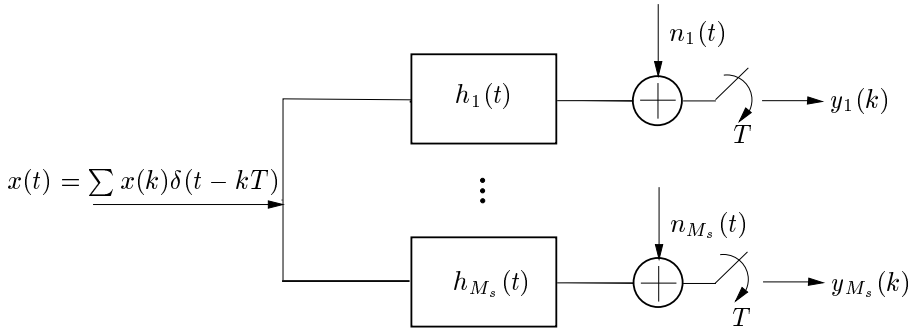


Figure 2.4: Spatial oversampling: M_s antennas at the receiver sample the received signal at the symbol rate, resulting in a one input M_s output data model.

to use a so-called polyphase description

$$\begin{cases} y_r(k) &= y(t_o + (k + \frac{r-1}{M_t}) \cdot T) \\ n_r(k) &= n(t_o + (k + \frac{r-1}{M_t}) \cdot T) \\ h_r(k) &= h(t_o + (k + \frac{r-1}{M_t}) \cdot T) \end{cases} \tag{2.4}$$

and to view the oversampled received signal as an M_t-channel output signal at the symbol rate [112, 119], see also figure 2.5.

Remark 2.3.2 The signal $y(t)$ is cyclostationary with period T [82]. When it is sampled at the symbol rate, the sampled signal is stationary. When oversampled the resulting (scalar) digital signal is cyclostationary and can be rewritten as a stationary vector sequence.

Remark 2.3.3 Communication systems employ band limited signals in order to establish a frequency division between different communication channels. A typical system has a Nyquist rate larger, but still close to the symbol rate. Hence temporal oversampling (with respect to the symbol rate) provides some additional information but the use of temporal oversampling is limited and depends on the system's (excess) bandwidth [127]. Strictly band limited systems on the other hand have an infinite impulse response (IIR), however these impulse responses are accurately modeled by their FIR approximations such that the FIR assumption of equation 2.3 remains valid.

Remark 2.3.4 The choice of the temporal oversampling factor is also influenced by the additive noise. The receiver is usually preceded by a noise-limiting

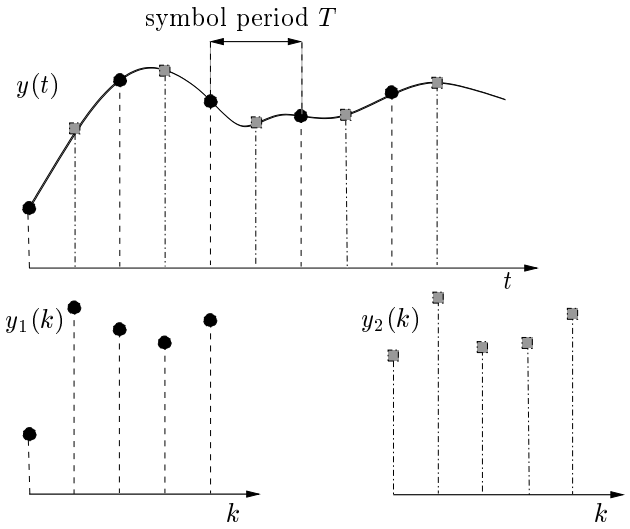


Figure 2.5: The received signal $y(t)$ is sampled at a rate $\frac{2}{T}$ (top figure), i.e. $M_t = 2$, and can be split into two sequences sampled at the symbol rate (bottom figures).

filter. The bandwidth of this (analog) filter determines the color of the noise samples. When the receiver uses a sampling rate much higher than the Nyquist rate and assumes white noise, the bandwidth of the analog filter has to be much wider than the signal bandwidth. However, this means that extra noise outside the signal bandwidth is allowed in the receiver, see also [13]. When the bandwidth of the prefilter equals the signal bandwidth, the noise power level will be lower but noise samples will be colored when the sampling rate exceeds the Nyquist rate.

Remark 2.3.5 It has been known for a long time that oversampling increases a receiver's robustness against timing errors [126]. Hence the popularity of fractionally spaced equalizers, where the delay between equalizer taps is a fraction of the symbol period.

With a *spatial oversampling factor* M_s, the impinging signal is captured simultaneously by M_s antennas that sample the received signal at the symbol rate, see figure 2.4. The antenna outputs are denoted $y_1(k),\ldots,y_{M_s}(k)$.

Both spatial and temporal oversampling may be combined to deliver a single-input M-output model where $M \triangleq M_t \cdot M_s$ [97]. Define the output vector and

noise vector at time k as:

$$\begin{aligned} \mathbf{y_k} &\triangleq \begin{bmatrix} y_1(k) & \dots & y_M(k) \end{bmatrix}^T \\ \mathbf{n_k} &\triangleq \begin{bmatrix} n_1(k) & \dots & n_M(k) \end{bmatrix}^T \end{aligned}$$

then

$$\mathbf{y_k} = \underbrace{\begin{bmatrix} h_1(L) & \dots & h_1(1) & h_1(0) \\ \vdots & & & \vdots \\ h_M(L) & \dots & h_M(1) & h_M(0) \end{bmatrix}}_{\boxed{H}} \begin{bmatrix} x(k-L) \\ \vdots \\ x(k) \end{bmatrix} + \mathbf{n_k} \quad (2.5)$$

$$= \begin{bmatrix} \mathbf{h_L} & \dots & \mathbf{h_0} \end{bmatrix} \cdot \begin{bmatrix} x(k-L) \\ \vdots \\ x(k) \end{bmatrix} + \mathbf{n_k}. \quad (2.6)$$

With the above input/output-formula, a data model can be put up as follows. Define

$$Y_{k|k+i-1} \triangleq \begin{bmatrix} \mathbf{y_k} & \mathbf{y_{k+1}} & \cdots & \mathbf{y_{k+j-1}} \\ \mathbf{y_{k+1}} & \mathbf{y_{k+2}} & \cdots & \mathbf{y_{k+j}} \\ \vdots & \vdots & & \vdots \\ \mathbf{y_{k+i-1}} & \mathbf{y_{k+i}} & \cdots & \mathbf{y_{k+i+j-2}} \end{bmatrix} \quad (2.7)$$

(the subscript refers to the time indices in the first column, the number of columns is j), and with a similar notation

$$X_{k-L|k+i-1} \triangleq \begin{bmatrix} x(k-L) & \dots & x(k-L+j-1) \\ x(k-L+1) & \dots & x(k-L+j) \\ \vdots & & \vdots \\ x(k+i-1) & \dots & x(k+i+j-2) \end{bmatrix} \quad (2.8)$$

then,

$$Y_{k|k+i-1} = \underbrace{\begin{bmatrix} \boxed{H} & 0 & 0 & \dots \\ 0 & \boxed{H} & 0 & \dots \\ & & \ddots & \\ 0 & 0 & \dots & \boxed{H} \end{bmatrix}}_{\mathcal{H}_i} X_{k-L|k+i-1} + N_{k|k+i-1} \quad (2.9)$$

with an obvious definition for the noise matrix $N_{k|k+i-1}$. The matrix $\mathcal{H}_i$ is an $M \cdot i \times (L+i)$ matrix, where i will be referred to as the so-called smoothing factor of the data model[1].

[1] In this thesis the symbol i denotes the smoothing factor as well as the imaginary unit. It should be clear from the context which one is meant.

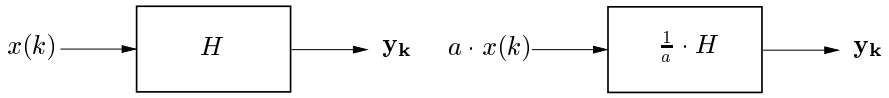


Figure 2.6: Two matrix channels with equal vector outputs $\mathbf{y_k}$.

We define the vector $\mathbf{x}$ of all unknown symbols as:

$$\mathbf{x} \stackrel{\triangle}{=} \left[\begin{array}{ccc} x(1) & \cdots & x(N) \end{array} \right]^T \tag{2.10}$$

where N denotes the burst length. *The aim of blind equalization is to compute the symbol sequence* $\mathbf{x}$ *from the sole knowledge of* $Y_{k|k+i-1}$. Crucial in the data model of equation 2.9 are the block Toeplitz structure of $\mathcal{H}_i$, the Hankel structure of $X_{k-L|k+i-1}$ and the rank properties of $\mathcal{H}_i$ and $X_{k-L|k+i-1}$.

2.4 Identifiability conditions

In the previous section we have defined a SIMO data model based on temporal and/or spatial oversampling. This section studies the blind identifiability of such a SIMO data model. First the concept of identifiability is defined. Next we review some standard results concerning identifiability based on second order statistics. We comment on the various identifiability conditions and relate these conditions to physical channel parameters.

2.4.1 Definition

The goal of blind equalization is to reconstruct the input based only on the observed output(s). However, only observing the output is not enough to completely determine the channel/input, see figure 2.6. The two matrix channels have identical vector outputs and hence identical (second order) statistics. The channel can only be determined up to a complex gain factor. Further exploiting the finite alphabet property of the input signal allows identification up to a phase rotation, i.e. the amplitude uncertainty is removed. The problem of an unknown phase is usually solved by means of differential encoding, see section 2.2. Therefore identifiability is defined in the following way.

Definition 2.4.1 *A channel is identifiable if it can be estimated up to a multiplicative constant.*

2.4.2 Identifiability conditions

The next theorem reviews the channel identifiability results of [119, 120].

Theorem 2.4.2 *Assume a channel of length L ($\mathbf{h_0} \neq 0$, $\mathbf{h_L} \neq 0$ and $\mathbf{h_r} = 0$, $r < 0$ and $r > L$). If the input signal has a linear complexity[2] greater than $2 \cdot L$ and if the matrix $\mathcal{H}_L$ is of full column rank then, in the noise free case, the channel matrix is uniquely identifiable.*

Remark 2.4.3 The condition that the channel matrix $\mathcal{H}_L$ has full column rank is also necessary for identification. The condition on the input signal is sufficient. A necessary (but not sufficient) condition is that the linear complexity of the input is greater than L, see [159].

In the next sections we comment on the restrictions on the input and the channel.

2.4.3 The input

The *linear complexity* r of a system or signal (sometimes referred to as the number of modes) is a measurement of diversity for a finite sequence [159]. If a sequence $\mathbf{x}$ has a linear complexity greater than or equal to r, then the matrix

$$\begin{bmatrix} x(1) & x(2) & \cdots & x(N-r+1) \\ x(2) & x(3) & \cdots & x(N-r+2) \\ \vdots & & & \vdots \\ x(r) & x(r+1) & \cdots & x(N) \end{bmatrix}$$

has full row rank.

Remark 2.4.4 Most statistical blind identification algorithms implicitly assume that the input is white and has a full rank covariance matrix, hence the identifiability condition on the input is often omitted.

Remark 2.4.5 In practice, the identifiability condition on the input will be almost certainly satisfied through the use of scrambling and interleaving [104].

[2] The term linear complexity is defined below.

2.4.4 The channel

Theorem 2.4.2 requires that $\mathcal{H}_L$ has full column rank. Now define

$$\begin{aligned} H_l(z) &\triangleq \sum_{k=0}^{L} h_l(k) z^{-k} \\ \mathbf{H}(z) &\triangleq \sum_{k=0}^{L} \mathbf{h_k} z^{-k} \\ &= \left[\begin{array}{ccc} H_1(z) & \cdots & H_M(z) \end{array} \right]^T \\ h(z) &\triangleq \sum_{l=1}^{M} H_l(z^M) \cdot z^{-(l-1)}. \end{aligned}$$

It has been shown in e.g. [120, 78] that the rank condition on $\mathcal{H}_L$ can be reformulated as shown below.

Property 2.4.1 *Under the identifiability conditions of Theorem 2.4.2,*

$$\begin{array}{c} \mathcal{H}_L \textit{ has full column rank.} \\ \Updownarrow \\ \textit{The rows of } \mathbf{H}(z) \textit{ do not have common zeros.} \\ \Updownarrow \\ h(z) \textit{ does not have } \frac{2\pi}{M} \textit{ spaced zeros, uniformly distributed around a circle.} \end{array}$$

We now interpret these results in several ways.

Remark 2.4.6 The $M \cdot i \times (L+i)$ matrix $\mathcal{H}_i$ $(\forall i)$ can never have full column rank for $M = 1$ (unless $L = 0$), which motivates the use of oversampling techniques. In the absence of common zeros between the subchannels, $\mathcal{H}_i$ will have full column rank at least for all $i \geq L$. Most likely however $\mathcal{H}_i$ will have full column rank as soon at its number of rows exceeds the number of columns, i.e. $M \cdot i \geq (L+i)$.

Remark 2.4.7 The rank property of $\mathcal{H}_i$ forms the basis for blind subspace algorithms. The other two identifiability conditions of property 2.4.1 will give rise to other types of (stochastic) blind identification algorithms.

Remark 2.4.8 The rank condition on $\mathcal{H}_L$ implies that all channels are FIR (otherwise the number of rows of $\mathcal{H}_L$ could never exceed the number of columns).

Remark 2.4.9 If the channels $H_l(z)$ have a common root r, then the channels $H_l(z^M)$ will have M common roots $r^{\frac{1}{M}} e^{\frac{i2\pi k}{M}}$, $k = 0, \ldots, M-1$. Hence $h(z) = \sum_{l=1}^{M} H_l(z^M) \cdot z^{-(l-1)}$ will have M zeros uniformly spaced around a circle.

2.4.5 Physical channels

The previous section explained channel identifiability conditions. This section relates this identifiability conditions to physical channel parameters. We investigate whether multi-channel systems based on temporal or spatial oversampling lack common zeros.

Temporal oversampling

When only temporal oversampling is used, Tugnait showed in [125] that multi-path channels with delays that are integer multiples of the symbol period T are unidentifiable from second order statistics. In [29] three more classes of low pass systems unidentifiable from second order statistics are described. Robustness issues were discussed in [35].

However, in [106] it was shown that when spatial oversampling is combined with temporal oversampling, the unidentifiable channel classes of [125] and [29] become identifiable. Robustness problems can further be alleviated by incorporating extra knowledge into the problem, e.g. the finite alphabet property of the input signal.

Spatial oversampling

When spatial oversampling is used, a mobile user transmits to a base station equipped with multiple antennas. It is widely assumed that when a multi-path is present, it is reasonable to assume that no common zeros exist between the different subchannels. However, if transmitter and receiver filters are also included in the channel model common zeros may result from a common pulse shaping filter see figure 2.7. A solution is to use synchronized symbol rate sampling (i.e. sampling at those specific time instants where the sampled pulse shaping filter $g(t)$ reduces to a Dirac impulse) or to combine spatial and temporal oversampling [106].

Remark 2.4.10 Perhaps even more important is the band limiting character of pulse shaping filters. These filters have infinite support in the time domain, but can be approximated by FIR filters which have slowly fading tails. These

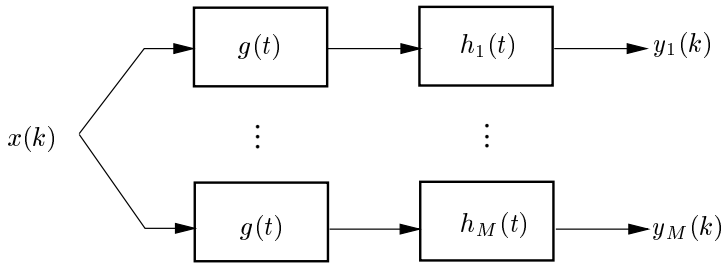


Figure 2.7: Common zeros resulting from a common pulse shaping filter $g(t)$, $h_i(t)$ denotes the physical channel.

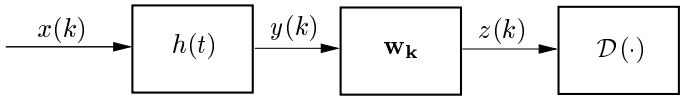


Figure 2.8: A SISO communication system, consisting of an input $x(k)$, a channel $h(t)$, an output $y(k)$ and an equalizer $\mathbf{w_k}$. The equalizer coefficients are adapted such that $z(k)$ resembles as closely as possible (a delayed version of) the input. $\mathcal{D}(\cdot)$ denotes a memoryless decision device that maps its input $z(k)$ to the closest constellation symbol.

tails may result in small (overall) channel taps, ill defined channel lengths and zeros close to the origin (and hence close to each other).

2.5 Overview of SISO and SIMO algorithms

Besides blind algorithms that exploit the structure of SIMO data models, more traditional blind equalization algorithms have been based on SISO data models. Since for the SISO case second order statistics are insufficient for channel identification/equalization these "traditional" algorithms implicitly or explicitly exploit higher order statistics [90, 48][3]. The typical setup of these algorithms is depicted in figure 2.8. $x(k)$ and $y(k)$ form respectively the channel input and output at time instant k. $h(t)$ represents the (time invariant) channel and $\mathbf{w_k}$ is a vector that contains the time varying equalizer parameters. The output $y(k)$ is filtered with $\mathbf{w_k}$ and the filter parameters are chosen such that the output of this filter $z(k)$ closely resembles a possibly delayed version of the input sequence $z(k) \approx x(k-\delta)$, with δ a demodulation delay. The output of

[3] These "traditional" algorithms can also be generalized to the SIMO case, but this will not be considered here.

the FIR equalizer $\mathbf{w_k}$ of length $B+1$ can be written as

$$\begin{aligned} z(k) &= \mathbf{w_k}^T \cdot \begin{bmatrix} y(k) \\ \vdots \\ y(k-B) \end{bmatrix} \\ &\approx x(k-\delta). \end{aligned}$$

The equalizer coefficients are usually updated via LMS-type algorithms:

$$\mathbf{w_{k+1}} = \mathbf{w_k} + \mu \cdot \begin{bmatrix} y^*(k) \\ \vdots \\ y^*(k-B) \end{bmatrix} \cdot \Psi(z(k))$$

where μ is a step size that determines the rate of adaptation, $y^*(k)$ is the complex conjugate of $y(k)$ and $\Psi(z(k))$ is a cost function. If perfect input recovery is achieved, i.e. $z(k) = x(k-\delta)$, $\Psi(z(k))$ should be zero. Depending on the cost function that is chosen different algorithms result [104, 59]. In classical training based LMS algorithms a training sequence is available at the receiver side and the cost function equals $\Psi_{LMS}(z(k)) = x(k-\delta) - z(k)$. In a (blind) decision directed mode $\Psi_{DD}(z(k)) = \mathcal{D}(z(k)) - z(k)$, where $\mathcal{D}(\cdot)$ represents a mapping to the closest constellation symbol, e.g. for BPSK the alphabet contains two elements ($\Omega = \{-1, 1\}$) and:

$$\begin{aligned} \mathcal{D}(x) &= 1 \qquad x \geq 0 \\ \mathcal{D}(x) &= -1 \qquad x < 0. \end{aligned} \tag{2.11}$$

In [109], Sato proposed the following cost function for multi level PAM signaling $\Psi_S(z(k)) = \gamma \cdot \text{sign}(z(k)) - z(k)$. Finally the constant modulus algorithm (CMA) also known as the Godard algorithm [52, 122] uses the following cost function $\Psi_{CMA}(z(k)) = z(k) \cdot (\frac{E\{|x(k)|^4\}}{E\{|x(k)|^2\}} - |z(k)|^2)$. All these algorithms have a low computational complexity, but have rather slow convergence and may suffer from spurious local minima [31, 63].

Since the work of Tong, Xu and Kailath [119, 120], blind channel identification of oversampled systems based on second order statistics or equivalent deterministic properties has been an active area of research [82, 117]. One of the attractive features of algorithms based on second order statistics is their fast convergence (typically of the order of hundreds of symbols). However, there is a need for sufficient channel diversity as explained in section 2.4. Extensive literature overviews can be found [82, 117]. Here we only sketch some main ideas.

If algorithms make assumptions about the distribution of the input statistics they are categorized as *stochastic*, while if input statistics are not exploited the algorithms are labeled as *deterministic*. Algorithms exploiting the input

statistics may suffer from performance degradation when only a limited number of output samples are available. The convergence of the input statistics requires a considerable number of samples. Even without noise, there is an estimation error for any fixed sample size. Deterministic algorithms on the other hand are very data-efficient. In the absence of noise, they provide perfect channel estimates with a finite number of data samples as long as model assumptions hold. However, deterministic methods are sensitive to performance degradations when the problem is ill conditioned.

Stochastic methods are usually based on cyclic correlation/spectral fitting techniques. Frequency domain methods exploiting cyclostationary statistics can be found in e.g. [118, 160].

Transmitter induced cyclostationarity methods [19, 49, 124] introduce cyclostationarity by applying coding at the transmitter side. They enable blind identification based on second order statistics without restriction on channel zeros, noise color or channel order overestimation errors. In [124] repetition coding is applied: blocks of signals are repeated before transmission. In [19, 49] the transmitted symbol sequences are multiplied by an (almost) periodic deterministic sequence also allowing the identification of any FIR channel irrespective of the location of channel zeros. A multi rate filterbank interpretation was given in [45].

Linear prediction methods were proposed in [111, 4, 41]. They rely on the fact that the received signal which is "moving average" by construction, can also be expressed as a finite order auto regressive signal. The input sequence is then found as the finite-horizon innovation process of the output, and can be recovered by prediction error filtering. The main advantage of linear prediction methods is their robustness against channel order overestimation.

In *optimal moment matching* methods [163, 162, 46] the second order statistics of the estimated channel are mapped in an optimal way to those of the received signal, reducing sensitivity to ill conditioned channels and channel order selection. However, the cost function to be optimized exhibits local minima which may render these techniques ineffective. In [162], an algorithm combining moment matching and subspace techniques is presented.

In *direct equalizer design* algorithms an equalizer is designed directly from the observed output data without first estimating the channel, see figure 2.9. In [47] zero forcing and MMSE equalizers were designed based on stochastic criteria. Both batch and adaptive algorithms were developed.

Deterministic algorithms are based on the algebraic structure of the data model of equation 2.9. One of the first deterministic algorithms was the *least-squares* approach [159] (also called cross relation approach). The basic obser-

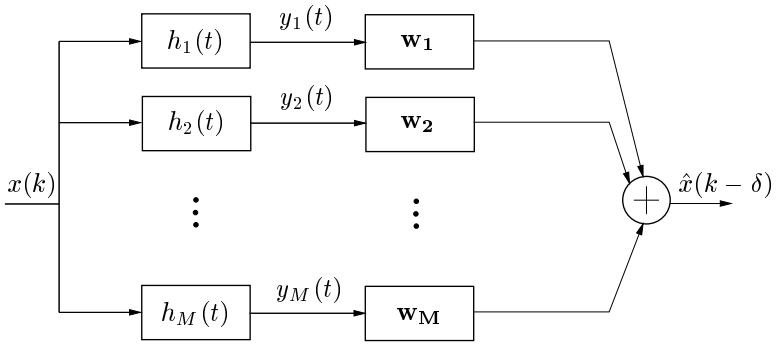


Figure 2.9: Combined channel and equalizer for a single input M output system. Each channel output $y_r(t)$ is followed by an equalizer $\mathbf{w_r}$. The equalizer outputs are combined to form an estimate $\hat{x}(k-\delta)$ of the (delayed) channel input.

vation is that:

$$\begin{aligned} h_l \odot y_k &= h_l \odot (h_k \odot x) \\ &= h_k \odot (h_l \odot x) \\ &= h_k \odot y_l, \end{aligned}$$

where $\odot$ denotes the convolution operator. The output of channel k filtered by channel l equals the output of channel l filtered by channel k. When this relationship is applied to all possible output combinations, one obtains an overdetermined set of homogeneous equations in the unknown channel parameters.

Subspace algorithms form a second class of deterministic algorithms [97, 1, 111, 80, 132, 82] and all algorithms developed in this thesis belong to this class. Subspace algorithms are based on the following two observations:

- If $X_{k-L|k+i-1}$ has full row rank, then $\mathrm{col}(Y_{k|k+i-1}) = \mathrm{col}(\mathcal{H}_i)$.
- If $\mathcal{H}_i$ has full column rank, then $\mathrm{row}(Y_{k|k+i-1}) = \mathrm{row}(X_{k-L|k+i-1})$.

The first observation says that the column space of the channel matrix can be identified from the column space of the output matrix $Y_{k|k+i-1}$, if the input matrix has full row rank. The subspace algorithm [97] (which is summarized in section 11.1) identifies the channel from its column space. Optimally weighted subspace fitting algorithms have been derived in [1, 68]. In [161] it was shown that the subspace algorithm [97] and the least squares method [159] provide identical channel estimates for $M = 2$. In the third part of this thesis we develop similar subspace based blind channel estimation algorithms. The new idea is to

precede the subspace based channel estimation by a (noise filtering) step based on orthogonal and oblique projections such that the resulting algorithms are robust against the spatial noise color.

The second observation leads to algorithms that aim at direct symbol recovery [80, 81, 133] (the algorithm [80] is summarized in section 5.1). In these algorithms the transmitted symbol sequences are identified directly, without first estimating the channel. Once the row space of $X_{k-L|k+i-1}$ has been determined, the input sequence $\mathbf{x}$ can be recovered by exploiting the Hankel structure of $X_{k-L|k+i-1}$. The algorithms [80, 81, 133] are all based on block processing (i.e. they process all received data at once). In the first part of this thesis, we show how the estimation of the row space of $X_{k-L|k+i-1}$ can be modified such as to obtain adaptive algorithms that aim at direct symbol recovery. The new algorithms are then better suited for a time varying environment and have both reduced computational and memory requirements.

In general, it is not clear whether direct symbol estimation or channel estimation (with subsequent symbol estimation) will have the best performance. Estimating the channel can be interesting for large bursts since the number of parameters to be estimated does not increase with the burst length and hence parameters can be estimated consistently. However, direct symbol estimation can be more accurate than channel estimation, especially in the presence of model mismatch or ill defined channel lengths [133].

Deterministic direct equalizer design methods and their adaptive implementation were presented in [42].

Deterministic algorithms do not make any assumption on the statistics of the input signal and hence can be applied for a wide range of source signals. However, ignoring source statistics affects asymptotic performance, especially when the identifiability condition are close to being violated.

When the additive noise is white Gaussian, the **maximum likelihood** principle can be used for second order based blind identification. Maximum likelihood (ML) estimators are usually optimal for large data records as they approximate the minimum variance unbiased estimators. The variance of ML estimators approaches the Cramer-Rao bound. Mostly, ML methods cannot be obtained in closed form. Furthermore, their implementation is hampered by the existence of local minima. Therefore, subspace (or other suboptimal) algorithms are often used as initialization for ML algorithms. Both stochastic and deterministic ML (SML and DML) techniques can be developed. In the stochastic case the input is modeled as a random variable with known distribution and only the channel is estimated. In deterministic maximum likelihood, both the channel and the input parameters are estimated. An important difference is that in DML the dimension of the parameter vector increases with the observation window, while it is fixed for SML methods. Deterministic ML methods based

on the Iterative Quadratic Maximum Likelihood (IQML) algorithm have been derived in [62, 111]. Implementations of SML algorithms have been presented in e.g. [11, 21, 58].

2.6 Related topics

For subspace algorithms, *order detection* may proceed via ad hoc eigenbased methods [133] or via the exploitation of information theoretic criteria [157]. Exact order detection may be difficult due to the long fading tails of pulse shaping filters (e.g. raised cosine pulses, see section 5.5). As explained in the previous section, order detection is not that crucial for several stochastic algorithms where only an upper bound on the channel length is required. However, for deterministic (subspace) algorithms, accurate order detection is crucial for good performance. In the second part of this thesis we will demonstrate that coding renders deterministic subspace algorithms robust against order estimation problems. When coding is applied, parameters can be successfully estimated, even when the system order is largely overestimated or when channel lengths are underestimated.

As for any problem, exploiting *prior knowledge* can strongly improve performance. In [30, 17, 99] prior knowledge of the pulse shaping filter is used. The channel $h(t)$ then consists of a known part (the pulse shaping filters) and an unknown part (the actual propagation environment). One can also exploit the physical structure of the channel. In [55] two frequency domain algorithms are derived that estimate the relative delays and spatial signature of the multi-path channels. The finite alphabet or constant modulus property of the input signal was exploited in [115, 89, 130]. Another option is to combine training based and blind approaches into so-called *semi-blind* algorithms, which outperform both blind and training based approaches, see e.g. [105].

2.7 Multi-user identification

When d users are transmitting digital symbols at the same symbol rate over a linear (vector) channel with additive noise, the sampled received (vector) signal can be expressed as:

$$\mathbf{y_k} = \sum_{r=0}^{L} H_r \cdot \mathbf{x_{k-r}} + \mathbf{n_k}. \tag{2.12}$$

Here

$$\mathbf{y_k} \quad = \quad \left[\; y_1(k) \quad \cdots \quad y_M(k) \; \right]^T$$

$$\begin{aligned} \mathbf{x_k} &= \left[\begin{array}{ccc} x^{(1)}(k) & \cdots & x^{(d)}(k) \end{array}\right]^T \\ \mathbf{n_k} &= \left[\begin{array}{ccc} n_1(k) & \cdots & n_M(k) \end{array}\right]^T \end{aligned}$$

and

$$H_k = \left[\begin{array}{cccc} h_{11}(k) & h_{12}(k) & \cdots & h_{1d}(k) \\ \vdots & & & \vdots \\ h_{M1}(k) & h_{M2}(k) & \cdots & h_{Md}(k) \end{array}\right].$$

$x^{(l)}(\cdot)$ and $h_{rl}(\cdot)$ denote respectively the transmitted symbol sequence for the lth user and the composite channel impulse response (which includes transmitter, channel and receiver filters) for the lth user and the rth channel. For the sake of a simple notation we assume that all users have equal channel lengths LT, the generalization to unequal channel orders will be given in chapters 6 and 7.

Remark 2.7.1 If $d = 1$, the data model of equation 2.12 coincides with the single user data model of equation 2.6. Again both spatial and temporal oversampling fit into the above multi-channel framework.

For the clarity of exposition, it is assumed that there is no additive noise, i.e. $\mathbf{n_k} = 0$.

As for the single user case, we construct a block Hankel matrix of output data:

$$Y_{k|k+i-1} = \left[\begin{array}{cccc} \mathbf{y_k} & \mathbf{y_{k+1}} & \cdots & \mathbf{y_{k+j-1}} \\ \mathbf{y_{k+1}} & \mathbf{y_{k+2}} & \cdots & \mathbf{y_{k+j}} \\ \vdots & \vdots & & \vdots \\ \mathbf{y_{k+i-1}} & \mathbf{y_{k+i}} & \cdots & \mathbf{y_{k+i+j-2}} \end{array}\right]$$

(again the subscript refers to the time indices in the first column), and with a similar notation we define a block Hankel matrix of input symbols:

$$X_{k-L|k+i-1} = \left[\begin{array}{ccc} \mathbf{x_{k-L}} & \cdots & \mathbf{x_{k-L+j-1}} \\ \mathbf{x_{k-L+1}} & \cdots & \mathbf{x_{k-L+j}} \\ \vdots & & \vdots \\ \mathbf{x_{k+i-1}} & \cdots & \mathbf{x_{k+i+j-2}} \end{array}\right]$$

and

$$\boxed{H} = \left[\begin{array}{cccc} H_L & \ldots & H_1 & H_0 \end{array}\right]$$

then,

$$Y_{k|k+i-1} = \underbrace{\begin{bmatrix} H & 0 & 0 & \dots \\ 0 & H & 0 & \dots \\ & & \ddots & \\ 0 & 0 & \dots & H \end{bmatrix}}_{\mathcal{H}_i} \cdot X_{k-L|k+i-1}. \tag{2.13}$$

Finally we define the $d \times N$ matrix X where each row contains the symbol sequence of one user:

$$X = \begin{bmatrix} \mathbf{x}^{(1)} \\ \vdots \\ \mathbf{x}^{(d)} \end{bmatrix}. \tag{2.14}$$

The aim of blind equalization is now to compute X from the matrix $Y_{k|k+i-1}$.

Again, algorithmic developments rely on the assumption that $\mathcal{H}_i$ has full column rank and $X_{k-L|k+i-1}$ has full row rank.

For a single user, the channel matrix has full column rank if there are no common zeros amongst the rows of $\mathbf{H}(z)$ (see property 2.4.1). In the multi-user case we have the following result [64, 3, 88].

Definition 2.7.2

$$H(z) \triangleq \sum_{r=0}^{L} H_r \cdot z^{-r}$$

- *$H(z)$ is irreducible if rank $(H(z)) = d$, $\forall z$ and rank $(H_0) = d$.*
- *$H(z)$ is column reduced if rank $(H_L) = d$.*

Theorem 2.7.3 *If $H(z)$ is irreducible and column reduced, $\mathcal{H}_i$ will have full column rank at least for all $i \geq d \cdot L$.*

In a stochastic setting the inputs are assumed to be persistently excited and statistically independent (automatically leading to full rank input matrix properties). In a deterministic framework the matrix $X_{k-L|k+i-1}$ is assumed to have full row rank. An SVD analysis of the optimal choice of temporal oversampling factor, spatial oversampling factor and smoothing factor for multiple band limited signals can be found in [127].

Using the above assumptions several subspace algorithms have been developed for multi-user blind channel or symbol estimation [133, 81, 3]. In the single user scenario the channel (input) could only be determined up to a multiplicative constant. In the multi-user scenario the channel (input) can only be recovered up to a full rank mixing matrix. I.e. for d users we obtain:

$$\begin{bmatrix} \hat{H}_0 \\ \vdots \\ \hat{H}_L \end{bmatrix} = \begin{bmatrix} H_0 \\ \vdots \\ H_L \end{bmatrix} \cdot A$$

with A a $d \times d$ mixing matrix.

The mixing matrix can then be identified using higher order statistics or using the finite alphabet or constant modulus property of the input signals [115, 6, 128, 130].

Remark 2.7.4 Estimating the channel/symbol sequences up to a mixing matrix can be interpreted as an ISI removal step. The second step is then a blind signal separation step that removes MAI. In the second part of this thesis we develop multi-user coding schemes that "automatically" remove MAI and allow to split a d-dimensional problem (for d users) into d one-dimensional problems that may be solved in parallel.

Remark 2.7.5 All single and multi-user algorithms described so far and all algorithms developed in this thesis are meant for the uplink (the mobile to base station connection). The flexibility of the mobile station is rather limited both in terms of processing power and number of antennas. Downlink algorithms (base station to mobile connection) usually foresee in a pre-equalization of the channel at the base station. The pre-equalization enables a spatially selective transmission from the base station and reduces ISI and MAI at the mobile station. The downlink channel can be estimated from feedback information of the mobile users [40]. However, this system is only feasible for slowly varying environments. In time division duplexing systems, up- and downlink work at the same frequency and channel reciprocity allows to use uplink channel estimates for downlink pre-equalization [83]. For more references on downlink space-time processing we refer to [103, pp 68-70].

2.8 Algebraic tools

The blind algorithms developed in this thesis are deterministic subspace algorithms (expect for one stochastic algorithm presented in chapter 9). They rely on two matrix decompositions: the QR decomposition (QRD) and the singular

value decomposition (SVD). In the next two subsections, we review these two decompositions along with their most important properties.

2.8.1 QR decomposition

The QR decomposition is defined in the following way.

Definition 2.8.1 *Given a matrix $A \in \mathbb{C}^{m \times n}$, $m \geq n$, then the (skinny) QR decomposition of A reads:*

$$A = Q \cdot R$$

with $Q \in \mathbb{C}^{m \times n}$ an orthonormal matrix, $Q^H \cdot Q = I_n$, and $R \in \mathbb{C}^{n \times n}$ a square upper triangular matrix.

If A has full (column) rank, this decomposition is unique up to a diagonal matrix D with unit modulus diagonal elements. The QR decomposition has a number of properties which make it a valuable tool in linear algebra problems.

Property 2.8.1 *The matrix Q forms an orthonormal basis for the column space of A.*

Property 2.8.2 *The QR decomposition can be used to solve an overdetermined set of linear equations. Assume a matrix $A \in \mathbb{C}^{m \times n}$, a vector $\mathbf{b} \in \mathbb{C}^{m \times 1}$ with $m \geq n$ and $A = Q \cdot R$. Then the optimal solution $\mathbf{x}$ (in least squares sense) to the equation $A \cdot \mathbf{x} \approx \mathbf{b}$ can be computed as:*

$$\begin{aligned} (Q^H \cdot A) \cdot \mathbf{x}_{LS} &= Q^H \cdot \mathbf{b} \\ R \cdot \mathbf{x}_{LS} &= Q^H \cdot \mathbf{b}. \end{aligned}$$

The last equation is easily solved through back substitution.

The QR decomposition can be computed and updated via Givens or Householder transformations [53].

2.8.2 Singular value decomposition

The singular value decomposition is a fundamental tool in system identification, signal processing and control [110, 155, 129]. It is defined in the following way:

Definition 2.8.2 *Given a matrix $A \in \mathbb{C}^{m \times n}$, $m \geq n$, the singular value decomposition (SVD) of A is defined as:*

$$A = U \cdot \Sigma \cdot V^H$$

with $U \in \mathbb{C}^{m \times m}$ and $V \in \mathbb{C}^{n \times n}$ unitary matrices and $\Sigma \in \mathbb{R}^{m \times n}$ a diagonal matrix containing the singular values σ_r, $r = 1, \ldots n$ and $\sigma_r \geq \sigma_{r+1}$.

Since,

$$\begin{aligned} A^H \cdot A &= V \cdot \Sigma^H \cdot \Sigma \cdot V^H \\ A \cdot A^H &= U \cdot \Sigma \cdot \Sigma^H \cdot U^H \end{aligned}$$

the square of the singular values σ_r^2, $r = 1, \ldots, n$ are the eigenvalues of the Hermitian, nonnegative definite matrices $A^H \cdot A$ and $A \cdot A^H$. The right singular vectors V and the left singular vectors U are the corresponding eigenvectors. Many algorithms are based on an eigenvalue decomposition of the matrix $A^H \cdot A$. However, for an actual implementation, the SVD is preferred over the eigenvalue decomposition because it avoids the explicit calculation of the matrix product $A^H \cdot A$. Excellent numerical algorithms exist to compute the SVD [53].

In a subspace estimation context, a common problem is to determine a low rank subspace in a (noise corrupted) higher dimensional space. The following property shows that the SVD provides an optimal estimate of this subspace.

Property 2.8.3 *Given the SVD of a matrix $A \in \mathbb{C}^{m \times n}$, $\sum_{l=1}^{n} \sigma_l \cdot \mathbf{u_l} \cdot \mathbf{v_l}^H$, then the matrix $B \in \mathbb{C}^{m \times n}$, nearest in Frobenius norm or two-norm to A and of rank $r \leq \min(m, n)$:*

$$\min_{B,\ rank(B)=r} \|A - B\|_F$$

$$\min_{B,\ rank(B)=r} \|A - B\|_2$$

is given by the truncated SVD of A:

$$B = \sum_{l=1}^{r} \sigma_l \cdot \mathbf{u_l} \cdot \mathbf{v_l}^H$$

with $\mathbf{u_l}$,$\mathbf{v_l}$ respectively the lth column of U, V.

2.9 Conclusions

In this introductory chapter we have reviewed basic concepts from the telecommunications, blind equalization and linear algebra domain.

From the telecommunications area, we have shown that a baseband system representation forms a valuable tool. We further demonstrated how digital information sequences can be mapped to transmitted symbol sequences via various modulation formats.

We have reviewed fundamentals from blind equalization. First we have presented identifiability conditions. We gave special attention to the practical implications of these conditions and indicated potential problems. Next, we gave a short literature overview and showed how the transition between single and multi-user blind equalization can be made.

Finally, we have presented the QR and singular value decomposition along with some of their properties.

Part I

Adaptive single user blind symbol estimation algorithms

Deterministic blind algorithms are attractive because of their finite sample convergence property: they provide perfect parameter estimates with a finite number of data samples as long as model assumptions hold (noiseless case). Subspace algorithms take an important place amongst the deterministic blind algorithms developed in recent years [97, 133, 80, 1, 111]. These algorithms divide the observation space in so-called signal and noise subspaces that are orthogonal to each other. This orthogonality property then allows to estimate the channel [97, 1, 111] (see also the third part of this thesis) or to directly estimate the symbols [80, 133]. The mathematical tool underlying subspace algorithms is the singular value decomposition (SVD).

Two disadvantages are usually associated with subspace algorithms. First they are sensitive to modeling errors, e.g. wrong estimates of system orders or channel lengths, while many stochastic algorithms are robust against these problems (see also section 2.5). In the second part of this thesis we will develop multi-user coding schemes that make subspace algorithms robust against these estimation problems. Secondly, subspace algorithms are based on block processing (i.e. they process all received data in one step) and have a high computational complexity. *The aim of the algorithms developed in the first part of this thesis is to reduce the computational complexity and memory requirements of subspace based blind symbol estimation algorithms via the use of adaptive signal processing techniques. Furthermore, adaptive algorithms allow time-varying channels to be tracked.*

Several (not subspace based) adaptive blind algorithms have already been proposed in literature. In [61] an adaptive channel estimation algorithm based on the least squares approach of [159] was presented. The algorithm adaptively estimates the channel using a least mean squares (LMS) or recursive least squares (RLS) approach and inserts this channel estimate in a Viterbi algorithm for subsequent symbol recovery. In [47] and [42] direct equalizer design methods were developed both for adaptive and batch mode operation. The adaptive algorithms proposed in [47] and [42] are also based on RLS and LMS solutions. In [116] a direct symbol estimation algorithm is proposed based on the Viterbi algorithm. The algorithm exploits the algebraic structure of the received data sequence together with statistical source properties.

In part one (chapters 3 and 4) we present adaptive algorithms which aim at direct symbol estimation. Unlike the adaptive algorithms mentioned above, the algorithms presented here use an adaptive subspace decomposition as a first step of the algorithm. This subspace decomposition allows to construct a set of homogeneous equations in the unknown transmitted symbols. The main contribution of this first step is the reformulation of the original subspace decomposition idea of [80] such that it allows an adaptive implementation. The second step then recovers the transmitted symbol sequence from this set of homogeneous equations by applying a non-triviality constraint. We apply a finite alphabet, a monic and a quadratic non-triviality constraint which lead

to respectively a Viterbi algorithm (section 3.2), a Kalman filter algorithm (section 3.3) and two recursive total least squares (RTLS) algorithms (chapter 4). While the Viterbi and Kalman filter algorithms are fairly standard building blocks, the RTLS algorithms are not. The new algorithms exploit the structure of the set of homogeneous equations and apply a specific decision feedback mechanism to obtain recursive algorithms with constant computational and memory requirements in each iteration step, as will be explained.

In chapter 5 we compare the computational complexity and the performance of these algorithms with that of the subspace based block processing symbol estimation algorithm [80]. Finally, we succinctly discuss a DSP implementation of the recursive total least squares algorithm.

Chapter 3

Viterbi and Kalman filter algorithms

The outline of this chapter is as follows. Section 3.1 presents the subspace decomposition step that is common to all adaptive algorithms developed in this thesis. This subspace decomposition constructs a set of homogeneous equations in the unknown transmitted symbols. The idea of using a subspace decomposition to construct a homogeneous matrix equation in the unknown transmitted symbols is not new, see [80] (this algorithm is summarized in section 5.1) and [133]. *However, here we combine several subspace decompositions to construct an alternative problem formulation which lends itself better to an adaptive implementation.*

The second step involves symbol recovery and adaptively solves this set of homogeneous equations under an appropriate non-triviality constraint. Section 3.2 presents a Viterbi algorithm that applies a finite alphabet constraint. Section 3.3 proposes a Kalman filter solution that is based on a monic constraint. In both sections we rearrange the set of homogeneous equations such that the well known Viterbi and Kalman filter algorithm can be applied straightforwardly. In the next chapter we will present a recursive total least squares solution that applies a quadratic non-triviality constraint.

As will be shown, the Viterbi algorithm has both high performance and complexity. The Kalman filter has a lower computational complexity at the expense of some performance loss. Finally section 3.4 presents some conclusions.

3.1 Subspace decomposition

In this section we introduce the first step of the algorithms presented in the next sections/chapter. It consists of an adaptive subspace decomposition which allows to construct a set of homogeneous equations in the unknown transmitted symbols. The actual recovery of the transmitted symbols happens in a second step and will be described in sections 3.2, 3.3 and chapter 4. The subspace decomposition is adaptive in the sense that a new noise subspace (of a small matrix) is computed in each iteration step. It forms an alternative for the block processing subspace approach of [80] (which is summarized in section 5.1) where only one noise subspace is computed from a large matrix. We start from the data model of equation 2.9:

$$Y_{k|k+i-1} = \mathcal{H}_i \cdot X_{k-L|k+i-1} + N_{k|k+i-1},$$

where the output matrix $Y_{k|k+i-1}$ and the noise matrix $N_{k|k+i-1}$ are $(M \cdot i) \times j$ matrices, the channel matrix $\mathcal{H}_i$ is an $(M \cdot i) \times (L+i)$ matrix and the input matrix $X_{k-L|k+i-1}$ is an $(L+i) \times j$ matrix. In each iteration step k (i.e. in each symbol period), a new SVD is computed,

$$\begin{aligned} Y_{k|k+i-1} &= U_k \cdot \Sigma_k \cdot V_k^H & (3.1) \\ &= \begin{bmatrix} U_k^s & U_k^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma_k^s & 0 \\ 0 & \Sigma_k^\perp \end{bmatrix} \cdot \begin{bmatrix} V_k^{sH} \\ V_k^{\perp H} \end{bmatrix}. & (3.2) \end{aligned}$$

The parameter j, i.e. the number of columns of $Y_{k|k+i-1}$, is much smaller than the burst length N. Increasing j provides a better noise averaging, but on the other hand, for highly time-varying channels, averaging over long data sequences may not be meaningful and so the usage of smaller matrices may be imposed by practical considerations anyway. The influence of j both on complexity and performance will be analyzed in chapter 5.

It is assumed that $\mathcal{H}_i$ is of full column rank (see section 2.4.4). Without noise, the matrix $Y_{k|k+i-1}$ has rank $L+i$ and hence Σ_k will contain $L+i$ singular values different from zero (i.e. the diagonal elements of Σ_k^s). With noise, Σ_k will contain $L+i$ 'large' singular values above the 'noise threshold'. The value $L+i$ can then be estimated from the gap in the singular value spectrum or from information theoretic criteria such as Aikaikes information criterion (AIC) or minimum description length (MDL) see [5, 107, 157]. Since i is known, the order of the system L can be determined. For the time being, we consider the *noiseless case*, in the next (sub)sections we will then account for the effects of noise.

The columns of $V_k^\perp$ may be viewed as a number of ('virtual') FIR channels that produce a zero-output when fed with a specific segment of the symbol sequence:

$$X_{k-L|k+i-1} \cdot V_k^\perp = 0. \qquad (3.3)$$

We now focus on the recovery of one row of $X_{k-L|k+i-1}$: $\left[\begin{array}{ccc} x(k) & \cdots & x(k+j-1) \end{array}\right]$. First note that

$$\begin{aligned} X_{k-(L+i)+1|k} \cdot V_{k-i+1}^{\perp} &= 0 \\ \vdots & \quad \vdots \\ X_{k|k+(L+i)-1} \cdot V_{k+L}^{\perp} &= 0. \end{aligned}$$

Since $\left[\begin{array}{ccc} x(k) & \cdots & x(k+j-1) \end{array}\right]$ forms a row of each of the above input matrices, it is orthogonal to $V_{k-i+1}^{\perp}, \ldots, V_{k+L}^{\perp}$. Therefore, after rearranging the indices, one obtains an equivalent equation:

$$W_k \cdot \mathbf{x}(k : k+j-1) = 0 \tag{3.4}$$

$$W_k \stackrel{\triangle}{=} \left[\begin{array}{ccccccc} V_{k-i+1}^{\perp} & \cdots & V_k^{\perp} & \cdots & V_{k+L}^{\perp} \end{array}\right]^T. \tag{3.5}$$

Here W_k is a $p \times j$ matrix with $p \stackrel{\triangle}{=} (i+L) \cdot (j-(i+L))$. $\mathbf{x}$ is a length N column vector containing all unknown symbols, see equation 2.10, and $\mathbf{x}(k : k+j-1)$ then represents a length j segment of the vector $\mathbf{x}$. The matrix W_k thus contains $(L+i)$ null spaces computed at subsequent time instants and provides a homogeneous matrix equation in a segment of the transmitted symbol sequence. Equation 3.4 was first derived in [139] and also found independently in [56]. The following theorem ensures unique identifiability of $\mathbf{x}(k : k+j-1)$ from equation 3.4.

Theorem 3.1.1 *If the matrices* $X_{k-L-i+1|k+1}, \cdots, X_{k-L|k+i}, \cdots,$ $X_{k-1|k+i+L-1}$ *are of full row rank (which implies* $j > (i+L)$*) then* $\mathbf{x}(k : k+j-1)$ *will be uniquely identifiable from equation 3.4.*

For the proof presented below we make use of the dimension law of vector spaces.

Property 3.1.1 *Given two subspaces* $\mathcal{A}$ *and* $\mathcal{B} \in \mathbb{C}^n$ *then:*

$$dim(\mathcal{A}) + Dim(\mathcal{B}) = dim(\mathcal{A} \cup \mathcal{B}) + dim(\mathcal{A} \cap \mathcal{B}). \tag{3.6}$$

Proof: Since we assume that $X_{k-L-i+1|k+1}$ is of full row rank, $X_{k-L-i+1|k}$ certainly is and the space spanned by the rows of $X_{k-L-i+1|k}$ is the left null space of $V_{k-i+1}^{\perp}$. The same relationship holds for $X_{k-L-i+2|k+1}$ and $V_{k-i+2}^{\perp}$. The left null space of $[V_{k-i+1}^{\perp} V_{k-i+2}^{\perp}]$ is thus formed by an intersection of the row spaces of $X_{k-L-i+1|k}$ and $X_{k-L-i+2|k+1}$. From the dimension law of vector spaces (property 3.1.1), it follows that the intersection has dimension $L+i-1$.

Algorithm 3.1
Subspace decomposition

1. *initialization:*

$$W_1 \leftarrow [\quad]$$

for $k = 2 - i, \ldots, L + 1$

$$Y_{k|k+i-1} \leftarrow \begin{bmatrix} U_k^s & U_k^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma_k^s & 0 \\ 0 & \Sigma_k^\perp \end{bmatrix} \cdot \begin{bmatrix} V_k^{sH} \\ V_k^{\perp H} \end{bmatrix}$$

$$W_1 \leftarrow \begin{bmatrix} W_1 \\ V_k^{\perp T} \end{bmatrix}$$

end

2. *iteration:*
for $k = L + 2, \ldots, N - j + 1 + L$

$$Y_{k|k+i-1} \leftarrow \begin{bmatrix} U_k^s & U_k^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma_k^s & 0 \\ 0 & \Sigma_k^\perp \end{bmatrix} \cdot \begin{bmatrix} V_k^{sH} \\ V_k^{\perp H} \end{bmatrix}$$

$$W_{k-L} \leftarrow \begin{bmatrix} W_{k-L-1}(j-(i+L)+1 : (i+L)(j-(i+L)), :) \\ V_k^{\perp T} \end{bmatrix}$$

end

The rows of $X_{k-L-i+2|k}$ certainly belong to this intersection (they appear in $X_{k-L-i+1|k}$ and $X_{k-L-i+2|k+1}$) and since the rank of $X_{k-L-i+2|k}$ is $L + i - 1$ it spans the left null space of $[V_{k-i+1}^\perp V_{k-i+2}^\perp]$. If we proceed in this way, the right null space of W_k has dimension 1 and $\mathbf{x}(k : k + j - 1)$ forms the unique solution to equation 3.4 (up to an unknown constant). □

Remark 3.1.2 In chapter 6, it will be shown how Theorem 3.1.1 can be extended to the multi-user scenario.

The adaptive algorithms presented further on are fairly robust against a (temporary) violation of the rank conditions of Theorem 3.1.1. Each symbol is part of equation 3.4 during j consecutive time steps and will be estimated in j consecutive iteration steps (see also Theorems 3.2.3 and 4.2.1).

The subspace decomposition step is summarized in algorithm 3.1. In each iteration, a new W_k is formed by removing the first block row of W_{k-1} and appending a new block row. A total of $(N - j + L + i)$ singular value decompositions need to be computed, each of a $t \times j$ matrix, with $t \triangleq M \cdot i$. Using

the standard SVD algorithm of [33], this leads to the following computational complexity [1] :

$$(N - j + L + i) \cdot (16 \cdot j^2 \cdot t + 12 \cdot j \cdot t^2).$$

Note that the matrix $Y_{k|k+i-1}$ undergoes a rank-two modification between two iteration steps. In a down-dating step, the first column is removed, while in the updating step a new column is added at the end. This opens perspectives for adaptive SVD techniques [16, 15, 95, 114, 10] which compute (an approximation of) the SVD by up/downdating the (approximate) SVD computed in the previous time step. A modified version of the algorithm [95], computes an approximate SVD at each time step and is based on a QR-up/down-dating step [53, pp 592-597] and Jacobi iterations. The approximation error can be controlled by varying the number of Jacobi iterations, see also [95]. The algorithm has the following computational complexity, see also [148]:

$$(N - j + L + i) \cdot (18 \cdot j^2 + (29 + 36 \cdot it) \cdot t^2 + (24 + 18 \cdot it) \cdot j \cdot t)$$

where it is the number of iterations in the Jacobi step of the adaptive algorithm. Note that for the adaptive algorithm only second-order terms appear in the formula for the computational complexity. However since j and $t = M \cdot i$ are both fairly small (typically smaller than 15), the computational savings with respect to a straightforward SVD calculation are not large.

Remark 3.1.3 With noise, only an approximate null space $\widehat{W}_k$ is computed:

$$\widehat{W}_k \cdot \hat{\mathbf{x}}(k : k + j - 1) = 0 \tag{3.7}$$

with $\hat{\mathbf{x}}(k : k+j-1)$ an estimate of the exact solution $\mathbf{x}(k : k+j-1)$. If we solve this equation in least squares sense[2], the solution will be statistically optimal only if the residuals are uncorrelated and of equal variance. This requires that a weighting factor is applied to equation 3.7.

This weighting factor is a function of the statistical characterization of $\widehat{W}_k$ in terms of W_k and the additive noise $N_{k|k+i-1}$. It can be obtained via perturbation expansion theory [113, 77], but is complicated and is further hampered by the correlation between the noise samples in $N_{k|k+i-1}$ ($N_{k|k+i-1}$ has block Hankel structure). Some of the algorithms developed in the next sections include a quantization step as part of a decision feedback mechanism, i.e. the results are mapped to the closest (Euclidean distance) discrete symbol value. The performance of the algorithms is then primarily determined by the feedback of (in)correct symbol decisions (rather than by the (sub)optimal weighting).

[1] Here both multiplications and additions are counted. Only third order terms were retained.

[2] Note that we still have to apply a non-triviality constraint to avoid the trivial null solution.

Therefore equation 3.4 will be solved in a straightforward (but statistically suboptimal) way. For simplicity of notation we do not make the explicit distinction between W_k and $\widehat{W}_k$ further on, it should be clear from the context whether the exact null space or the noisy estimate is meant.

At iteration k, a new W_k is computed which provides a set of homogeneous equations in a corresponding data segment $\mathbf{x}(k : k+j-1)$, equation 3.4. At that moment, all previous equations in W_1, W_2,..., W_{k-1} together with the new equations in W_k may be assembled in one matrix $\mathcal{W}_k$ to form one large overdetermined set of homogeneous equations in $\mathbf{x}(1 : k+j-1)$:

$$\mathcal{W}_k \triangleq \begin{bmatrix} \boxed{W_1} & 0 & 0 & \dots \\ 0 & \boxed{W_2} & 0 & \dots \\ & & \ddots & \\ 0 & 0 & \dots & \boxed{W_k} \end{bmatrix} \tag{3.8}$$

$$\mathcal{W}_k \cdot \mathbf{x}(1 : k+j-1) = 0, \tag{3.9}$$

or with noise:

$$\min_{\hat{\mathbf{x}}(1:k+j-1)} \|\mathcal{W}_k \cdot \hat{\mathbf{x}}(1 : k+j-1)\|. \tag{3.10}$$

This last equation gives an (approximate) relationship between the (desired) inputs and a 'virtual' channel. In order to recover the input sequence we have to apply a *non-triviality constraint*, which then also automatically allows to take the noise into account.

Depending on the non-triviality constraint that is chosen, a different algorithm results.

1. If we prefer a finite alphabet constraint, the solution of equation 3.10 naturally leads to a Viterbi algorithm [39] as explained in section 3.2. The algorithm has a high performance, but its complexity is exponential in the window length j. Therefore, it is interesting to look at computationally cheaper alternatives.

2. If we choose a monic constraint, we obtain a Kalman filter type of solution, see section 3.3.

3. Finally, in chapter 4 , two algorithms are presented which apply a quadratic non-triviality constraint. This leads to different forms of a computationally efficient recursive total least squares (RTLS) algorithm.

Remark 3.1.4 The Viterbi algorithm explicitly imposes a finite alphabet constraint on the solution, while the Kalman filter and RTLS algorithms first estimate the parameters, followed by a subsequent mapping on the constellation symbols and are in this sense suboptimal.

3.2 Viterbi algorithm

The Viterbi algorithm is a fundamental building block in a plethora of applications in the domains of digital communications and signal processing. Originally proposed as a method to decode convolutional codes [156], it is also used to detect signals in communication channels with memory [104] and in speech and character recognition tasks where the speech signals or characters are modeled by hidden Markov models [27]. The use of the Viterbi algorithm in receivers of ISI corrupted channels dates back to the work of Forney [38, 39] where it was shown that the Viterbi algorithm preceded by a symbol rate sampled whitened matched filter forms the maximum likelihood sequence estimator of a digital sequence in the presence of inter symbol interference.

In the context of blind equalization the Viterbi algorithm has been proposed in [44] for an iterative joint channel and data estimation algorithm. The Viterbi algorithm estimates the data for a fixed channel estimate, alternated with a least squares channel estimation for a fixed data estimate. In [66], a channel model with quantized parameters is developed, and a separate channel and data trellis are used to search for the maximum likelihood channel and data estimates using the Viterbi algorithm. Other variants of blind trellis search techniques can be found in [60, Ch. 6].

Here we explain how the Viterbi algorithm allows to solve equation 3.10 under a finite alphabet constraint. Section 3.2.1 studies the application of the Viterbi algorithm to the problem at hand. Section 3.2.2 discusses the computational complexity of the algorithm.

3.2.1 Viterbi algorithm

Without noise we have:

$$\mathcal{W}_{N-j+1} \cdot \mathbf{x} = 0,$$

with $\mathcal{W}_{N-j+1}$ as defined in equation 3.8. With noise, this relationship will no longer hold exactly. The question then rises which symbol sequence $\hat{\mathbf{x}}$ minimizes:

$$\min_{\hat{\mathbf{x}} \in \Omega^{N \times 1}} \|\mathcal{W}_{N-j+1} \cdot \hat{\mathbf{x}}\|_2^2 \tag{3.11}$$

with Ω the finite alphabet to which the symbols belong. One can minimize equation 3.11 by enumeration, i.e. by computing the norm for each of the η^N possible symbol sequences, with $\eta = \#(\Omega)$. The symbol sequence $\hat{\mathbf{x}}$ minimizing the norm then forms the estimated symbol sequence. Clearly, this method is impractical since its complexity grows exponentially in the length of the data sequence N.

The Viterbi algorithm provides a way to compute the solution to equation 3.11 recursively. The complexity of the Viterbi algorithm is exponential in j but linear in N.

Note that equation 3.11 is separable, i.e.

$$\min_{\hat{\mathbf{x}} \in \Omega^{N \times 1}} \|\mathcal{W}_{N-j+1} \cdot \hat{\mathbf{x}}\|_2^2 = \min_{\hat{\mathbf{x}} \in \Omega^{N \times 1}} \sum_{k=1}^{N-j+1} \|W_k \cdot \hat{\mathbf{x}}(k : k+j-1)\|_2^2 . \qquad (3.12)$$

The presentation of the Viterbi algorithm will be accompanied by an example with downscaled dimensions: $j = 4$ and for a BPSK modulation format, i.e. $\Omega = \{-1, 1\}$, $\eta = 2$. The extension to larger j and other finite alphabets is straightforward. The Viterbi algorithm enumerates over several possible solutions for the symbol sequence $\mathbf{x}$ before choosing the $\hat{\mathbf{x}}$ that minimizes the cost function of 3.12. In the next paragraphs we will denote the symbol sequences which are checked by the Viterbi algorithm by $\mathbf{v}$.

First construct a *trellis*, see figure 3.1. Each *node* in the trellis corresponds to a distinct state at a given time. The state $\mathbf{s_k^l}$ represents the lth state at time k and equals $\mathbf{v}(k : k+j-2)$, i.e. each state contains $j-1$ parameters. E.g. referring to figure 3.1, state 4 at time instant 3 represents $\mathbf{s_3^4} = \mathbf{v}(3:5) = \begin{bmatrix} -1 & 1 & 1 \end{bmatrix}^T$. The number of states at each time instant equals $\eta^{j-1} = 8$, i.e. the trellis of figure 3.1 has 8 rows. Each *branch* in the trellis represents a transition to some new state at the next time instant. A transition between states l and p is possible if and only if

$$\mathbf{s_k^l}(2 : j-1) = \mathbf{s_{k+1}^p}(1 : j-2).$$

Each state has η branches entering the node and η branches leaving the node. We denote the set of branches entering a state p as $\mathcal{S}_p$. There is a one-to-one correspondence between each possible symbol sequence $\mathbf{v}$ and each path through the trellis. Further more, to every possible state sequence $\mathbf{s_1^l}, \cdots, \mathbf{s_{N-j+2}^p}$, corresponds a unique path through the trellis, and vice versa. Hence estimating the most probable symbol sequence is equivalent to estimating the most probable state sequence/path through the trellis.

To compute the most likely state sequence we introduce the concepts *branch metric* and *path metric*. The *path metric* represents the cost associated with each path through the trellis. To compute path metrics we associate a *branch*

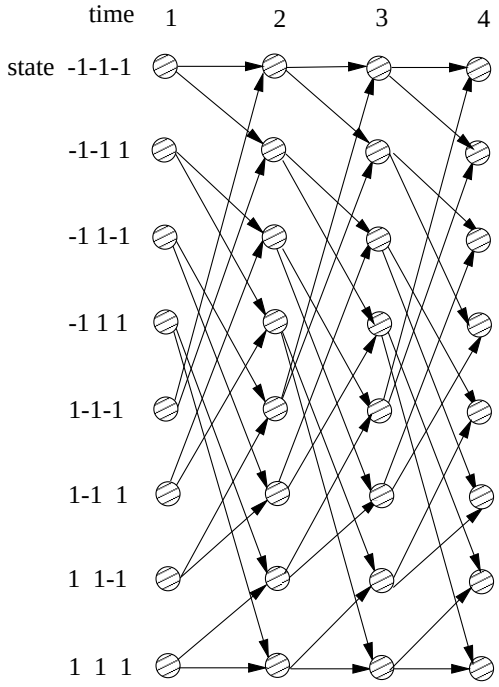


Figure 3.1: A trellis for $j = 4$ and BPSK modulation ($\eta = 2$). The trellis contains η^{j-1}=8 states (the number of rows) and extends over 4 time steps (the number of columns). The arrows, also called branches, denote possible state transitions.

metric to each branch, i.e. to each state transition. The path metric then equals the sum of all branch metrics belonging to the path. The branch metric $C_k^{l,p}$ for a transition from state l at time k to state p at time $k+1$ is given by equation 3.12:

$$\begin{aligned} C_k^{l,p} &= \|W_k \cdot \mathbf{v}(k : k+j-1)\|_2^2 \\ &= \left\| W_k \cdot \begin{bmatrix} \mathbf{s_k^l} \\ \mathbf{s_{k+1}^p}(j-1) \end{bmatrix} \right\|_2^2 . \end{aligned}$$

In general, there will be several paths or state sequences through the trellis which end in a state p at time k, each with their own path metric. The path with the minimum path metric is called the *survivor path* $\mathbf{p_k^p}$. For any time instant k there are η^{j-1} survivor paths, one for each state. *The critical observation is now that the complete path through the trellis with the minimum path metric will have to start with one of the survivors.* If it would not, and passed through state p at time k, then we could replace its initial segment by $\mathbf{p_k^p}$ and get a lower

Algorithm 3.2.1
Viterbi algorithm

1. *initialization:*
   for $l = 1 : \eta^{j-1}$

$$\begin{aligned} M_1^l &\leftarrow 0 \\ \mathbf{p_1^l} &\leftarrow \mathbf{s_1^l} \end{aligned}$$

   end

2. *iteration equations:*
   for $k = 1, \ldots, N-j+1$
   - *compute path metrics:*
     for $p = 1, \ldots, \eta^{j-1}$

$$\begin{aligned} M_{k+1}^p &\leftarrow \min_{l \in \mathcal{S}_p} M_k^l + C_k^{l,p} \\ &\leftarrow M_k^{l_{min}} + C_k^{l_{min},p} \\ \mathbf{p_{k+1}^p} &\leftarrow \left[\begin{array}{cc} \mathbf{p_k^{l_{min}}} & \mathbf{s_{k+1}^p}(j-1) \end{array} \right] \end{aligned}$$

     end
   end

3. *sequence estimation:*

$$\begin{aligned} M_{N-j+2}^{l_{min}} &\leftarrow \min_{l=1,\ldots,\eta^{j-1}} M_{N-j+2}^l \\ \hat{\mathbf{x}} &\leftarrow (\mathbf{p_{N-j+2}^{l_{min}}})^T \end{aligned}$$

path metric. Thus at time k, we need to remember only the η^{j-1} survivor paths and their path metrics M_k^p. To get to time $k+1$, we need only extend all time-k survivors by one time unit, compute the branch metrics of the extended path segments and for each state $\mathbf{s_{k+1}^p}$ select the shortest extended path segment terminating in $\mathbf{s_{k+1}^p}$ as the corresponding survivor at time $k+1$ for state p:

$$\begin{aligned} M_{k+1}^p &= \min_{l \in \mathcal{S}_p} M_k^l + C_k^{l,p} \\ &= M_k^{l_{min}} + C_k^{l_{min},p} \\ \mathbf{p_{k+1}^p} &= \left[\begin{array}{cc} \mathbf{p_k^{l_{min}}} & \mathbf{s_{k+1}^p}(j-1) \end{array} \right], \end{aligned}$$

with $\mathcal{S}_p$ the set of branches entering the node p as defined previously. Finally, the estimated symbol sequence $\hat{\mathbf{x}}$ equals the survivor path corresponding to the lowest path metric at the end of the trellis, i.e. at time step $N-j+2$. The Viterbi algorithm is formally summarized as algorithm 3.2.1.

Remark 3.2.1 It is obvious that both steps of the algorithm, i.e. the subspace decomposition (algorithm 3.1) and the Viterbi algorithm (algorithm 3.2.1) do

not have to be executed one after the other but can be intertwined into one algorithm.

Remark 3.2.2 The algorithm 3.2.1 does not exploit all information contained in the convolutional tails at the beginning and end of the bursts (assuming a guard band between subsequent bursts). Although notation is somewhat cumbersome, the modifications necessary to include this information are straightforward. The same remark applies for all algorithms developed in this thesis.

As already mentioned, the identifiability conditions imposed by Theorem 3.1.1 may be too stringent for the problem at hand. The next theorem clarifies the robustness of the Viterbi algorithm against a violation of these rank conditions.

Theorem 3.2.3 *Assume that at the beginning of iteration step k, there is only one state l for which $M_k^l = 0$ (only one state has a zero path metric, no measurement noise), i.e. the symbol sequence segment $\mathbf{x}(1 : k + j - 2)$ can be determined uniquely ($k \geq 1$). If the rank conditions of Theorem 3.1.1 are violated less than $j - 1$ consecutive iteration steps, then the input symbol stream can still be recovered correctly by the Viterbi algorithm.*

Proof: If the rank conditions of Theorem 3.1.1 are not satisfied at iteration step k, there will be more than one solution to $W_k \cdot \mathbf{v}(k : k + j - 1) = 0$. The exact solution $\mathbf{x}(k : k + j - 1)$ however, still belongs to the solution space. Assume a worst case scenario, i.e. when the rank conditions of Theorem 3.1.1 are not satisfied, all branch metrics are zero. If the rank conditions of Theorem 3.1.1 are violated during l subsequent iteration steps, then the rank conditions hold again at iteration step $k+l$, and the 'optimal' path (the path corresponding to $\mathbf{x}(1 : k + l + j - 2)$) still has a zero path metric. At iteration $k + l$, equation 3.4 again has a unique solution (up to a scaling factor) for the symbols $k + l$ up to $k + l + j - 1$. The scaling factor ambiguity can be removed by observing that l is at most $j - 2$. Consequently, at least one of the symbols estimated at iteration $k + l + 1$ has already been estimated before iteration k (where there was a unique solution), which resolves the scaling ambiguity. □

In the next section we discuss some issues related to the computational complexity of algorithm 3.2.1.

3.2.2 Further discussion

The Viterbi algorithm as presented above computes η^j branch metrics during $N - j + 1$ time steps. The computation of each branch metric involves $j \cdot p$

complex multiply-accumulations (W_k is a $p \times j$ matrix) and the computation of the two-norm of a vector of length p. The total computational complexity thus amounts to $(8 \cdot j \cdot p + 4 \cdot p) \cdot \eta^j \cdot (N - j + 1)$ flops, which is exponential in j but linear in N. In the following remarks we point out a number of different alternatives to reduce the computational complexity of the Viterbi algorithm.

Remark 3.2.4 The cost function of equation 3.12:

$$\min_{\hat{\mathbf{x}} \in \Omega^{N \times 1}} \sum_{k=1}^{N-j+1} \| W_k \cdot \hat{\mathbf{x}}(k : k+j-1) \|_2^2$$

can also be replaced by:

$$\max_{\hat{\mathbf{x}} \in \Omega^{N \times 1}} \sum_{k=1}^{N-j+1} \| \underbrace{\begin{bmatrix} V_{k-i+1}^{sH} \\ \vdots \\ V_k^{sH} \\ \vdots \\ V_{k+L}^{sH} \end{bmatrix}}_{Z_k} \cdot \hat{\mathbf{x}}(k : k+j-1) \|_2^2$$

i.e. without noise the symbol sequence segment $\mathbf{x}(k : k+j-1)$ lies in the joint row space of V_{k-i+1}^{sH} to V_{k+L}^{sH}, see also [133]. Without noise $\| Z_k \cdot \mathbf{x}(k : k+j-1) \|_2^2$ equals $j \cdot (i+L)$. The maximization of this criterion using the Viterbi algorithm will provide the same performance as the algorithm using the noise subspaces. It can be more attractive from a computational point of view when the dimension of the noise subspace is larger than that of the signal subspace. The computation of an (approximate) signal subspace can be done via orthogonal iteration [53].

Remark 3.2.5 W_k is a $p \times j$ matrix of rank $j-1$ built from $(i+L)$ null spaces, with $p = (i+L) \cdot (j - (i+L))$, and usually $p \gg j - 1$. It is possible to reduce the complexity of the Viterbi algorithm by including less null spaces in W_k. Note that

$$\begin{aligned} X_{k-(L+i)+1|k} \cdot V_{k-i+1}^{\perp} &= 0 \\ X_{k|k+(L+i)-1} \cdot V_{k+L}^{\perp} &= 0. \end{aligned}$$

If the matrix $X_{k-(L+i)+1|k+i+L-1}$ has full row rank $2(i+L)-1$, then via the dimension law of vector spaces (equation 3.6) the row space intersection of $X_{k-(L+i)+1|k}$ and $X_{k|k+i+L-1}$ is one-dimensional (proof analog to that of Theorem 3.1.1) and the equation $\begin{bmatrix} V_{k-i+1}^{\perp} & V_{k+L}^{\perp} \end{bmatrix}^T \cdot \mathbf{x}(k : k+j-1) = 0$ uniquely determines $\mathbf{x}(k : k+j-1)$. The vector $\mathbf{x}(k : k+j-1)$ is now determined from two null spaces where in the original problem formulation (equation 3.4) it

was estimated using $(i+L)$ null spaces. Reducing the complexity of the Viterbi algorithm in this way will of course have its penalty on the noise robustness of the algorithm.

Remark 3.2.6 In literature several approaches have been proposed to reduce the complexity of the Viterbi algorithm. In [37], preprocessing techniques are developed to shorten the channel impulse response length. A linear prefilter is computed to force the overall impulse response of the channel + filter combination to approximate a desired truncated impulse response of acceptably short duration. In [36] decision feedback equalization is combined with set partitioning principles [104, Ch. 5.4] to obtain a reduced state trellis.

Remark 3.2.7 Although the computational complexity of the Viterbi algorithm is high (even when the above changes are included), its performance can serve as a reference point for the computationally cheaper algorithms that will be developed in the next section/chapter.

In chapter 5 the performance of the subspace + Viterbi algorithm will be analyzed. In chapter 6 it is shown how the algorithm can be generalized to the multi-user scenario, i.e. when multiple users are transmitting digital symbol sequences at the same time, in the same frequency band.

3.3 Kalman filter algorithm

The second algorithm applies a monic non-triviality constraint when solving equation 3.4 and is based on a Kalman filter algorithm. In the last decades Kalman filtering [65] has become a fundamental tool in signal processing, control and communications. Early applications of the Kalman filter in an equalization context can be found in [71, 72]. The Kalman filter estimates the states of linear dynamic systems from noisy measurements. It uses a least squares criterion and exploits the available inputs, outputs, system matrices and noise covariance matrices, see [59].

Usually Kalman filter equations are developed from a conditional probability theory to obtain linear minimum variance state estimates. It has however been shown in [34, 101] that Kalman filter state estimates can equally well be computed by solving a set of weighted linear least squares problems. We will adhere to this last approach when presenting the algorithm.

In section 3.3.1 the data model is rewritten in state space form. Next, in section 3.3.2 a square root form of the Kalman filter algorithm is presented using an approach similar to [34, 101].

3.3.1 State space data model

First we embed equation 3.4 (which is repeated here),

$$W_k \cdot \mathbf{x}(k : k+j-1) = 0 \tag{3.13}$$

in a state space model. The number of states equals $j-1$, and the state vector $\mathbf{s_k}$ is defined as:

$$\mathbf{s_k} = \mathbf{x}(k : k+j-2).$$

We now introduce a monic non-triviality constraint: at time k we assume to have a reliable estimate of $x(k-1)$: $\hat{x}(k-1)$. The state and output equation at iteration step k read:

$$\mathbf{s_{k+1}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & & 1 \\ 0 & \cdots & & & 0 \end{bmatrix}}_{A} \cdot \mathbf{s_k} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \cdot \mathbf{x}(k+j-1)}_{\mathbf{q_k}} \tag{3.14}$$

$$\underbrace{-W_{k-1}(:,1) \cdot \hat{x}(k-1)}_{\mathbf{z_k}} = \underbrace{W_{k-1}(:,2:j)}_{C_k} \cdot \mathbf{s_k} + \mathbf{r_k} \tag{3.15}$$

We adopted a *stochastic* state space model where the unknown input is treated as a random white noise term $\mathbf{q_k}$, the vector $\mathbf{r_k}$ represents 'measurement noise'.

Remark 3.3.1 Note that the above model only applies for $k \geq 2$. Without loss of generality, we assumed $x(1) = 1$, the unknown phase factor can be recovered through differential encoding (see section 2.2).

3.3.2 Kalman filter algorithm

For the derivation of the square-root form of the Kalman filter, we take an approach similar to [101] which will turn out to provide further insight. The idea is to rewrite the Kalman filter as a weighted least squares problem. Rearranging the state and output equations 3.14 and 3.15 yields:

$$\begin{bmatrix} 0 \\ \mathbf{z_2} \\ 0 \\ \mathbf{z_3} \\ 0 \\ \vdots \end{bmatrix} = \begin{bmatrix} I & 0 & \cdots & 0 \\ C_2 & 0 & & \\ A & -I & 0 & \\ 0 & C_3 & & \\ 0 & A & -I & \\ \vdots & & & \vdots \end{bmatrix} \cdot \begin{bmatrix} \mathbf{s_2} \\ \mathbf{s_3} \\ \mathbf{s_4} \\ \vdots \end{bmatrix} + \begin{bmatrix} \mathbf{q_1} \\ \mathbf{r_2} \\ \mathbf{q_2} \\ \mathbf{r_3} \\ \mathbf{q_3} \\ \vdots \end{bmatrix}. \tag{3.16}$$

We assume a zero initial state estimate and $\mathbf{q_1}$ is defined as $\mathbf{q_1} \stackrel{\triangle}{=} -\mathbf{s_2}$.

Exploiting the (known) structure of the matrix A in the above equation, allows to rewrite the state equations as:

$$0 = A \cdot \mathbf{s_k} - \mathbf{s_{k+1}} + \mathbf{q_k} \tag{3.17}$$

$$= \begin{bmatrix} 0_{(j-2)\times 1} \\ -\mathbf{x}(k+j-1) \end{bmatrix} + \begin{bmatrix} 0_{(j-2)\times 1} \\ \mathbf{q_k}(j-1) \end{bmatrix} \tag{3.18}$$

Each state equation thus reduces to one scalar equation. Equation 3.16 then reads:

$$\begin{bmatrix} 0 \\ \mathbf{z_2} \\ 0 \\ \mathbf{z_3} \\ 0 \\ \mathbf{z_4} \\ \vdots \\ 0 \\ \mathbf{z_k} \end{bmatrix} = \begin{bmatrix} I & 0 & \cdots & 0 \\ \boxed{C_2} & 0 & & \\ 0 & -1 & \cdots & 0 \\ 0 & \boxed{C_3} & 0 & \\ 0 & & -1 & \\ 0 & 0 & \boxed{C_4} & \\ \vdots & & \ddots & \\ 0 & & & -1 \\ & & & \boxed{C_k} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}(2) \\ \mathbf{x}(3) \\ \vdots \\ \mathbf{x}(k+j-2) \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{q_1} \\ \mathbf{r_2} \\ \mathbf{q_2}(j-1) \\ \mathbf{r_3} \\ \mathbf{q_3}(j-1) \\ \mathbf{r_4} \\ \vdots \\ \mathbf{q_{k-1}}(j-1) \\ \mathbf{r_k} \end{bmatrix}}_{\mathbf{e_k}} \tag{3.19}$$

We have obtained a least squares problem in the unknown data. For a statistically optimal solution, equation 3.19 should now be prewhitened. However, as explained in remark 3.1.3, the covariance matrix $E\{\mathbf{e_k} \cdot \mathbf{e_k}^H\}$ is hard to estimate. Furthermore the performance of the algorithm will primarily be determined by the feedback of (in)correct symbol estimates in the output $\mathbf{z_k}$.

Therefore we make the following ad hoc approximation of the noise covariance matrix:

$$E\{\mathbf{e_k} \cdot \mathbf{e_k}^H\} \approx \begin{bmatrix} I_{j-1} & & & & \\ & \zeta \cdot I_p & & & \\ & & 1 & & \\ & & & \zeta \cdot I_p & \\ & & & & \ddots \end{bmatrix}.$$

Simulations results show that the choice of ζ has little influence on performance, we choose ζ equal to the noise covariance σ^2. It can be estimated from the singular value spectrum computed during the subspace decomposition (see section 3.1). Next we weight the equations corresponding to the (reduced) state

equations by a factor σ.

$$\underbrace{\begin{bmatrix} 0 \\ \mathbf{z_2} \\ 0 \\ \mathbf{z_3} \\ \vdots \\ 0 \\ \mathbf{z_k} \end{bmatrix}}_{\mathbf{t_k}} = \underbrace{\begin{bmatrix} \sigma \cdot I & 0 & \cdots & 0 \\ \boxed{C_2} & 0 & & \\ 0 & -\sigma & \cdots & 0 \\ 0 & \boxed{C_3} & 0 & \\ \vdots & \ddots & & \\ 0 & & & -\sigma \\ & & & \boxed{C_k} \end{bmatrix}}_{F_k} \begin{bmatrix} \mathbf{x}(2) \\ \mathbf{x}(3) \\ \vdots \\ \mathbf{x}(k+j-2) \end{bmatrix} + \begin{bmatrix} \sigma \cdot \mathbf{q_1} \\ \mathbf{r_2} \\ \sigma \cdot \mathbf{q_2}(j-1) \\ \mathbf{r_3} \\ \vdots \\ \sigma \cdot \mathbf{q_{k-1}}(j-1) \\ \mathbf{r_k} \end{bmatrix} \tag{3.20}$$

Remark 3.3.2 Without noise, σ is zero and the equations according to the reduced state equations can be removed from matrix equation 3.20.

Now we solve equation 3.20 in least squares sense using a recursive QR decomposition alternated with back substitution (see also section 2.8). Assume that at time k, the skinny QR decomposition of F_k has been computed:

$$F_k = Q_k \cdot R_k$$

with Q_k an orthonormal $(p \cdot (k-1) + (k+j-3)) \times (k+j-3)$ matrix and R_k a square upper triangular $(k+j-3) \times (k+j-3)$ matrix, then

$$\underbrace{\begin{bmatrix} \star & \star & \star & \star & \star & \star \\ & \star & \star & \star & \star & \star \\ & & \star & \star & \star & \star \\ & & & \star & \star & \star \\ & & & & \star & \star \\ & & & & & \star \end{bmatrix}}_{R_k = Q_k^H \cdot F_k} \cdot \underbrace{\begin{bmatrix} \mathbf{v_k}(2) \\ \vdots \\ \mathbf{v_k}(k+j-2) \end{bmatrix}}_{\mathbf{v_k}(2:k+j-2)} \overset{LS}{=} \underbrace{\begin{bmatrix} \star \\ \star \\ \star \\ \star \\ \star \\ \star \end{bmatrix}}_{\mathbf{d_k} = Q_k^H \cdot \mathbf{t_k}} \tag{3.21}$$

where the vector $\mathbf{v_k}(2 : k+j-2)$ is the least squares estimate of the exact solution $[\mathbf{x}(2) \cdots \mathbf{x}(k+j-2)]^T$ and $\star$ should be replaced by the appropriate complex entry. To go to time $k+1$, the following problem has to be solved:

$$\begin{bmatrix} \boxed{\begin{matrix} \star & \star & \star & \star & \star & \star \\ & \star & \star & \star & \star & \star \\ & & \star & \star & \star & \star \\ & & & \star & \star & \star \\ & & & & \star & \star \\ & & & & & \star \end{matrix}} & 0 \\ & -\sigma \\ \boxed{C_{k+1}} & \end{bmatrix} \cdot \mathbf{v_{k+1}}(2 : k+j-1) = \begin{bmatrix} \boxed{\begin{matrix} \star \\ \star \\ \star \\ \star \\ \star \\ \star \end{matrix}} \\ 0 \\ \mathbf{z_{k+1}} \end{bmatrix}. \tag{3.22}$$

Note that at iteration $k+1$ a decision has already been made on $\mathbf{v_{k+1}}(2:k)$ (see equation 3.15), i.e.

$$\mathbf{v_{k+1}}(2:k) \approx \left[\begin{array}{ccc} \hat{\mathbf{x}}(2) & \cdots & \hat{\mathbf{x}}(k) \end{array} \right]^T . \tag{3.23}$$

So we are only interested in the solution of the above matrix equation in $\mathbf{v_{k+1}}(k+1:k+j-1)$. We now approximate equation 3.22 by its lower triangular part:

$$\left[\begin{array}{c} \boxed{\begin{array}{ccc} \star & \star & \star \\ & \star & \star \\ & & \star \end{array}} \quad 0 \\ -\sigma \\ \boxed{C_{k+1}} \end{array} \right] \cdot \left[\begin{array}{c} \mathbf{v_{k+1}}(k+1) \\ \vdots \\ \mathbf{v_{k+1}}(k+j-1) \end{array} \right] = \left[\begin{array}{c} \boxed{\begin{array}{c} \star \\ \star \\ \star \end{array}} \\ 0 \\ \mathbf{z_{k+1}} \end{array} \right] . \tag{3.24}$$

This matrix equation can be made triangular by applying unitary matrix transformations. Householder transformations [53, pp 211-212] or Givens rotations [53, pp 214-215] can be used. The transmitted symbols are then estimated through back substitution. Note that after triangularization we obtain $j-1$ equations in $j-1$ unknowns and the problem dimensions are independent of the iteration step k. To conclude, we summarize the above derivation as algorithm 3.3.2. The symbol $\mathcal{D}(\cdot)$ denotes a mapping to the closest constellation symbol, see also equation 2.11[3].

The computational complexity of the Kalman filter algorithm is approximately

$$(N-j+1)\cdot((8\cdot p+11)\cdot j^2+2\cdot p\cdot j+4\cdot j^2)$$

where the first two terms in the summation account for the QR updating and the last term is the complexity of the back substitution.

3.4 Conclusions

In this chapter we have presented a preprocessing step based on an adaptive subspace decomposition (i.e. a subspace decomposition is computed in each iteration step). Combining several decompositions has allowed to construct a set of homogeneous equations in the unknown transmitted symbols. We have applied two different non-triviality constraints to solve this set of equations recursively. A finite alphabet constraint has led to a Viterbi algorithm while a monic constraint has provided a Kalman filter type solution. From the derivations in this chapter we expect the Viterbi algorithm to have a good performance (it explicitly imposes a finite alphabet constraint on the solution) but

[3] When the argument of $\mathcal{D}(\cdot)$ is a vector each vector entry is mapped to the closest constellation symbol.

Algorithm 3.3.2
Kalman algorithm

1. *initialization:*

$$\begin{aligned} \mathbf{d} &\leftarrow 0_{(j-1)\times 1} \\ R &\leftarrow \sigma \cdot I_{(j-1)\times(j-1)} \\ \hat{\mathbf{x}}(1) &\leftarrow 1 \end{aligned}$$

2. *iteration equations:*
   for $k = 2, \ldots, N-j+2$

   - *update* R

$$\begin{aligned} \mathbf{z_k} &\leftarrow -W_{k-1}(:,1) \cdot \hat{\mathbf{x}}(k-1) \\ C_k &\leftarrow W_{k-1}(:, 2:j) \\ \left[\begin{array}{c|c} R & \mathbf{d} \\ \hline 0 & \star \end{array}\right] &\leftarrow Q^H \cdot \left[\begin{array}{c|c} R & \mathbf{d} \\ \hline C_k & \mathbf{z_k} \end{array}\right] \end{aligned}$$

   - *back substitution*

$$R \cdot \mathbf{v} \leftarrow \mathbf{d}$$

   - *make a decision:*

$$\hat{\mathbf{x}}(k) \leftarrow \mathcal{D}(\mathbf{v}(1))$$

   -

$$\begin{aligned} R &\leftarrow \begin{bmatrix} R(2:j-1, 2:j-1) & 0_{(j-2)\times 1} \\ 0_{1\times(j-2)} & -\sigma \end{bmatrix} \\ \mathbf{d} &\leftarrow \begin{bmatrix} \mathbf{d}(2:j-1) \\ 0 \end{bmatrix} \end{aligned}$$

   end

$$\hat{\mathbf{x}}(N-j+3:N) \leftarrow \mathcal{D}(\mathbf{v}(2:j-1))$$

its complexity is exponential in the window length. The Kalman filter is sub-optimal (the transmitted data are not directly estimated in the finite alphabet but the initial estimates are mapped onto the finite alphabet in a second step) but has a lower computational complexity. A detailed comparison of these two algorithms, both in terms of complexity and performance, will be presented in chapter 5. In the next chapter we present yet another solution based on a quadratic non-triviality constraint.

Chapter 4

Recursive total least squares algorithms

The Viterbi algorithm presented in the previous chapter has a high performance, but on the other hand, its complexity is exponential in the window length j. Therefore we introduced a computationally cheaper alternative imposing a monic constraint based on a Kalman filter. In this chapter we apply a unit-norm constraint, which will lead to two computationally efficient recursive total least squares algorithms. In chapter 5 it will be shown that the RTLS algorithms have approximately the same computational complexity as the Kalman filter algorithm but have a better performance (in terms of bit error rate). *The main contribution of this chapter is the exploitation of the specific structure of the set of homogeneous equations (equation 3.8) to develop recursive total least squares algorithms that have constant computational and memory requirements in each iteration step.*

This chapter starts with a short introduction to the total least squares problem in section 4.1. Sections 4.2 and 4.3 present two recursive total least squares algorithms that impose a quadratic non-triviality constraint when solving equation 3.4. Both algorithms consist of two steps. The first step is based on QR updating in the first algorithm, while it is based on inverse updating in the second algorithm. The second step in both algorithms is based on inverse iteration.

In section 4.4 we show how the algorithm based on QR updating can be generalized to solve deconvolution problems starting from a set of non-homogeneous equations. Finally section 4.5 presents some conclusions.

4.1 The total least squares problem

In many applications mathematical problems of the following form are encountered: find a vector $\mathbf{x}$ such that $A \cdot \mathbf{x} \approx \mathbf{b}$ where the *full column rank* data matrix $A \in \mathbb{C}^{m \times n}$, $\mathbf{b} \in \mathbb{C}^{m \times 1}$ and $m \geq n$. This set of equations is overdetermined and when $\mathbf{b}$ does not belong to range(A) there is no exact solution to this problem. A common way to solve the problem is to invoke the least squares solution [53] which minimizes

$$\min_{\mathbf{x} \in \mathbb{C}^{n \times 1}} \|A \cdot \mathbf{x} - \mathbf{b}\|_2 .$$

This can be recast into the following equivalent form:

$$\min_{\substack{\Delta \mathbf{b} \in \mathbb{C}^{m \times 1}, \\ \mathbf{b} + \Delta \mathbf{b} \in \text{range}(A)}} \|\Delta \mathbf{b}\|_2 .$$

The least squares solution is then given by the exact solution of

$$A \cdot \mathbf{x}_{LS} = \mathbf{b} + \Delta \mathbf{b}.$$

Assume that the true parameter vector $\mathbf{x}$ adheres to the data model:

$$A \cdot \mathbf{x} = \mathbf{b} + \Delta \tilde{\mathbf{b}}.$$

When A has rank n and $\Delta \tilde{\mathbf{b}}$ is zero mean and has a covariance matrix $\sigma^2 \cdot I$ (i.e. the errors are uncorrelated with equal variance) then the least squares estimator is the linear unbiased estimator with the smallest variance. This assumes that the errors are confined to the vector $\mathbf{b}$.

When the matrix A also contains errors one can consider the following problem:

$$\min_{\substack{\Delta A \in \mathbb{C}^{m \times n}, \Delta \mathbf{b} \in \mathbb{C}^{m \times 1}, \\ \mathbf{b} + \Delta \mathbf{b} \in \text{range}(A + \Delta A)}} \left\| \begin{bmatrix} \Delta A & \Delta \mathbf{b} \end{bmatrix} \right\|_{\text{F}} . \tag{4.1}$$

This problem is called the total least squares (TLS) problem [53, 134], and for the ΔA, $\Delta \mathbf{b}$ that minimizes the above equation, the total least squares solution $\mathbf{x}_{TLS}$ is given by:

$$\begin{aligned} (A + \Delta A) \cdot \mathbf{x}_{TLS} &= \mathbf{b} + \Delta \mathbf{b} \\ \begin{bmatrix} (A + \Delta A) & \mathbf{b} + \Delta \mathbf{b} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}_{TLS} \\ -1 \end{bmatrix} &= 0. \end{aligned}$$

Assume that the true parameter vector $\mathbf{x}$ adheres to the data model:

$$(A + \Delta \tilde{A}) \cdot \mathbf{x} = \mathbf{b} + \Delta \tilde{\mathbf{b}}.$$

When all rows of $\left[\begin{array}{cc} \Delta\tilde{A} & \Delta\tilde{\mathbf{b}} \end{array} \right]$ are independently and identically distributed with zero mean and covariance $\sigma^2 \cdot I$ (i.e. all errors are uncorrelated with equal variance) then the TLS solution will estimate the true parameter vector $\mathbf{x}$ consistently.

The TLS solution can be computed efficiently via the SVD of $\left[\begin{array}{cc} A & \mathbf{b} \end{array} \right]$. Denote:

$$\begin{aligned} A &= U' \cdot \Sigma' \cdot V'^H \\ \left[\begin{array}{cc} A & \mathbf{b} \end{array} \right] &= U \cdot \Sigma \cdot V^H. \end{aligned}$$

If the nth singular value of Σ' is greater than the $(n+1)$th singular value of Σ: $\sigma'_n > \sigma_{n+1}$, then the total least squares estimate of $\mathbf{x}$ is given by

$$\mathbf{x}_{TLS} = -\frac{V(1:n, n+1)}{V(n+1, n+1)}.$$

Remark 4.1.1 It can be shown that the condition $\sigma'_n > \sigma_{n+1}$ is equivalent to $\sigma_n > \sigma_{n+1}$ and $V(n+1, n+1) \neq 0$, see [134].

Remark 4.1.2 Whenever the set of equations $A \cdot \mathbf{x} = \mathbf{b}$ is highly incompatible σ'_n will be close to σ_{n+1} and the TLS problem will be more ill-conditioned than the full rank LS problem.

The TLS problem allows two other interesting formulations [134]. For a set of homogeneous equations $C \cdot \mathbf{x} \approx 0$ the TLS problem is formulated as:

$$\begin{aligned} \min_{\Delta C \in \mathbb{C}^{m \times n}, \mathbf{x}_{TLS} \in \mathbb{C}^{n \times 1}} & \|\Delta C\|_{\mathrm{F}} \\ \text{with} \quad (C + \Delta C) \cdot \mathbf{x}_{TLS} &= 0 \\ \|\mathbf{x}_{TLS}\|_2 &= 1. \end{aligned} \tag{4.2}$$

The TLS solution is given by the right singular vector corresponding to the smallest singular value of C (note that for a set of homogeneous equations a scaling is not relevant). A third interpretation is given by the orthogonal l_2 approximation problem:

$$\min_{\mathbf{x} \in \mathbb{C}^{n \times 1}, \|\mathbf{x}\|_2 = 1} \|C \cdot \mathbf{x}\|_2 . \tag{4.3}$$

The solution to this problem is given by the eigenvector corresponding to the smallest eigenvalue of $C^H \cdot C$, but this is exactly the right singular vector corresponding to the smallest singular value of C (see also section 2.8) and hence coincides with the TLS solution.

Remark 4.1.3 When the matrix C contains structure (e.g. Hankel or Toeplitz structure) and the same structure is imposed on ΔC, the optimal solution to equation 4.1 and 4.2 is no longer given by the SVD. In this case computationally more involved structured total least squares methods can be applied , see e.g. [25, 26, 74, 73]. However, the solution to the orthogonal l_2 approximation approach of equation 4.3 is still given by the SVD (independent of structure) see [134].

In the next two sections we develop two TLS algorithms which solve the set of homogeneous equations (equation 3.9) recursively. Both algorithms start from the third formulation of the TLS problem (equation 4.3) and are based on respectively QR updating (RTLS-QR algorithm, section 4.2) and inverse updating (RTLS-I algorithm, section 4.3). In section 4.4, it is shown how the RTLS-QR algorithm can be generalized to solve a set of non-homogeneous equations (RTLS-D algorithm). This last algorithm starts from the first formulation of the TLS problem, i.e. from equation 4.1.

4.2 RTLS based on QR updating (RTLS-QR)

If a quadratic non triviality constraint is applied to the solution of equation 3.10 we obtain:

$$\min_{\hat{\mathbf{x}}(1:k+j-1),\ \|\hat{\mathbf{x}}(1:k+j-1)\|_2=1} \|\mathcal{W}_k \cdot \hat{\mathbf{x}}(1:k+j-1)\|_2. \tag{4.4}$$

This problem can be viewed as a total least squares problem, see equation 4.3. The solution is then given by the right singular vector $\mathbf{v_k}$ corresponding to the smallest singular value of the matrix $\mathcal{W}_k$[1]. When there is no noise, $\mathbf{v_k}$ will equal the transmitted symbol sequence $\mathbf{x}(1:k+j-1)$, up to a scaling factor. With noise, $\mathbf{v_k}$ provides an estimate of $\mathbf{x}(1:k+j-1)$ which minimizes equation 4.4. Analog to the Viterbi algorithm (Theorem 3.2.3), we can also relax the rank condition of Theorem 3.1.1 for the (R)TLS algorithm as is shown in the next theorem:

Theorem 4.2.1 *Assume that the rank conditions of Theorem 3.1.1 hold up to iteration k ($k \geq 1$), and hence that $\mathcal{W}_k$ has a one-dimensional null space (noiseless case). If the rank conditions are violated in l consecutive iteration steps, with $l < j-1$, then $\mathcal{W}_{k+l+1}$ will still have a one-dimensional null space.*

[1] Note that the singular vector $\mathbf{v_k}$ is different from the least squares solution $\mathbf{v_k}$ used in the Kalman filter algorithm of the previous chapter.

Proof: Assume a worst case scenario: if the rank conditions of Theorem 3.1.1 are not satisfied at iteration $k+r$ then the null space of W_{k+r} is j-dimensional, i.e. W_{k+r} is a zero matrix. If the rank conditions hold again at time $k+l+1$ then W_{k+l+1} has a unique solution in the data segment $\mathbf{x}(k+l+1:k+l+j)$ (up to a scaling factor). Now $\mathcal{W}_k$ had a unique solution in the data segment $\mathbf{x}(1:k+j-1)$. Since l is at most $j-2$, the scaling ambiguity can be resolved and the null space of

$$\begin{bmatrix} \mathcal{W}_k & 0 \ \ldots \\ 0 \ \ldots & \boxed{W_{k+l+1}} \end{bmatrix}$$

will be one-dimensional. □

We could now construct the matrix $\mathcal{W}_{N-j+1}$ and obtain an estimate of all unknown symbols $\mathbf{x}$ by computing the right singular vector corresponding to the smallest singular value of $\mathcal{W}_{N-j+1}$. However, we take a different approach here and compute the right singular vector recursively, i.e. for the computation of the right singular vector at iteration k we use the results obtained at iteration step $k-1$. We do this by *exploiting the sparsity of* $\mathcal{W}_k$. In this section we present a recursive solution based on QR updating combined with inverse iteration. In the next section we present a second algorithm based on inverse updating combined with inverse iteration.

The presentation of the algorithm consists of two parts. First we will present an algorithm with growing matrix dimensions (which computes an estimate $\hat{\mathbf{x}}(1:k+j-1)$ at iteration k), from which we will then derive a modified algorithm with computational requirements that do not grow in time (which computes an estimate $\hat{\mathbf{x}}(k+j-b:k+j-1)$ at iteration k, with b a design parameter, as will be explained).

4.2.1 RTLS-QR: version 1

The singular vector that forms the solution to the above mentioned TLS problem is updated in two steps. The *first step* is a QR updating operation in which the triangular R-factor of the QR decomposition of the matrix $\mathcal{W}_k$ is tracked [53]. This first step can be considered as a data compression step, as will be explained in the next section. A similar algorithm in which the R^{-1}-factor is tracked, is presented in section 4.3.

In the *second step* the singular vector is computed through inverse iteration [53] using the R-factor computed in the first step.

In the derivation of the algorithm we exploit the fact that in the noiseless case, $\mathbf{v_k}$ should equal $\mathbf{x}(1:k+j-1)$ and hence its entries should belong to a finite

alphabet Ω.

1. QR UPDATING

Denote the 'skinny' QR decomposition of $\mathcal{W}_{k-1}$ as:

$$\mathcal{W}_{k-1} = Q_{k-1} \cdot R_{k-1} \tag{4.5}$$

where Q_{k-1} is a $p \cdot (k-1) \times (k+j-2)$ orthonormal matrix and R_{k-1} is a square upper triangular $(k+j-2) \times (k+j-2)$ matrix. Obviously, the right singular vectors of $\mathcal{W}_{k-1}$ equal the right singular vectors of R_{k-1}. In this section we show how R_{k-1} can be updated to R_k without requiring the knowledge of Q_{k-1}. Instead of storing the large matrix $\mathcal{W}_{k-1}$ (or equivalently Q_{k-1} and R_{k-1}) , it then becomes possible to work with R_{k-1} only, which will have smaller dimensions. The QR decomposition can thus be considered as a data compression step.

Assume that R_{k-1} has been computed in iteration step $k-1$ and denote by $\mathbf{q}$ any unit-length vector in the null space of Q_{k-1}. Then we can compute R_k in the following way:

$$\begin{aligned}
\mathcal{W}_k &= \left[\begin{array}{c|c} \mathcal{W}_{k-1} & 0_{p\cdot(k-1)\times 1} \\ \hline 0_{p\times(k-1)} & \boxed{W_k} \end{array}\right] \\
&= \left[\begin{array}{c|c} \begin{bmatrix} Q_{k-1} & \mathbf{q} \end{bmatrix} \cdot \begin{bmatrix} R_{k-1} & 0 \\ 0 & 0 \end{bmatrix} & \\ \hline 0_{p\times(k-1)} & \boxed{W_k} \end{array}\right] \\
&= \left[\begin{array}{cc|c} Q_{k-1} & \mathbf{q} & 0 \\ 0 & 0 & I_p \end{array}\right] \cdot \left[\begin{array}{c|c} \begin{matrix} R_{k-1} \\ 0_{1\times(k+j-2)} \end{matrix} & \begin{matrix} 0_{(k+j-2)\times 1} \\ 0 \end{matrix} \\ \hline 0_{p\times(k-1)} & \boxed{W_k} \end{array}\right] \\
&= \underbrace{\left[\begin{array}{cc|c} Q_{k-1} & \mathbf{q} & 0 \\ 0 & 0 & I_p \end{array}\right] \cdot Q}_{\left[\begin{array}{c|c} Q_k & \star \end{array}\right]} \underbrace{Q^H \cdot \left[\begin{array}{c|c} \begin{matrix} R_{k-1} \\ 0_{1\times(k+j-2)} \end{matrix} & \begin{matrix} 0_{(k+j-2)\times 1} \\ 0 \end{matrix} \\ \hline 0_{p\times(k-1)} & \boxed{W_k} \end{array}\right]}_{\left[\begin{array}{c} R_k \\ \hline 0 \end{array}\right]}.
\end{aligned} \tag{4.6}$$

The matrix R_{k-1} is updated to R_k using a standard QR updating procedure [53]. The matrix Q^H is a product of unitary matrix transformations which cancel out the bottom part in the rhs compound matrix. Householder trans-

formations [53, pp 211-212] or Givens rotations [53, pp 214-215] can be used. In this way, we have updated R_{k-1} to R_k without using Q_{k-1} (and $\mathbf{q}$).

Remark 4.2.2 The first $k-1$ rows of R_k are identical to the first $k-1$ rows of R_{k-1} (except for the extra zero elements in the last column of R_k) due to the zero matrix $0_{p\times(k-1)}$ which appears in the appended rows (equation 4.6). This property will be exploited when the algorithm with constant computational requirements is derived (section 4.2.2).

2. Inverse Iteration

The right singular vector $\mathbf{v_k}$, corresponding to the smallest singular value of R_k, is now computed using R_k in an inverse iteration procedure [53, p 383]. If a vector $\mathbf{v}$ is multiplied repeatedly by the inverse of a Hermitian matrix A, then it will converge to the eigenvector associated with the smallest eigenvalue of A:

- initialize $\mathbf{v}$
- for $r = 1, \ldots, P$

$$\mathbf{t} = A^{-1} \cdot \mathbf{v} \tag{4.7}$$

$$\mathbf{v} = \frac{\mathbf{t}}{\|\mathbf{t}\|_F} \tag{4.8}$$

  end

where P denotes the number of iterations in the inverse iteration procedure. The second equation in the loop is a (fairly arbitrary) rescaling. Since the right singular vectors of R_k equal the eigenvectors of $(R_k^H \cdot R_k)$, the right singular vector $\mathbf{v_k}$ can be computed (up to a scaling factor) by replacing A by $(R_k^H \cdot R_k)$ in the above loop. The equations then read:

- initialize $\mathbf{v_k}$
- iteration:
  for $r = 1, \ldots, P$

$$\mathbf{t_k} = (R_k^H \cdot R_k)^{-1} \cdot \mathbf{v_k} \tag{4.9}$$

$$\mathbf{v_k} = \frac{\mathbf{t_k}}{\|\mathbf{t_k}\|_F} \cdot \sqrt{k+j-1} \tag{4.10}$$

  end

The first equation in the iteration loop is solved for $\mathbf{t_k}$. The last equation corresponds to a constant norm constraint which replaces the unit norm constraint based operation of equation 4.8 (we assume a unit-modulus finite alphabet). Without noise, the entries in $\mathbf{v_k}$ should have an absolute value 1, and equal the constellation symbols up to a rotation.

Equation 4.9 can be refined as:

$$R_k^H \cdot \mathbf{u_k} = \mathbf{v_k} \tag{4.11}$$

$$R_k \cdot \mathbf{t_k} = \mathbf{u_k}. \tag{4.12}$$

The first equation is solved for $\mathbf{u_k}$ (using forward substitution) and the second one for $\mathbf{t_k}$ (using back substitution).

The inverse iteration procedure will require less iterations when a good initial guess of $\mathbf{v_k}$ is available from the previous time step. The convergence rate is proportional to the square of the ratio of the two smallest singular values of R_k:

$$(\frac{\sigma_{R_k}(k+j-2)}{\sigma_{R_k}(k+j-1)})^2$$

and is typically very fast (small number of iterations P) [134]. Further note that we only need an accuracy of one or a few digits because of the subsequent mapping to constellation symbols.

Above we have presented a two step iterative scheme to compute the TLS solution. However, both the number of operations and the size of the memory required in this procedure grow in each iteration step since the dimensions of R_k and R_k^H grow. The next section explains how to transform the above algorithm into an algorithm where the computational and memory requirements are constant in each iteration step.

4.2.2 RTLS-QR: version 2

1. QR UPDATING

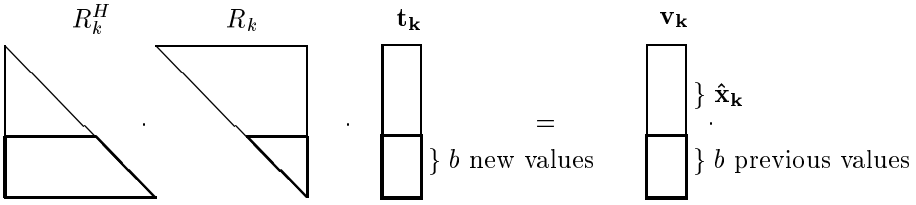
As explained in remark 4.2.2, the QR updating step only updates the last j rows of R_k in each iteration step, while the first $k-1$ rows remain unchanged. As a result, if the inverse iteration step is modified such that such 'old' rows are no longer used, they do not need to be stored any longer. We now show how the inverse iteration step can be modified such that the first $k+j-b-1$ rows of R_k are not used and hence do not have to be stored ($b \geq j$). The parameter b is a design parameter and will determine the complexity of the inverse iteration step of the final algorithm as will become clear further on.

2. Inverse Iteration

At iteration step k, we assume that the first $k+j-b-1$ components of the vector $\mathbf{v_k}$ have been accurately estimated, so that they can be replaced by a fixed value in all subsequent inverse iteration vectors, namely:

$$\mathbf{v_k}(1:k+j-b-1) \approx \hat{\mathbf{x}}(1:k+j-b-1) \tag{4.13}$$

where $\hat{\mathbf{x}}(1:k+j-b-1)$ represents the hard decisions on the first $k+j-b-1$ transmitted symbols. We choose $b \geq j$ such that the corresponding rows of R_k (rows 1 to $k+j-b-1$) remain unchanged during the QR updating step. This means that the inverse iteration step will truly iterate only over the last b components of the vector $\mathbf{v_k}$, *i.e.* $\mathbf{v_k}(k+j-b:k+j-1)$. Equation 4.9 then reads

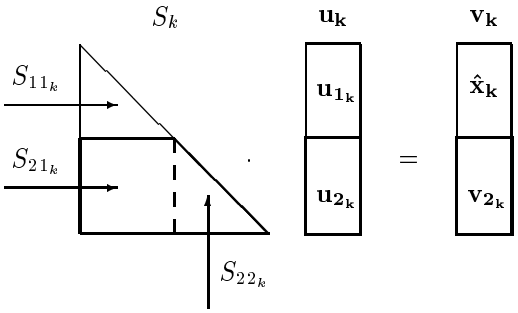


(4.14)

We are only interested in the solution of the boldfaced part of the above equation. To simplify notation further on we define:

$$S_k = R_k^H.$$

Equation 4.14 can now be split into two steps as in the equations 4.11 and 4.12. The first equation reads:



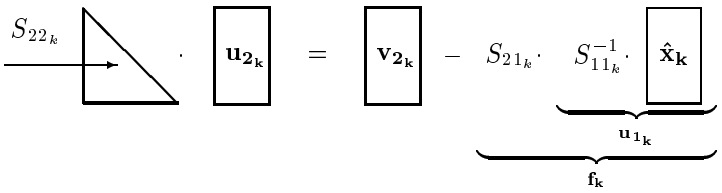
(4.15)

where we adopted the following notation:

$$\begin{aligned} S_{11_k} &= S_k(1:k+j-b-1, 1:k+j-b-1) \\ S_{21_k} &= S_k(k+j-b:k+j-1, 1:k+j-b-1) \\ S_{22_k} &= S_k(k+j-b:k+j-1, k+j-b:k+j-1) \end{aligned}$$

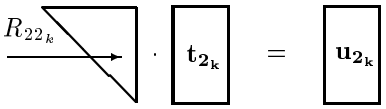
$$
\begin{aligned}
\mathbf{u_{1_k}} &= \mathbf{u_k}(1 : k+j-b-1) \\
\mathbf{u_{2_k}} &= \mathbf{u_k}(k+j-b : k+j-1) \\
\mathbf{\hat{x}_k} &= \mathbf{\hat{x}}(1 : k+j-b-1) \\
\mathbf{v_{2_k}} &= \mathbf{v_k}(k+j-b : k+j-1).
\end{aligned}
$$

Equation 4.15 is solved for $\mathbf{u_{2_k}}$ as follows:



(4.16)

where we introduced a new variable $\mathbf{f_k}$. Once $\mathbf{u_{2_k}}$ has been computed, the second step (corresponding to equation 4.12) consists of a simple back substitution



(4.17)

with

$$
\begin{aligned}
R_{22_k} &= R_k(k+j-b : k+j-1 : k+j-b : k+j-1) \\
\mathbf{t_{2_k}} &= \mathbf{t_k}(k+j-b : k+j-1).
\end{aligned}
$$

$\mathbf{t_{2_k}}$ can then be scaled appropriately so as to obtain a new estimate of $\mathbf{v_{2_k}}$. Note that the first $k+j-b-1$ rows of R_k are only needed for the computation of $\mathbf{u_{1_k}}$. The next theorem states that $\mathbf{f_k}$ can be computed recursively, i.e. using $\mathbf{f_{k-1}}$ without computing $\mathbf{u_{1_k}}$.

Theorem 4.2.3 *The vector* $\mathbf{f_k}$ *computed from the following recursive formula:*

$$
\begin{aligned}
\mathbf{f_{b-j+1}} &= 0_{b\times 1} \\
\mathbf{f_k} &= \begin{bmatrix} \mathbf{f_{k-1}}(2:b) \\ 0 \end{bmatrix} + \frac{\mathbf{\hat{x}}(k+j-b-1) - \mathbf{f_{k-1}}(1)}{S_{k-1}(k+j-b-1, k+j-b-1)} \cdot \\
&\begin{bmatrix} S_{k-1}(k+j-b : k+j-2, k+j-b-1) \\ 0 \end{bmatrix} \qquad k > b-j+1 \quad (4.18)
\end{aligned}
$$

is identical to the vector $\mathbf{f_k}$ *defined in equation 4.16 (repeated here for convenience)*

$$\mathbf{f_{b-j+1}} = 0_{b\times 1}$$

$$\mathbf{f_k} = \sum_{q=1}^{k+j-b-1} S_k(k+j-b:k+j-1,q)\cdot \mathbf{u_k}(q) \quad k > b-j+1. \quad (4.19)$$

Before we present the proof we first prove the following lemma.

Lemma 4.2.4 *For ($l \geq k$),*

$$\mathbf{u_l}(1:k+j-b-2) = \mathbf{u_{k-1}}(1:k+j-b-2). \quad (4.20)$$

Proof: The proof is based on remark 4.2.2 which can be summarized as:

$$R_l(1:k-1,:) = \left[\begin{array}{cc} R_{k-1}(1:k-1,:) & 0_{(k-1)\times(l-k+1)} \end{array}\right] \quad (4.21)$$

$$S_l(:,1:k-1) = \left[\begin{array}{c} S_{k-1}(:,1:k-1) \\ 0_{(l-k+1)\times(k-1)} \end{array}\right] \quad (4.22)$$

for $l \geq k$. Further note that for $l \geq k-1$, $\mathbf{u_l}(1:k+j-b-2)$ is computed from the following two equations,

$$\mathbf{u_l}(1) = \frac{\hat{\mathbf{x}}(1)}{S_l(1,1)} \quad (4.23)$$

for $r = 2 : k+j-b-2$

$$\mathbf{u_l}(r) = \frac{\hat{\mathbf{x}}(r) - \sum_{q=1}^{r-1} S_l(r,q)\cdot \mathbf{u_l}(q)}{S_l(r,r)} \quad (4.24)$$

end

Filling in equation 4.22 into equation 4.24 shows that for $l \geq k$, $b \geq j$:

$$\mathbf{u_l}(1:k+j-b-2) = \mathbf{u_{k-1}}(1:k+j-b-2).$$

□

Now we prove Theorem 4.2.3.

Proof:

We show that the recursive formula of equation 4.18 indeed enables to compute $\mathbf{f_k}$ as defined in equation 4.19.

The vector $\mathbf{f_{b-j+1}}$ is equal in the two formulas by definition. Using equation 4.18, $\mathbf{f_{b-j+2}}$ can be written as:

$$\mathbf{f_{b-j+2}} = \frac{\hat{\mathbf{x}}(1)}{S_{b-j+1}(1,1)} \cdot \left[\begin{array}{c} S_{b-j+1}(2:b,1) \\ 0 \end{array}\right].$$

Now making use of equation 4.22 we rewrite $\mathbf{f_{b-j+2}}$ as:

$$\mathbf{f_{b-j+2}} = \frac{\hat{\mathbf{x}}(1)}{S_{b-j+2}(1,1)} \cdot S_{b-j+2}(2:b+1,1).$$

Finally substituting 4.23 gives:

$$\mathbf{f_{b-j+2}} = \mathbf{u_{b-j+2}}(1) \cdot S_{b-j+2}(2:b+1,1)$$

which equals equation 4.19.

Now it will be shown that, for an arbitrary k, if $\mathbf{f_{k-1}}$ satisfies definition 4.19 then $\mathbf{f_k}$ computed using equation 4.18 also will.

Note that if $\mathbf{f_{k-1}}$ is computed correctly, we can write:

$$\begin{aligned}
& \frac{\hat{\mathbf{x}}(k+j-b-1) - \mathbf{f_{k-1}}(1)}{S_{k-1}(k+j-b-1, k+j-b-1)} \\
= \; & \frac{\hat{\mathbf{x}}(k+j-b-1) - \sum_{q=1}^{k+j-b-2} S_{k-1}(k+j-b-1, q) \cdot \mathbf{u_{k-1}}(q)}{S_{k-1}(k+j-b-1, k+j-b-1)} \\
= \; & \frac{\hat{\mathbf{x}}(k+j-b-1) - \sum_{q=1}^{k+j-b-2} S_{k}(k+j-b-1, q) \cdot \mathbf{u_{k}}(q)}{S_{k}(k+j-b-1, k+j-b-1)} \\
= \; & \mathbf{u_k}(k+j-b-1).
\end{aligned}$$

Filling in this relationship in equation 4.18, gives:

$$\begin{aligned}
\mathbf{f_k} = \; & \begin{bmatrix} \mathbf{f_{k-1}}(2:b) \\ 0 \end{bmatrix} + \frac{\hat{\mathbf{x}}(k+j-b-1) - \mathbf{f_{k-1}}(1)}{S_{k-1}(k+j-b-1, k+j-b-1)} \\
& \cdot \begin{bmatrix} S_{k-1}(k+j-b : k+j-2, k+j-b-1) \\ 0 \end{bmatrix} \\
= \; & \begin{bmatrix} \mathbf{f_{k-1}}(2:b) \\ 0 \end{bmatrix} + \begin{bmatrix} S_{k-1}(k+j-b : k+j-2, k+j-b-1) \\ 0 \end{bmatrix} \\
& \cdot \mathbf{u_k}(k+j-b-1) \\
= \; & \begin{bmatrix} S_{k-1}(k+j-b : k+j-2, 1 : k+j-b-1) \\ 0 \end{bmatrix} \cdot \mathbf{u_k}(1 : k+j-b-1) \\
= \; & S_k(k+j-b : k+j-1, 1 : k+j-b-1) \cdot \mathbf{u_k}(1 : k+j-b-1)
\end{aligned}$$

which completes the proof. □

With the recursive formula for $\mathbf{f_k}$ we obtain the following inverse iteration scheme:

1. compute $\mathbf{f_k}$ using equation 4.18.

2. for $l = 1 : P$

   - compute $\mathbf{u_{2_k}}$: $S_{22_k} \cdot \mathbf{u_{2_k}} = \mathbf{v_{2_k}} - \mathbf{f_k}$
   - compute $\mathbf{t_{2_k}}$: $R_{22_k} \cdot \mathbf{t_{2_k}} = \mathbf{u_{2_k}}$
   - scale: $\mathbf{v_{2_k}} = \frac{\mathbf{t_{2_k}}}{\|\mathbf{t_{2_k}}\|_F} \cdot \sqrt{b}$

   end

Note that the vector $\mathbf{v_{2_k}}$ computed in the inverse iteration procedure can also be interpreted to converge to the singular vector corresponding to the smallest singular value of R_{22_k} with a correction factor based on $\mathbf{f_k}$ (and hence based on previous symbol decisions).

This inverse iteration scheme thus avoids the use of the first $k + j - b - 1$ rows of R_k. Consequently they do not have to be stored and only the lower right triangular part of R_k needs to be tracked in the QR updating step. The final algorithm, including both the QR and inverse updating step is summarized in algorithm 4.2.2 (at this point, time indices are omitted since matrices are overwritten).

The RTLS algorithm has an approximate complexity[2]:

$$(N - j + 1) \cdot ((8 \cdot p + 11) \cdot j^2 + 2 \cdot p \cdot j) + (N - b + 1) \cdot P \cdot 8 \cdot b^2.$$

where the first term represents the complexity of the QR updating step, the last term refers to the inverse iteration.

The RTLS-QR algorithm presented above has been successfully implemented on DSP as will be detailed in chapter 5.

In the next two sections we derive two variants of the RTLS-QR algorithm. In section 4.3, an algorithm is developed that tracks the factor R^{-1} of the QR decomposition instead of the R factor. In section 4.4 an algorithm similar to the RTLS-QR algorithm is developed for a more general deconvolution problem, i.e. an algorithm that is also applicable for a set of non-homogeneous equations.

4.3 RTLS based on inverse updating (RTLS-I)

Like the RTLS algorithm based on QR updating, the RTLS algorithm based on inverse updating consists of two steps. The second step is again based on inverse iteration, but in the first step the factor R^{-1} is tracked instead of the factor R. This then allows to rewrite the inverse iteration step without

[2] We assumed Householder transformations in the QR updating step.

Algorithm 4.2.2
RTLS based on QR updating

1. *initialization:*

$$\begin{aligned} \mathbf{v} &\leftarrow \begin{bmatrix} 1 & 0_{1\times(b-1)} \end{bmatrix}^T \\ R &\leftarrow 0_{b\times b} \\ \mathbf{f} &\leftarrow 0_{b\times 1} \end{aligned}$$

2. *iteration equations:*
for $k = 1, \ldots, N-j+1$

- *update* R *and* S*:*
if $k \leq b-j$

$$\begin{aligned} \begin{bmatrix} \frac{R}{0} \end{bmatrix} &\leftarrow Q^H \cdot \left[\begin{array}{ccc} R & & \\ \hline 0_{p\times(k-1)} & W_k & 0_{p\times(b-j-k+1)} \end{array}\right] \\ S &\leftarrow R^H \end{aligned}$$

else

$$\begin{aligned} \begin{bmatrix} \frac{R}{0} \end{bmatrix} &\leftarrow Q^H \cdot \left[\begin{array}{c} R \\ \hline 0_{p\times(b-j)} W_k \end{array}\right] \\ S &\leftarrow R^H \end{aligned}$$

end
if $k \geq b-j+1$

- *update* $\mathbf{v}$*:*
for $r = 1, \ldots, P$

$$\begin{aligned} S\cdot\mathbf{u} &\leftarrow \mathbf{v}-\mathbf{f} && \textit{forward substitution} \\ R\cdot\mathbf{t} &\leftarrow \mathbf{u} && \textit{back substitution} \\ \mathbf{v} &\leftarrow \frac{\mathbf{t}}{\|\mathbf{t}\|_F}\cdot\sqrt{b} \end{aligned}$$

end

- *make a decision:*

$$\hat{\mathbf{x}}(k+j-b) \leftarrow \mathcal{D}(\mathbf{v}(1))$$

- *update for next iteration:*

$$\begin{aligned} \mathbf{f} &\leftarrow \begin{bmatrix} \mathbf{f}(2:b) \\ 0 \end{bmatrix} + \frac{\hat{\mathbf{x}}(k+j-b)-\mathbf{f}(1)}{S(1,1)}\cdot\begin{bmatrix} S(2:b,1) \\ 0 \end{bmatrix} \\ R &\leftarrow \begin{bmatrix} R(2:b,2:b) & 0_{(b-1)\times 1} \\ 0_{1\times(b-1)} & 0 \end{bmatrix} \\ \mathbf{v} &\leftarrow \begin{bmatrix} \mathbf{v}(2:b) \\ 0 \end{bmatrix} \end{aligned}$$

end
end

$$\hat{x}(N-b+2:N) \leftarrow \mathcal{D}(\mathbf{v}(1:b-1))$$

using back substitution, which may be interesting e.g. for pipelining purposes [102, 94]. Again we first present a version with growing matrix dimensions, followed by an algorithm that has computational and memory requirements that are constant in each iteration.

4.3.1 RTLS-I: version 1

1. Inverse Updating

First recall equation 4.6 which is repeated here for convenience:

$$\mathcal{W}_k = \left[\begin{array}{c|c} \mathcal{W}_{k-1} & 0_{p\cdot(k-1)\times 1} \\ \hline 0_{p\times(k-1)} & \boxed{W_k} \end{array}\right].$$

Assume that at iteration step $k-1$ the factor R_{k-1}^{-1} has been computed, such that

$$\mathcal{W}_{k-1} = Q_{k-1} \cdot (R_{k-1}^{-1})^{-1},$$

now we show how to update R_{k-1}^{-1} to R_k^{-1} without needing Q_{k-1}. The inverse updating can then be considered as a data compression step.

Adding a column of zeros to $\mathcal{W}_{k-1}$ is equivalent to the following operation:

$$\begin{aligned} \left[\begin{array}{cc} \mathcal{W}_{k-1} & 0 \end{array}\right] &= \left[\begin{array}{cc} Q_{k-1} & \mathbf{q} \end{array}\right] \cdot \left[\begin{array}{cc} R_{k-1} & 0 \\ 0 & 0 \end{array}\right] \\ &\approx \left[\begin{array}{cc} Q_{k-1} & \mathbf{q} \end{array}\right] \cdot \left(\left[\begin{array}{cc} R_{k-1}^{-1} & 0 \\ 0 & \alpha \end{array}\right]\right)^{-1}, \end{aligned}$$

with $\mathbf{q}$ again a unit-length vector in the null space of Q_{k-1}. The inverse of the updated rank-deficient matrix R_{k-1} does not exist, but it can be approximated by introducing a large constant α. In the second step, we compute R_k^{-1} using the square-root covariance RLS algorithm. We only state the main formula, details can be found in [102, 96].

$$F_k = \left[\begin{array}{cc} R_{k-1}^{-H} & 0 \\ 0 & \alpha \end{array}\right] \tag{4.25}$$

$$\left[\begin{array}{c|c} 0_{(k+j-1)\times p} & R_k^{-H} \\ \hline \delta & -\delta \cdot K^H \end{array}\right] = Q^{'H} \cdot \left[\begin{array}{c|c} -F_k \cdot \left[\begin{array}{c} 0_{(k-1)\times p} \\ W_k^H \end{array}\right] & F_k \\ \hline I_{p\times p} & 0_{p\times(k+j-1)} \end{array}\right] \tag{4.26}$$

Here $Q^{'H}$ is a product of unitary matrix transformations which cancel out the upper left part in the rhs compound matrix.

2. Inverse iteration

The desired right singular vector $\mathbf{v_k}$ is now computed using R_k^{-1} in an inverse iteration procedure. Analog to the previous section we have:

- initialize $\mathbf{v_k}$
- iteration:
  for $r = 1, \ldots, P$

$$\begin{aligned} \mathbf{u_k} &= R_k^{-H} \cdot \mathbf{v_k} && (4.27)\\ \mathbf{t_k} &= R_k^{-1} \cdot \mathbf{u_k} && (4.28)\\ \mathbf{v_k} &= \frac{\mathbf{t_k}}{\|\mathbf{t_k}\|_F} \cdot \sqrt{k+j-1}. && (4.29) \end{aligned}$$

  end

4.3.2 RTLS-I: version 2

For simplicity of notation we define:

$$\begin{aligned} \tilde{S}_k &= S_k^{-1} = R_k^{-H} \\ &= \begin{bmatrix} \tilde{S}_{11_k} & \\ \tilde{S}_{21_k} & \tilde{S}_{22_k} \end{bmatrix} = \begin{bmatrix} S_{11_k}^{-1} & \\ -S_{22_k}^{-1} S_{21_k} S_{11_k}^{-1} & S_{22_k}^{-1} \end{bmatrix} && (4.30)\\ \tilde{R}_k &= R_k^{-1} \\ &= \begin{bmatrix} \tilde{R}_{11_k} & \tilde{R}_{12_k} \\ & \tilde{R}_{22_k} \end{bmatrix} = \begin{bmatrix} R_{11_k}^{-1} & -R_{11_k}^{-1} R_{12_k} R_{22_k}^{-1} \\ & R_{22_k}^{-1} \end{bmatrix} && (4.31)\\ F_k &= \begin{bmatrix} F_{11_k} & \\ F_{21_k} & F_{22_k} \end{bmatrix} \end{aligned}$$

with

$$\begin{aligned} \tilde{S}_{11_k} &= \tilde{S}_k(1 : k+j-b-1, 1 : k+j-b-1) \\ \tilde{S}_{21_k} &= \tilde{S}_k(k+j-b : k+j-1, 1 : k+j-b-1) \\ \tilde{S}_{22_k} &= \tilde{S}_k(k+j-b : k+j-1, k+j-b : k+j-1), \end{aligned}$$

and all other submatrices defined in a similar way.

1. INVERSE UPDATING

We now modify the inverse updating step such that only the lower right triangular part of F_k (and $\tilde{S}_k$): F_{22_k} (and $\tilde{S}_{22_k}$) needs to be tracked. The crucial observation is that the orthogonal matrix Q' in equation 4.26 is completely determined by F_{22_k} and W_k $(b \geq j)$ because F_k is lower triangular:

$$-F_k \cdot \begin{bmatrix} 0_{(k-1)\times p} \\ W_k^H \end{bmatrix} = \begin{bmatrix} 0_{(k+j-1-b)\times p} \\ -F_{22_k} \cdot \begin{bmatrix} 0_{(b-j)\times p} \\ W_k^H \end{bmatrix} \end{bmatrix}.$$

This allow to track $\tilde{S}_{22_k}$ separately in the inverse updating step.

$$\left[\begin{array}{c|c} 0_{b\times p} & \tilde{S}_{22_k} \\ \hline \delta & -\delta \cdot K^H \end{array}\right] = Q^H \cdot \left[\begin{array}{c|c} -F_{22_k} \cdot \begin{bmatrix} 0_{(b-j)\times p} \\ W_k^H \end{bmatrix} & F_{22_k} \\ \hline I_{p\times p} & 0_{p\times b} \end{array}\right]. \quad (4.32)$$

Further note that

$$\tilde{S}_{21_k} = Q^H F_{21_k}. \quad (4.33)$$

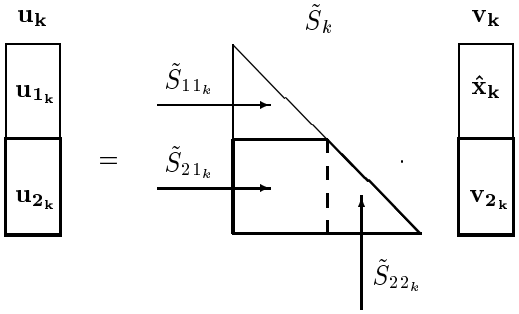
2. INVERSE ITERATION

As for the RTLS-QR algorithm, at iteration step k we assume that the first $k+j-b-1$ components of $\mathbf{v_k}$ have been accurately estimated:

$$\mathbf{v_k}(1:k+j-b-1) \approx \hat{\mathbf{x}}(1:k+j-b-1). \quad (4.34)$$

Again the inverse iteration step will iterate only over the last b components of $\mathbf{v_k}$. This will allow to rewrite the inverse iteration step such that only $\tilde{R}_{22_k}$ and $\tilde{S}_{22_k}$ are needed.

Based on equation 4.27 we get:



(4.35)

where we adopted the following notation:

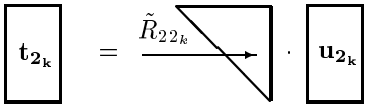
$$\mathbf{u_{1_k}} = \mathbf{u_k}(1:k+j-b-1)$$

$$\begin{aligned}
\mathbf{u_{2_k}} &= \mathbf{u_k}(k+j-b:k+j-1) \\
\hat{\mathbf{x}}_\mathbf{k} &= \hat{\mathbf{x}}(1:k+j-b-1) \\
\mathbf{v_{2_k}} &= \mathbf{v_k}(k+j-b:k+j-1).
\end{aligned}$$

Equation 4.35 has to be solved for $\mathbf{u_{2_k}}$ which can be done as follows:

$$\mathbf{u_{2_k}} = \underbrace{\tilde{S}_{21_k} \cdot \hat{\mathbf{x}}_\mathbf{k}}_{\mathbf{f_k}} + \tilde{S}_{22_k} \cdot \mathbf{v_{2_k}} \tag{4.36}$$

where we introduced the variable $\tilde{\mathbf{f}}_\mathbf{k}$. Once $\mathbf{u_{2_k}}$ has been computed, the second step (corresponding to equation 4.28) consists of a simple multiplication:



(4.37)

with $\mathbf{t_{2_k}} = \mathbf{t_k}(k+j-b:k+j-1)$. $\mathbf{t_{2_k}}$ can then be scaled appropriately so as to obtain a new estimate of $\mathbf{v_{2_k}}$. Note that $\tilde{S}_{21_k}$ is only needed for the computation of $\tilde{\mathbf{f}}_\mathbf{k}$. The next theorem states that $\tilde{\mathbf{f}}_\mathbf{k}$ can be computed recursively, i.e. using $\tilde{\mathbf{f}}_\mathbf{k-1}$ without needing $\tilde{S}_{21_k}$.

Theorem 4.3.1 *The vector $\tilde{\mathbf{f}}_\mathbf{k}$ computed from the following recursive formulas:*

$$\begin{aligned}
\tilde{\mathbf{f}}_\mathbf{b-j+1} &= 0_{b\times 1} \\
\mathbf{e_k} &= \begin{bmatrix} \tilde{\mathbf{f}}_\mathbf{k-1}(2:b) \\ 0 \end{bmatrix} + \\
&\quad \hat{\mathbf{x}}(k+j-b-1) \cdot \begin{bmatrix} \tilde{S}_{k-1}(k+j-b:k+j-2, k+j-b-1) \\ 0 \end{bmatrix} \\
\tilde{\mathbf{f}}_\mathbf{k} &= Q^H \cdot \mathbf{e_k} \qquad k > b-j+1
\end{aligned}$$

and Q^H as defined in equation 4.32 is identical to the vector $\tilde{\mathbf{f}}_\mathbf{k}$ defined in equation 4.36 (repeated here for convenience)

$$\begin{aligned}
\tilde{\mathbf{f}}_\mathbf{b-j+1} &= 0_{b\times 1} \\
\tilde{\mathbf{f}}_\mathbf{k} &= \tilde{S}_k(k+j-b:k+j-1, 1:k+j-b-1) \cdot \hat{\mathbf{x}}(1:k+j-b-1) \\
&\qquad\qquad k > b-j+1.
\end{aligned}$$

Proof: The proof is by induction. Clearly $\tilde{\mathbf{f}}_\mathbf{b-j+1} = 0_{b\times 1}$ is equal in both definitions. Assume $\tilde{\mathbf{f}}_\mathbf{k-1}$ has been computed correctly. Now we show that $\tilde{\mathbf{f}}_\mathbf{k}$ computed via the recursive formula will also be correct.

If we fill in the non-recursive definition of $\tilde{\mathbf{f}}_{\mathbf{k-1}}$ in the recursive formula for $\mathbf{e_k}$, we obtain:

$$\mathbf{e_k} = \begin{bmatrix} \tilde{S}_{k-1}(k+j-b : k+j-2, 1 : k+j-b-1) \\ 0_{1\times(k+j-b-1)} \end{bmatrix} \cdot \hat{\mathbf{x}}(1 : k+j-b-1).$$

What remains to be proven is:

$$\tilde{S}_k(k+j-b : k+j-1, 1 : k+j-b-1) = \\ Q^H \cdot \begin{bmatrix} \tilde{S}_{k-1}(k+j-b : k+j-2, 1 : k+j-b-1) \\ 0_{1\times(k+j-b-1)} \end{bmatrix}$$

which easily follows from equation 4.33 and the definition of F_k (equation 4.25). □

Remark 4.3.2 It is interesting to note that there exists a simple relation between the vector $\mathbf{f_k}$ computed in the RTLS-QR algorithm (equation 4.16) and the vector $\tilde{\mathbf{f}}_\mathbf{k}$ computed in the RTLS-I algorithm (equation 4.36):

$$\begin{aligned} \mathbf{f_k} &= S_{21_k} \cdot S_{11_k}^{-1} \cdot \hat{\mathbf{x}}_\mathbf{k} \\ \tilde{\mathbf{f}}_\mathbf{k} &= \tilde{S}_{21_k} \cdot \hat{\mathbf{x}}_\mathbf{k}. \end{aligned}$$

From equation 4.30 we have

$$\tilde{S}_{21_k} = -S_{22_k}^{-1} S_{21_k} S_{11_k}^{-1}.$$

Replacing this equation into the definition of $\tilde{\mathbf{f}}_\mathbf{k}$ we obtain:

$$\begin{aligned} \tilde{\mathbf{f}}_\mathbf{k} &= \tilde{S}_{21_k} \cdot \hat{\mathbf{x}}_\mathbf{k} \\ &= -S_{22_k}^{-1} \cdot \underbrace{S_{21_k} \cdot S_{11_k}^{-1} \cdot \hat{\mathbf{x}}_\mathbf{k}}_{\mathbf{f_k}}. \end{aligned}$$

The final algorithm is summarized as algorithm 4.3.2.

Remark 4.3.3 In the QR updating step of the RTLS-I algorithm only Givens rotations can be used (no Householder transformations), in order to maintain the lower triangular structure of F_{22_k} (equation 4.32). This leads to the following computational complexity:

$$(b-j)\cdot(18p^2j+21pj+18pbj-9pj^2)+(N-b+1)\cdot(18p^2j+39pj+18pbj-9pj^2) \\ +(N-b+1)\cdot(P(8b^2+14b)+8b).$$

The first two terms form the complexity of the QR updating step, using Givens rotations, and the last term is the approximate complexity of the inverse updating step.

Algorithm 4.3.2
RTLS based on inverse updating

1. *initialization:*

$$\begin{aligned} \mathbf{v} &\leftarrow \begin{bmatrix} 1 & 0_{1\times(b-1)} \end{bmatrix}^T \\ \alpha &\leftarrow \textit{large constant} \\ \tilde{S} &\leftarrow \alpha \cdot I_{b\times b} \\ \tilde{\mathbf{f}} &\leftarrow 0_{b\times 1} \end{aligned}$$

2. *iteration equations:*
for $k = 1, \ldots, N-j+1$

- *update* $\tilde{R}$ *and* $\tilde{S}$:
if $k \leq b-j$

$$\left[\begin{array}{c|c} 0_{b\times p} & \tilde{S} \\ \hline \delta & -\delta\cdot K^H \end{array}\right] \leftarrow Q^H \cdot \left[\begin{array}{c|c} -\tilde{S}\cdot \begin{bmatrix} 0_{(k-1)\times p} \\ W_k^H \\ 0_{(b-j-k+1)\times p} \end{bmatrix} & \tilde{S} \\ \hline I_{p\times p} & 0_{p\times b} \end{array}\right]$$

$$\tilde{R} \leftarrow \tilde{S}^H$$

else

$$\left[\begin{array}{c|c|c} 0_{b\times p} & \tilde{S} & \tilde{\mathbf{f}} \\ \hline \delta & -\delta\cdot K^H & \star \end{array}\right] \leftarrow Q^H \cdot \left[\begin{array}{c|c|c} -\tilde{S}\cdot \begin{bmatrix} 0_{(b-j)\times p} \\ W_k^H \end{bmatrix} & \tilde{S} & \tilde{\mathbf{f}} \\ \hline I_{p\times p} & 0_{p\times b} & 0 \end{array}\right]$$

$$\tilde{R} \leftarrow \tilde{S}^H$$

end
if $k \geq b-j+1$

- *update* $\mathbf{v}$:
for $r = 1, \ldots, P$

$$\begin{aligned} \mathbf{t} &\leftarrow \tilde{R}\cdot(\tilde{\mathbf{f}} + \tilde{S}\cdot\mathbf{v}) \\ \mathbf{v} &\leftarrow \frac{\mathbf{t}}{\|\mathbf{t}\|_F}\cdot\sqrt{b} \end{aligned}$$

end

- *make a decision:*

$$\hat{\mathbf{x}}(k+j-b) \leftarrow \mathcal{D}(\mathbf{v}(1))$$

- *update for next iteration:*

$$\begin{aligned} \tilde{\mathbf{f}} &\leftarrow \begin{bmatrix} \tilde{\mathbf{f}}(2:b) \\ 0 \end{bmatrix} + \hat{\mathbf{x}}(k+j-b)\cdot\begin{bmatrix} \tilde{S}(2:b,1) \\ 0 \end{bmatrix} \\ \tilde{S} &\leftarrow \begin{bmatrix} \tilde{S}(2:b,2:b) & 0_{(b-1)\times 1} \\ 0_{1\times(b-1)} & \alpha \end{bmatrix} \\ \mathbf{v} &\leftarrow \begin{bmatrix} \mathbf{v}(2:b) \\ 0 \end{bmatrix} \end{aligned}$$

end
end

$$\hat{x}(N-b+2:N) \leftarrow \mathcal{D}(\mathbf{v}(1:b-1))$$

Remark 4.3.4 The performance of the RTLS-I algorithm is identical to that of the RTLS-QR algorithm (infinite precision). Performance differences will only occur in simulations with (short) finite word lengths, which is beyond the scope of this work. Therefore in chapter 5 we will analyze only the performance of the RTLS-QR algorithm.

The RTLS-QR and RTLS-I algorithms have both shown how the TLS problem can be solved recursively for the set of homogeneous equations (equation 3.9) by exploiting the sparsity of $\mathcal{W}_k$. This has led to two elegant algorithms in which complexity and memory requirements can be traded against performance. In the next section we extend these ideas to a set of non-homogeneous equations.

4.4 RTLS for deconvolution problems (RTLS-D)

In this section, we show how the RTLS algorithm based on QR updating can be generalized to solve other deconvolution problems (the extension of the algorithm based on inverse updating can then be made in a similar way), i.e. deconvolution problems where a set of non-homogeneous equations are encountered.

Deconvolution problems arise in digital signal processing applications as diverse as high speed data transmission, reverberation cancelation, reflection seismology, biomedical signal processing and image restoration [91, 60]. In these situations an unknown input signal can only be observed after propagation through one or more noise corrupted FIR channels. The FIR channels can be estimated through training based or blind algorithms. Once these estimates are available, the input signal can be retrieved in several ways.

In digital communications, the input belongs to a finite alphabet and a Viterbi algorithm can be used. However, the complexity of the Viterbi algorithm grows exponentially with the length of the FIR channels (see also section 3.2). Therefore cheaper solutions are desirable. Alternatively, one can consider the problem as a least squares problem in the unknown transmitted sequence. A recursive implementation with constant computational cost is based on the Kalman filter. If we assume that errors are present in both the output samples and the estimates of the channel parameters, a total least squares (TLS) solution [53, 134] can provide better parameter estimates than conventional least squares methods. For reasons of computational complexity it is desirable to turn the classical TLS algorithm into an adaptive form.

In the next sections we present such a recursive solution that has a constant computational complexity in each iteration step. First section 4.4.1 succinctly describes the data model. Section 4.4.2 and 4.4.3 explain how the RTLS-QR

algorithm presented in section 4.2 is modified to accommodate a non-trivial right-hand-side.

4.4.1 Data model

We again assume the SIMO data model of equation 2.5, where the channel may be time-varying:

$$\underbrace{\begin{bmatrix} \mathbf{y_{L+1}} \\ \vdots \\ \mathbf{y_N} \end{bmatrix}}_{\mathbf{y_{L+1|N}}} = \underbrace{\begin{bmatrix} \boxed{H^{(L+1)}} & 0 & 0 & \dots \\ 0 & \boxed{H^{(L+2)}} & 0 & \dots \\ & & \ddots & \\ 0 & 0 & \dots & \boxed{H^{(N)}} \end{bmatrix}}_{\mathcal{H}_{N-L}} \cdot \underbrace{\begin{bmatrix} x(1) \\ \vdots \\ x(N) \end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix} \mathbf{n_{L+1}} \\ \vdots \\ \mathbf{n_N} \end{bmatrix}}_{\mathbf{n_{L+1|N}}}. \tag{4.38}$$

$\mathcal{H}_{N-L}$ is an $(N-L)M \times N$ matrix. We assume that a (noisy) estimate of $\mathcal{H}_{N-L}$ has been obtained and focus on the recovery of the input vector $\mathbf{x}$ using a TLS type approach. Note that this corresponds to the TLS problem formulation of equation 4.1. We will *not* enforce $\Delta\mathcal{H}_{N-L}$ to have the same structure as $\mathcal{H}_{N-L}$ although this may give additional performance advantages [25, 26, 74, 73]. Recall that the TLS solution of equation 4.38 is proportional to the right singular vector corresponding to the smallest singular value of the matrix $\mathcal{W}_N = \begin{bmatrix} \mathcal{H}_{N-L} & \mathbf{y_{L+1|N}} \end{bmatrix}$, where a scaling is applied such that the last entry in the singular vector equals -1. We now present an algorithm which computes this solution recursively.

As for the algorithms presented in the previous sections, we first derive an algorithm with growing matrix dimensions, which we will then transform into a solution with constant computational requirements.

4.4.2 RTLS-D: version 1

At time $k-1$, one obtains a large overdetermined set of equations in $\mathbf{x}(1:k-1)$. The total least squares solution $\mathbf{x}(1:k-1)$ then equals the right singular vector (up to a scaling factor) associated with the smallest singular value of $\mathcal{W}_{k-1}$:

$$\mathcal{W}_{k-1} = \left[\begin{array}{cccc|c} \boxed{H^{(L+1)}} & 0 & 0 & \dots & \mathbf{y_{L+1}} \\ 0 & \boxed{H^{(L+2)}} & 0 & \dots & \mathbf{y_{L+2}} \\ & & \ddots & & \vdots \\ 0 & 0 & \dots & \boxed{H^{(k-1)}} & \mathbf{y_{k-1}} \end{array}\right]. \tag{4.39}$$

Remark 4.4.1 In this data model $L+2$ corresponds to the parameter j of the previous sections.

As for the RTLS-QR algorithm presented in section 4.2, the right singular vector is computed iteratively in a two-step procedure.

1. QR UPDATING

Denote the 'skinny' QR decomposition of $\mathcal{W}_{k-1}$ as $\mathcal{W}_{k-1} = Q_{k-1} \cdot R_{k-1}$ and assume that the $k \times k$ matrix R_{k-1} is known at time step $k-1$. We now show how to update R_{k-1} to R_k.

First we define the permutation matrix P_k as:

$$P_k = \begin{bmatrix} I_{k-1} & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{4.40}$$

The matrix $\mathcal{W}_k$ is computed from the matrix $\mathcal{W}_{k-1}$ in the following way:

$$\begin{aligned} \mathcal{W}_k &= \begin{bmatrix} \begin{bmatrix} \mathcal{W}_{k-1} & 0 \end{bmatrix} \cdot P_k \\ \hline \begin{bmatrix} 0_{M\times(k-(L+1))} & \boxed{H^{(k)}} & \mathbf{y_k} \end{bmatrix} \end{bmatrix} \\ &= \begin{bmatrix} \begin{bmatrix} Q_{k-1} & \mathbf{q} \end{bmatrix} \cdot \begin{bmatrix} R_{k-1} & 0 \\ 0 & 0 \end{bmatrix} \cdot P_k \\ \hline \begin{bmatrix} 0_{M\times(k-(L+1))} & \boxed{H^{(k)}} & \mathbf{y_k} \end{bmatrix} \end{bmatrix} \\ &= \begin{bmatrix} Q_{k-1} & \mathbf{q} & 0 \\ 0 & 0 & I_M \end{bmatrix} \cdot \begin{bmatrix} \begin{bmatrix} R_{k-1} & 0 \\ 0 & 0 \end{bmatrix} \cdot P_k \\ \hline \begin{bmatrix} 0_{M\times(k-(L+1))} & \boxed{H^{(k)}} & \mathbf{y_k} \end{bmatrix} \end{bmatrix} \\ &= \underbrace{\begin{bmatrix} Q_{k-1} & \mathbf{q} & 0 \\ 0 & 0 & I_M \end{bmatrix}}_{\begin{bmatrix} Q_k & \star \end{bmatrix}} Q \, \underbrace{Q^H \begin{bmatrix} \begin{bmatrix} R_{k-1} & 0 \\ 0 & 0 \end{bmatrix} \cdot P_k \\ \hline \begin{bmatrix} 0_{M\times(k-(L+1))} & \boxed{H^{(k)}} & \mathbf{y_k} \end{bmatrix} \end{bmatrix}}_{\begin{bmatrix} R_k \\ \hline 0 \end{bmatrix}}, \end{aligned} \tag{4.41, 4.42}$$

where $\mathbf{q}$ is again an arbitrary unit-length vector in the null space of Q_{k-1}. The matrix Q^H is the product of Givens rotations or Householders transformations. Again we have updated R_{k-1} to R_k without needing Q_{k-1} (or $\mathbf{q}$).

2. INVERSE ITERATION

The inverse iteration step is analog to that of the previous section, except that we now scale the TLS solution such that its last entry equals -1, as explained in section 4.1.

- initialize $\mathbf{v_k}$, $S_k = R_k^H$
- iteration:
  for $r = 1, \ldots, P$

$$S_k \cdot \mathbf{u_k} = \mathbf{v_k} \tag{4.43}$$

$$R_k \cdot \mathbf{t_k} = \mathbf{u_k} \tag{4.44}$$

$$\mathbf{v_k} = -\frac{\mathbf{t_k}}{\mathbf{t_k}(k+1)}. \tag{4.45}$$

  end

Clearly the computations required in this procedure grow in time since the dimensions of R_k and R_k^H grow. The next section explains how to transform the above algorithm in an algorithm with constant computational requirements.

4.4.3 RTLS-D: version 2

1. QR UPDATING

The first $k - L - 1$ rows of R_{k-1} remain unchanged in the QR updating step after iteration $k - 1$ (except for the insertion of some zero columns), because of the initial zero block in the appended rows in equation 4.41. In the next section the inverse iteration step will be modified such that the first $k + 1 - b$ rows of R_{k-1} are no longer used ($b \geq L + 2$). Hence in the final algorithm these rows do not have to be stored any longer.

2. INVERSE ITERATION

At time step k, we assume that the first $k + 1 - b$ components of the vector $\mathbf{v_k}$ have been accurately estimated ($b \geq L + 2$):

$$\mathbf{v_k}(1 : k + 1 - b) \approx \hat{\mathbf{x}}(1 : k + 1 - b). \tag{4.46}$$

The inverse iteration step will truly iterate only over $\mathbf{v_k}(k + 2 - b : k + 1)$. In order to simplify notation, we introduce the following definitions:

$$\begin{aligned} S_{11_k} &= S_k(1 : k + 1 - b, 1 : k + 1 - b) \\ S_{21_k} &= S_k(k + 2 - b : k + 1, 1 : k + 1 - b) \\ S_{22_k} &= S_k(k + 2 - b : k + 1, k + 2 - b : k + 1) \end{aligned}$$

$$
\begin{aligned}
\mathbf{u_{1_k}} &= \mathbf{u_k}(1 : k+1-b) \\
\mathbf{u_{2_k}} &= \mathbf{u_k}(k+2-b : k+1) \\
\mathbf{\hat{x}_k} &= \mathbf{\hat{x}}(1 : k+1-b) \\
\mathbf{v_{2_k}} &= \mathbf{v_k}(k+2-b : k+1) \\
R_{22_k} &= R_k(k+2-b : k+1, k+2-b : k+1) \\
\mathbf{t_{2_k}} &= \mathbf{t_k}(k+2-b : k+1)
\end{aligned}
$$

We then rewrite the inverse iteration step as a function of $\mathbf{v_k}(k+2-b : k+1)$:

$$
\begin{aligned}
S_{11_k} \cdot \mathbf{u_{1_k}} &= \mathbf{\hat{x}_k} && (4.47) \\
\mathbf{f_k} &= S_{21_k} \cdot \mathbf{u_{1_k}} \text{ for } k > b-1 \text{ and } \mathbf{f_{b-1}} = 0_{b\times 1}. && (4.48) \\
S_{22_k} \cdot \mathbf{u_{2_k}} &= \mathbf{v_{2_k}} - \mathbf{f_k} && (4.49) \\
R_{22_k} \cdot \mathbf{t_{2_k}} &= \mathbf{u_{2_k}} && (4.50) \\
\mathbf{v_{2_k}} &= -\frac{\mathbf{t_{2_k}}}{\mathbf{t_k}(k+1)} && (4.51)
\end{aligned}
$$

In the following theorem it is shown how $\mathbf{f_k}$ is computed recursively.

Theorem 4.4.2 *The vector* $\mathbf{f_k}$ *computed from the following recursive formulas:*

$$
\begin{aligned}
\mathbf{f_{b-1}} &= 0_{b\times 1} \\
\mathbf{f_k} &= \begin{bmatrix} \mathbf{f_{k-1}}(2 : b-1) \\ 0 \\ \mathbf{f_{k-1}}(b) \end{bmatrix} + \frac{\hat{x}(k+1-b) - \mathbf{f_{k-1}}(1)}{S_{k-1}(k+1-b, k+1-b)} \cdot \\
&\quad \begin{bmatrix} S_{k-1}(k+2-b : k-1, k+1-b) \\ 0 \\ S_{k-1}(k, k+1-b) \end{bmatrix}
\end{aligned}
$$

is identical to the vector $\mathbf{f_k}$ *defined in equation 4.48 (repeated here for convenience)*

$$
\begin{aligned}
\mathbf{f_{b-1}} &= 0_{b\times 1} \\
\mathbf{f_k} &= \sum_{q=1}^{k+1-b} S_k(k+2-b : k+1, q) \cdot \mathbf{u_k}(q).
\end{aligned}
$$

Proof: The zero matrix in the rhs of equation 4.41 enforces the following relationships for $l \geq k$:

$$
R_l(1 : k-L-1, :) = \begin{bmatrix} R_{k-1}(1 : k-L-1, 1 : k-1) & 0_{(k-L-1)\times(l-k+1)} & R_{k-1}(1 : k-L-1, k) \end{bmatrix}
$$

$$\mathbf{u_l}(1:k-b) = \mathbf{u_{k-1}}(1:k-b).$$

The proof can now easily be constructed as in Theorem 4.2.3. □

In the inverse updating step, now only $\mathbf{f_k}$ and R_{22_k} are needed. In the QR updating step, we can easily update R_{22_k} (see the previous section). The final algorithm is summarized as algorithm 4.4.3.

Remark 4.4.3 The computational complexity of the RTLS-D algorithm equals approximately:

$$(N-L)\cdot(8\cdot M\cdot L^2+34\cdot M\cdot L+11\cdot L^2)+(N-b+2)\cdot 8\cdot P\cdot b^2.$$

4.5 Conclusions

In this chapter we have presented two recursive total least squares algorithms for single user blind symbol estimation. Both algorithms start from a set of homogeneous equations in the unknown transmitted symbols, as derived in the previous chapter. In the first algorithm (RTLS-QR algorithm) the total least squares solution is computed based on QR updating and inverse iteration. In the second algorithm (RTLS-I algorithm), the solution is based on inverse updating and inverse iteration. In both algorithms, we have shown that exploiting the sparse structure of the set of homogeneous equations is essential in obtaining the recursive solutions. Furthermore, both algorithms employ a specific decision feedback mechanism to keep computational and memory requirements constant in each iteration step. In section 4.4 we have shown how the RTLS-QR algorithm can be used to solve more general deconvolution problems where we start from a set of non-homogeneous equations (RTLS-D algorithm). In the next chapter we will compare the complexity and performance of the RTLS algorithms of this chapter, the Viterbi and Kalman algorithm of the previous chapter and the block processing algorithm of [80]. Furthermore we compare the complexity and performance of the RTLS-D algorithm with that of the standard TLS and LS algorithms.

Algorithm 4.4.3
RTLS algorithm for deconvolution problems

1. *initialization:*

$$\begin{aligned} \mathbf{v} &\leftarrow \begin{bmatrix} 0_{1\times(b-1)} & -1 \end{bmatrix}^T \\ R &\leftarrow 0_{b\times b} \\ \mathbf{f} &\leftarrow 0_{b\times 1} \end{aligned}$$

2. *iteration equations:*
for $k = L+1, \ldots, N$
   - *update* R *and* S
     if $k < b-1$

$$\begin{aligned} \begin{bmatrix} \frac{R}{0} \end{bmatrix} &\leftarrow Q^H \cdot \begin{bmatrix} R & & & \\ 0_{M\times(k-(L+1))} & H^{(k)} & 0_{M\times(b-k-1)} & \mathbf{y_k} \end{bmatrix} \\ S &\leftarrow R^H \end{aligned}$$

     else

$$\begin{aligned} \begin{bmatrix} \frac{R}{0} \end{bmatrix} &\leftarrow Q^H \cdot \begin{bmatrix} \frac{R}{0_{M\times(b-(L+2))} \quad H^{(k)} \quad \mathbf{y_k}} \end{bmatrix} \\ S &\leftarrow R^H \end{aligned}$$

     end
   if $k \geq b-1$
   - *update the* v*-vector,*
     for $r = 1, \ldots, P$

$$\begin{aligned} S\cdot\mathbf{u} &\leftarrow \mathbf{v}-\mathbf{f} && \textit{forward substitution} \\ R\cdot\mathbf{t} &\leftarrow \mathbf{u} && \textit{back substitution} \\ \mathbf{v} &\leftarrow -\frac{\mathbf{t}}{\mathbf{t}(b)} \end{aligned}$$

     end
   - *estimate the input:*

$$\hat{\mathbf{x}}(k+2-b) \leftarrow \mathbf{v}(1)$$

   - *append a column of zeros:*

$$\begin{aligned} \mathbf{f} &\leftarrow \begin{bmatrix} \mathbf{f}(2:b-1) \\ 0 \\ \mathbf{f}(b) \end{bmatrix} + \frac{\hat{\mathbf{x}}(k+2-b)-\mathbf{f}(1)}{S(1,1)} \cdot \begin{bmatrix} S(2:b-1,1) \\ 0 \\ S(b,1) \end{bmatrix} \\ R &\leftarrow \begin{bmatrix} R(2:b,2:b-1) & 0_{(b-1)\times 1} & R(2:b,b) \\ 0_{1\times(b-2)} & 0 & 0 \end{bmatrix} \\ \mathbf{v} &\leftarrow \begin{bmatrix} \mathbf{v}(2:b-1) \\ 0 \\ \mathbf{v}(b) \end{bmatrix} \end{aligned}$$

end
end

$$\hat{\mathbf{x}}(N-b+3:N) \leftarrow \mathbf{v}(1:b-2)$$

Chapter 5

Performance analysis and DSP implementation

This chapter compares the computational complexity and performance of the algorithms developed in previous chapters with the block processing algorithm of [80]. We choose the algorithm [80] for comparison for several reasons. First, the algorithm of [80] is also deterministic. It has been shown elsewhere that stochastic algorithms suffer from serious performance degradations for short burst lengths, and hence a comparison with stochastic algorithms is less meaningful. Secondly, the algorithm of [80] also aims at direct symbol estimation and hence allows a straightforward performance comparison with the algorithms developed here in terms of bit error rate (performance differences between blind channel estimation and symbol estimation have been commented on in e.g. [133], see also section 2.5). Finally, the algorithm [80] is subspace based and employs block processing. Hence it allows to evaluate the reduction in computational complexity obtained by the use of adaptive processing.

Section 5.1 reviews the block processing algorithm for direct symbol recovery of [80]. This algorithm will serve as a reference for the comparison study in the rest of this chapter. Section 5.2 compares the computational complexity of the blind symbol estimation algorithms developed in previous chapters. It will be shown that the Viterbi algorithm has the highest complexity. The RTLS and Kalman filter algorithms have a lower and approximately equal complexity. Furthermore it is shown that the complexity of the Kalman filter and RTLS algorithms is lower than that of the block algorithm of [80] and that this complexity gap increases with the burst length. Section 5.3 studies the performance of the adaptive blind algorithms. It analyses the different parameter trade-offs and demonstrates that the adaptive algorithms are able to track time varying environments (which is not the case for the block processing

algorithm of [80]). Section 5.4 presents simulation results for the RTLS-D algorithm and shows that when the signal to noise ratio is sufficiently high, the algorithm provides results similar to the classical TLS solution at a much lower computational cost. Section 5.5 discusses a DSP implementation of the RTLS-QR algorithm and shows how the RTLS-QR algorithm can be integrated into a modem structure. Finally section 5.6 presents some conclusions.

5.1 Block Processing Algorithm [80]

This section reviews the block processing algorithm [80] for direct symbol recovery. In a first subsection we discuss the algorithm itself, in a second subsection we analyze its computational complexity.

5.1.1 Algorithm

If N again denotes the burst length, then

$$Y_{L+1|L+i} = \mathcal{H}_i \cdot X_{1|L+i} + N_{L+1|L+i}$$

where the number of columns of $Y_{L+1|L+i}$, $X_{1|L+i}$ and $N_{L+1|L+i}$ is equal to $j_b = N - i - L + 1$. The subscript b explicitly refers to the block processing algorithm. The matrix $X_{1|L+i}$ contains all unknown input symbols, and as for the adaptive algorithms presented in previous chapters, $\mathcal{H}_i$ is assumed to have full column rank.

The algorithm [80] is based on a singular value decomposition of $Y_{L+1|L+i}$ (which is a 'short-fat' matrix, i.e. with many more columns than rows)

$$\begin{aligned} Y_{L+1|L+i} &= U \cdot \Sigma \cdot V^H && (5.1) \\ &= \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & \Sigma^\perp \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}. && (5.2) \end{aligned}$$

The order of the system L is estimated from the singular value spectrum of $Y_{L+1|L+i}$. Next, a symbol sequence is sought that best matches the row space of $Y_{L+1|L+i}$, i.e.

$$\min_{\hat{X}_{1|L+i}} \|\hat{X}_{1|L+i} \cdot V^\perp\|^2 \qquad (5.3)$$

where $V^\perp$ is the null space of $Y_{L+1|L+i}$ ($Y_{L+1|L+i} \cdot V^\perp \approx 0$), which is extracted from the V-matrix of the SVD (columns of $V^\perp$ correspond to singular values below the 'noise threshold').

Imposing a Hankel structure on $\hat{X}_{1|L+i}$, equation 5.3 can be rewritten as:

$$\min_{\hat{\mathbf{x}}} \|\hat{\mathbf{x}}^T \cdot \begin{bmatrix} \boxed{V^\perp} & 0 & \cdots & 0 \\ 0 & \boxed{V^\perp} & \ddots & \\ & & & \boxed{V^\perp} \end{bmatrix} \|^2 \tag{5.4}$$

subject to some constraint to avoid the trivial solution. One may for example estimate the transmitted symbol sequence as a scaled version of the left singular vector corresponding to the smallest singular value of the above matrix.

5.1.2 Computational complexity

The *first step* in the block processing algorithm [80] requires the computation of the long right null space of $Y_{L+1|L+i}$, see equation 5.2. The SVD algorithm of [33] was used. For a burst length N we obtained the following flop count estimates, (define $j_b = N - (L+i) + 1$, $t = M \cdot i$):

$$16 \cdot j_b^2 \cdot t + 12 \cdot j_b \cdot t^2.$$

The *second step* in the algorithm [80], requires the computation of the short left null space of a $N \times ((i+L) \cdot (j_b - (i+L)))$ matrix, see equation 5.4 (actually one singular vector of this null space is required). The computational complexity can be lowered by first triangularizing the matrix before computing the short null space as in [18]. This leads to a computational complexity:

$$8 \cdot (i+L) \cdot (j_b - (i+L)) \cdot N^2 + 25 \cdot N^3.$$

Remark 5.1.1 The computational complexity of this algorithm can further be reduced by the use of SVD algorithms designed to compute a few singular vectors, e.g. the partial SVD (PSVD) algorithm of [134, Ch. 4]. However, the complexity will still contain quadratic and/or cubic terms in the burst length.

Remark 5.1.2 The algorithm of [133] is similar to the algorithm [80] but is based on a signal subspace intersection instead of a null space intersection and its complexity is linear in the burst length (as for the adaptive algorithms derived here). However, compared to the adaptive algorithms, memory storage requirements are higher due to the block nature of the algorithm [133] (see also remark 5.2.3) and the channel is assumed stationary during a burst.

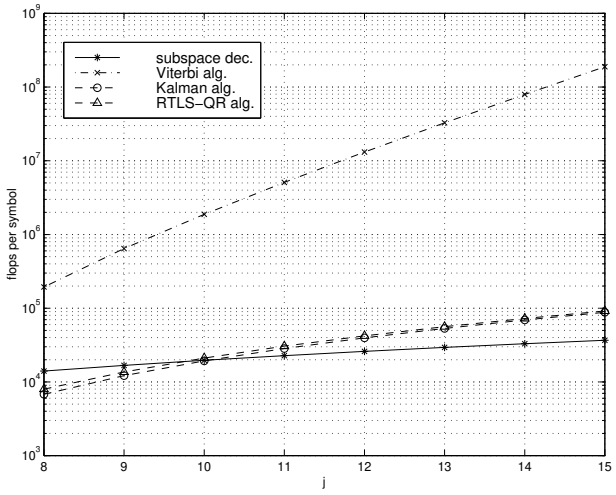


Figure 5.1: Complexity of the different adaptive algorithms for varying j, $b = j$, $P = 3$, $M = 4$, $i = 2$, $L = 4$, $N = 100$, BPSK.

5.2 Complexity comparison

This section compares the computational complexity of the different adaptive algorithms and of the block processing algorithm presented in the previous section. At the end of the section we also briefly discuss the memory requirements of the different approaches.

All adaptive algorithms have a complexity which is linear in the burst length, i.e. the complexity per decoded symbol is burst length independent. The following complexity formulas were derived in previous chapters:

$$\begin{aligned}
\text{Subspace decomposition:}\quad & (N-j+L+i)\cdot(16\cdot j^2\cdot t+12\cdot j\cdot t^2)\\
\text{Viterbi algorithm:}\quad & (N-j+1)\cdot(8\cdot j\cdot p+4\cdot p)\cdot\eta^j\\
\text{Kalman algorithm:}\quad & (N-j+1)\cdot((8\cdot p+15)\cdot j^2+2\cdot p\cdot j)\\
\text{RTLS-QR algorithm:}\quad & (N-j+1)\cdot((8\cdot p+11)\cdot j^2+2\cdot p\cdot j)+\\
& (N-b+1)\cdot P\cdot 8\cdot b^2\\
\text{Liu \& Xu [80]:}\quad & 16\cdot j_b^2\cdot t+12\cdot j_b\cdot t^2\\
& +8\cdot(i+L)\cdot(j_b-(i+L))\cdot N^2+25\cdot N^3.
\end{aligned}$$

Figure 5.1 depicts the complexity per decoded symbol for each of the adaptive algorithm steps for varying j. The burst length, channel length, smoothing factor and number of channels were chosen as in the simulations presented in

the next section. On the horizontal axis j is varied, the vertical axis shows the number of flops per decoded symbol. The figure shows that the complexity of the Viterbi algorithm greatly exceeds that of the other algorithms (e.g. for $j = 15$ it is more than three orders of magnitude more complex than the Kalman and RTLS-QR algorithm). The complexity of the Kalman and RTLS-QR algorithm is comparable. The complexity of their QR updating step is almost identical and is large compared to the complexity of respectively the back substitution and inverse iteration. The complexity of the subspace decomposition is higher than that of the Kalman and RTLS-QR algorithm only for small values of j. For larger j the QR updating step will dominate complexity.

Remark 5.2.1 Based on the robustness result of Theorem 4.2.1, it is possible to reduce the complexity of the QR updating and inverse iteration step in the RTLS-QR algorithm by not executing them in every iteration, but e.g. only once every two iteration steps. In that case we use only W_1,W_3,W_5, ... and the complexity of the QR updating and inverse iteration step is reduced by a factor 2 (the complexity of the subspace decomposition step remains unchanged). As explained in Theorem 4.2.1 the QR updating has to be executed at least once every $j - 1$ time steps. In the next section, we will present a simulation example that illustrates that this complexity reduction leads to a relatively small performance loss.

Figure 5.2 shows the complexity ratio of the block processing algorithm [80] versus the adaptive algorithms as a function of the burst length N. For the adaptive algorithms we add the complexity of the (common) subspace decomposition step and that of the actual symbol recovery step. Even for large burst lengths N, the complexity of the Viterbi algorithm exceeds that of the block processing algorithm. However, the Kalman and RTLS-QR algorithms are less complex than the block processing approach even for relatively short bursts. For a burst length ranging between 100 and 200 symbol periods, the complexity reduction varies between a factor 5 to 20. For all algorithms the complexity ratio increases for an increasing burst length since the complexity of the block processing algorithm contains quadratic and cubic terms in the burst length while the complexity of the adaptive algorithms is linear in the burst length.

Finally we investigate the influence of the parameter b on the complexity of the RTLS-QR algorithm. In figure 5.3 the parameter b is varied for $j = 15$ (all other parameters remain the same). The figure shows that the number of flops increases rather slowly with an increase of b: only the complexity of the inverse iteration step increases for increasing b.

Remark 5.2.2 The Kalman filter algorithm can also be modified to include a delay parameter b, as is done in [76] for a recursive least squares algorithm (similar to the Kalman filter algorithm presented here). Simulation results [76] show that for increasing b performance first increases, but afterwards decreases

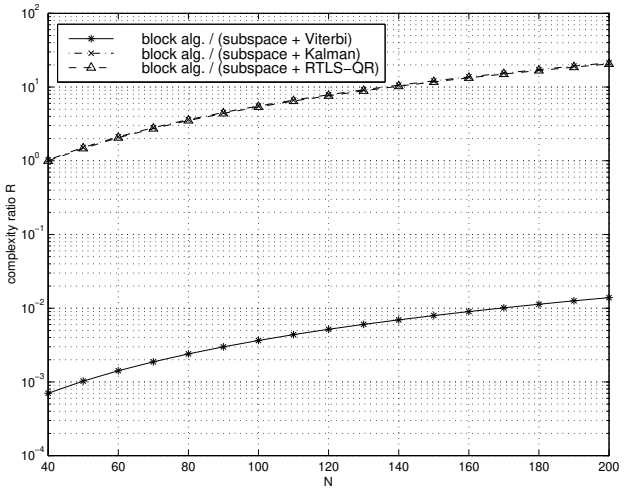


Figure 5.2: Complexity ratio of the block processing versus the adaptive algorithms as a function of the burst length N.

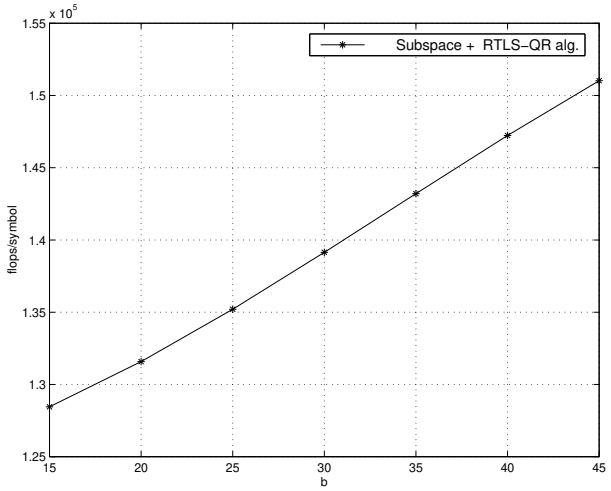


Figure 5.3: Complexity of the RTLS-QR algorithm for varying b, $j = 15$, $P = 3$, $M = 4$, $i = 2$, $L = 4$, $N = 100$.

for increasing b. The increase in performance is explained by the larger amount of information taken into account. The decrease is caused by the fact that as the data window enlarges, a monic constraint on one of the input symbols is not powerful enough to avoid the trivial null solution. I.e. as the data window

enlarges, a least squares problem is obtained with an 'almost zero' right hand side, hence the least squares solution is 'almost' the null solution, for more details we refer to [76]. Theoretically determining the optimal b for the RLS solution is an open problem. Further on we will not consider the introduction of a delay parameter in the Kalman filter algorithm.

Finally, we briefly discuss the memory requirements of the different algorithms. For each of the algorithms presented above we indicate the size of the largest matrices involved and give a rough estimate of the memory storage requirements based on the parameters used in the simulation examples of the next section.

The Viterbi trellis has η^{j-1} states and extends over $N-j+1$ time steps, so the number of memory locations required by the Viterbi algorithm for the parameters used in the next section ($\eta = 2, j = 12, N = 100$) will be of the order of $\eta^{j-1}N \approx 200000$ storage locations. For the Kalman filter algorithm ($p = (i+L)(j-(i+L)) = 54, j = 15$), at each iteration step a new W_k of size $p \times j$ has to be stored and the triangular R factor of size $(j-1) \times (j-1)$ needs to be tracked leading to approximately $(\frac{(j-1)^2}{2} + pj) \approx 900$ memory locations. For the RTLS-QR algorithm ($p = 54, j = 15, b = 30$) we need similarly $(\frac{b^2}{2} + pj) \approx 1300$ storage locations. In second step of the block processing algorithm of [80] ($N = 100, j_b = 95, i = 2, L = 4$) an SVD is computed from a $N \times (i+L)(j_b - (i+L))$ matrix leading to $N(i+L)(j_b - (i+L)) \approx 53000$ memory locations. These figures indicate that besides computationally complex, the Viterbi algorithm is also memory intensive. Contrary to the Viterbi and block processing algorithm of [80] the memory requirements of the Kalman and RTLS-QR algorithm are burst length independent, and large reductions of the memory requirements can be obtained.

Remark 5.2.3 For the block processing algorithm of [133] (see also remark 5.1.2) similar to the algorithm [80] the memory requirements increase linearly with the burst length (in contrast with the Kalman and RTLS-QR algorithm presented here).

5.3 Simulation results

In this section and section 5.4 we discuss the performance of the algorithms developed in previous chapters. In this section we compare the performance of the block processing algorithm [80] with that of the Viterbi algorithm, the Kalman filter algorithm and the RTLS-QR algorithm both in a time invariant and time varying environment. In section 5.4 we compare the performance of the RTLS-D algorithm and the non-recursive TLS and LS algorithms.

For the simulations in this section, bursts consist of $N = 100$ DBPSK symbols. Differential encoding avoids sign ambiguity problems after demodulation (see section 2.2). The SNR is defined as

$$SNR = 10 \cdot \log_{10} \frac{E\{\| \sum_{r=0}^{L} \mathbf{h_r} \cdot \mathbf{x}(k-r)\|_2^2\}}{E\{\|\mathbf{n_k}\|_2^2\}} \tag{5.5}$$

All simulations results were averaged over 500 Monte Carlo runs and in each run, a different symbol and white Gaussian noise realization was chosen. First we present simulation results for a time-invariant channel. We use the channel of length $L = 4$ with $M = 4$ outputs from [97]:

$$H = \begin{bmatrix} -0.049 + 0.359i & 0.482 - 0.569i & -0.556 + 0.587i & 1 \\ 0.443 - 0.0364i & 1 & 0.921 - 0.194i & 0.189 - 0.208i \\ -0.211 - 0.322i & -0.199 + 0.918i & 1 & -0.284 - 0.524i \\ 0.417 + 0.0030i & 1 & 0.873 + 0.145i & 0.285 + 0.309i \end{bmatrix} \begin{bmatrix} -0.171 + 0.061i \\ -0.087 - 0.054i \\ 0.136 - 0.19i \\ -0.049 + 0.161i \end{bmatrix}. \tag{5.6}$$

The smoothing factor i equals 2 and the number of iterations P in the inverse iteration step of the RTLS-QR algorithm was set to 3.

First we study the influence of the design parameter j on the bit error rate (BER) performance of the Kalman and RTLS-QR algorithm. We keep the SNR fixed at 14 dB, for the RTLS-QR algorithm we fix $b = j$. Figure 5.4 shows the results. For the RTLS-QR algorithm, performance increases with j and approaches the performance of the algorithm of [80] for large j. For the Kalman algorithm however, performance does not increase with increasing j. For larger j, the algorithm is more sensitive to error propagation (i.e. wrong symbol estimates tend to cause extra errors in subsequent iteration steps). The previous section showed that complexity also increases for increasing j. For the Kalman filter algorithm there does exist an optimal value for j beyond which performance does not increase. For the RTLS-QR algorithm, one can trade a larger j, more complexity, against better performance. Note that except for the lowest values of j the performance of the RTLS-QR algorithm exceeds that of the Kalman filter algorithm.

Next we study the influence of the parameter b on the performance of the RTLS-QR algorithm, we keep the parameter j fixed to 15 and vary the parameter b over the values 15, 30, 45. The SNR is again fixed at 14 dB. Figure 5.5 shows the results. On the horizontal axis the time index is shown, the vertical axis shows the number of bit errors that occurred on each time index in 500 runs. When b increases initial convergence problems of the algorithm can partially be alleviated. This is interesting since an increase of b only moderately increases the complexity of the algorithm (see the previous section). A few training symbols could further decrease the number of bit errors in the beginning of a burst.

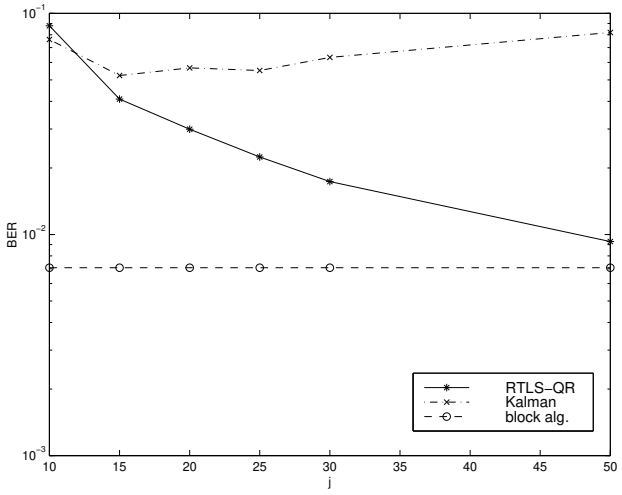


Figure 5.4: BER of the Kalman, the RTLS-QR algorithm and the algorithm [80] as a function of j (SNR 14 dB).

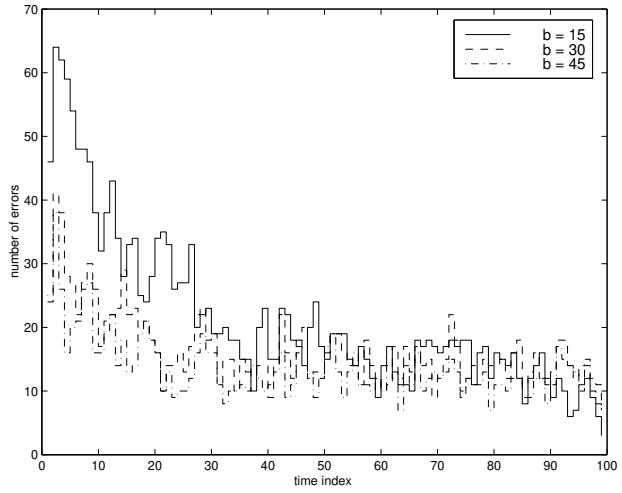


Figure 5.5: Accumulated number of bit errors over 500 runs as a function of the time index, for varying b for the RTLS-QR algorithm (SNR 14 dB).

Remark 5.3.1 Other simulations (results not displayed) have shown that the number of iterations P in the inverse iteration step has little influence on the performance of the RTLS-QR algorithm.

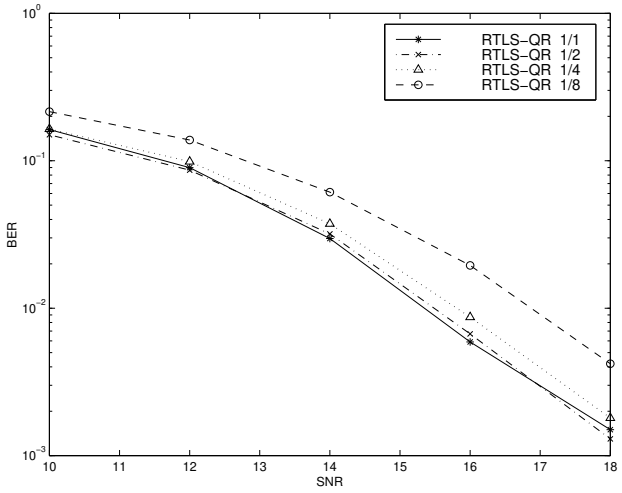


Figure 5.6: BER as a function of SNR for the RTLS-QR algorithm ($j = 15$, $b = 30$) when the QR updating and inverse iteration steps are executed every time step and once every 2, 4 and 8 time steps.

Next we illustrate the robustness of the RTLS-QR algorithm ($j = 15$ and $b = 30$). We compare the "standard" RTLS-QR algorithm with the case where the QR updating and inverse iteration are executed only once every second (use only W_1,W_3,...), fourth (use only W_1,W_5,...) and eighth iteration step (use only W_1,W_9,...). This allows to test the robustness of the RTLS-QR algorithm (see Theorem 4.2.1) but also allows to downscale the complexity of the QR updating and inverse iteration steps (remark 5.2.1). Figure 5.6 shows that the complexity of the QR updating step can be downscaled with a factor two at the expense of only a very limited performance loss. Further reducing the updating rate leads to a bigger performance loss but overall the RTLS-QR algorithm shows good robustness against the rank conditions of Theorem 3.1.1 (as was predicted by Theorem 4.2.1).

Figure 5.7 compares the performance of the algorithm [80] with the RTLS-QR algorithm ($j = 15$ and $b = 30$), the Kalman filter algorithm ($j = 15$) and the Viterbi algorithm ($j = 12$) for varying SNR. Figure 5.7 shows that the Viterbi algorithm has the best performance. So although it does not exploit the time invariance of the channel over the burst length, as does the block processing algorithm [80], the finite alphabet constraint is more powerful than the quadratic constraint in mitigating noise effects. The RTLS-QR and Kalman algorithms perform worse than the block processing algorithm but are computationally cheaper. Finally note that the performance of the RTLS-QR algorithm again exceeds that of the Kalman filter algorithm.

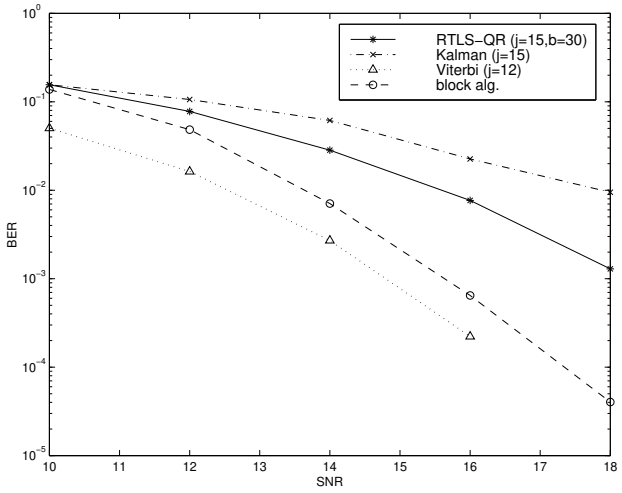


Figure 5.7: BER as a function of SNR for the RTLS-QR algorithm ($j = 15$, $b = 30$), the Kalman filter algorithm ($j = 15$), the Viterbi algorithm ($j = 12$) and the block algorithm [80].

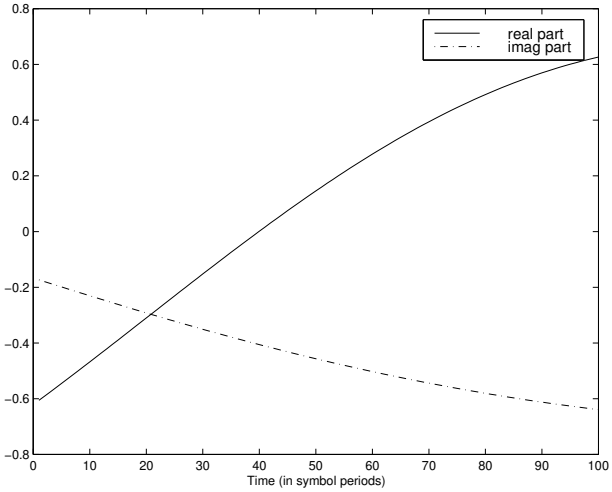


Figure 5.8: Typical time variation of a tap (100 symbol periods).

Next, we show some simulation results for a *time varying* channel. The simulation scenario was taken from [57]. Figure 5.8 shows the typical time variation of a channel tap over one burst. The full line represents the real part, the dotted

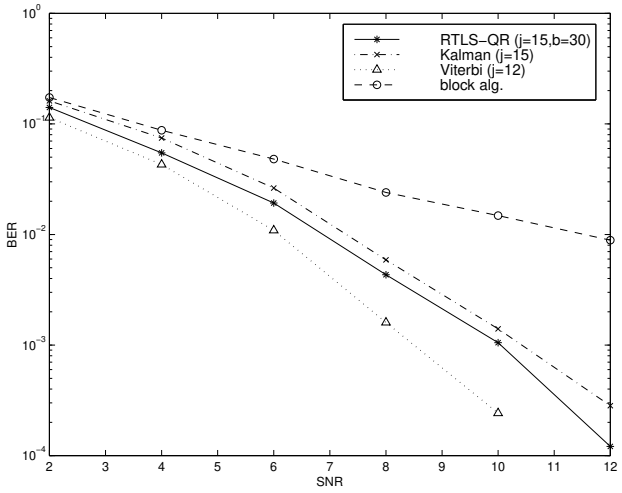


Figure 5.9: BER for a time varying channel as a function of SNR for the RTLS-QR algorithm ($j = 15$, $b = 30$), the Kalman filter algorithm ($j = 15$), the Viterbi algorithm ($j = 12$) and the block algorithm [80].

line the imaginary part. The taps were generated by filtering white Gaussian noise through a second order resonance filter with resonance frequency 70 Hz followed by a low pass filter. The Doppler frequency of 70 Hz corresponds to a mobile user moving with a velocity of approximately 83 km/h, at a carrier frequency of 900 MHz. The baseband sampling frequency was assumed to be 27 kHz.

Again, we transmit bursts of 100 DBPSK symbols. It should be noted that for a time varying channel, the block processing algorithm [80] can split the burst into different parts (turning it into a semi-adaptive algorithm), and operate on these parts separately, hence reducing the time variation of the channel over one part. But this induces additional boundary conditions at the beginning and end of each part (which still contain some equations in symbols from the previous/next part) and will not be investigated here. The channel length $L = 2$ and $M = 4$. We compare the same algorithms as in the previous simulation. Figure 5.9 depicts the results. Not surprisingly, the performance of the Viterbi algorithm is superior to that of the other algorithms. However, now the performance of the RTLS-QR and Kalman filter algorithm also exceeds that of the block processing algorithm.

In conclusion, the complexity of the Viterbi algorithm is high but its performance exceeds that of the block processing algorithm [80] even when the channel is time-invariant (see figures 5.7 and 5.9). The finite alphabet constraint is

thus a powerful constraint for symbol recovery. The Kalman filter and RTLS algorithms form attractive alternatives for the block processing algorithm [80] from the viewpoint of computational complexity. For burst lengths between 100 and 200 symbol periods, the computational complexity was reduced by a factor 5 to 20, see figure 5.2. We have further shown that the memory requirements of the Kalman and RTLS algorithms are burst length independent, which may lead to considerable memory storage reductions. Moreover if the environment is highly time varying, the algorithms are also interesting from a performance perspective (figure 5.9). Simulation results have also shown that the RTLS-QR algorithm should be preferred over the Kalman filter algorithm. In almost all scenarios it has a superior performance and this at a marginally higher computational cost (see figure 5.1). In the next chapter when we extend our results to the multi-user scenario, we will discard the monic constraint and develop solutions only under a finite alphabet and quadratic non-triviality constraint.

5.4 RTLS-D algorithm: complexity and performance

In this section we analyze complexity and performance of the RTLS-D algorithm. First we compare the computational savings of the RTLS-D algorithm with respect to a straightforward calculation of the TLS solution. From section 5.1.2 and section 4.4.3, we have the following complexity formulas:

$$\begin{aligned} \text{TLS algorithm:} \quad & 8 \cdot M \cdot (N-L) \cdot (N+1)^2 + 25 \cdot (N+1)^3 \\ \text{RTLS-D algorithm:} \quad & (N-L) \cdot (8 \cdot M \cdot L^2 + 34 \cdot M \cdot L + 11 \cdot L^2) + \\ & (N-b+2) \cdot 8 \cdot P \cdot b^2. \end{aligned}$$

For the complexity of the TLS algorithm, remark 5.1.1 holds again. Figure 5.10 compares the complexity per estimated symbol for these two algorithms for the parameters used in the simulation scenario presented below. The complexity of the RTLS-D algorithm is quasi independent of the burst length while the complexity of the TLS algorithm increases for increasing burst length.

Next we compare the performance of the RTLS-D algorithm with the non-recursive TLS and least squares (LS) solutions. The simulation parameters are depicted in table 5.1. The input sequence $\mathbf{x}$ has a unit modulus (i.e. $|\mathbf{x}(l)| = 1, l = 1, \ldots, N$), but a random phase. The channel is random complex valued of length $L = 4$. The SNR is defined as in equation 5.5. We add complex white Gaussian noise of variance $\sigma^2 = \frac{E\{\|\mathbf{n_k}\|_2^2\}}{M}$ to both the channel and the outputs. The different algorithms were run using the noisy output data and channel estimates. The results were averaged over 500 Monte-Carlo runs. The

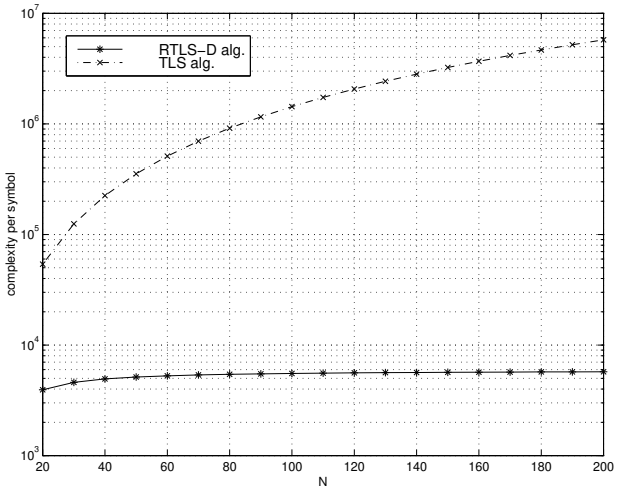


Figure 5.10: Complexity per estimated symbol for the RTLS-D and the TLS algorithm as a function of the burst length N, $b = 15$, $M = 15$, $P = 1$ and $L = 4$.

M	L	b	N	P
10-15-20	4	6-15	100	1

Table 5.1: Simulation parameters: M denotes the number of channels, L is the channel length, b is the window size, N is the burst length and P is the number of iterations in the inverse iteration step.

performance index used is the mean squared error, which is computed as:

$$MSE = E\{\sum_{k=1}^{N}(\hat{\mathbf{x}}(k) - \mathbf{x}(k))^2/N\}.$$

We study the influence of M (the number of channels) and b (the window length) on performance. Figure 5.11 shows the MSE of the algorithms for varying SNR and for $M = 10, 15, 20$. For the RTLS-D algorithm b is 6 or 15. We compare performance with the block TLS and LS algorithms. The figures show that except for the combination ($M = 10$, SNR=10 dB) the performance of the TLS algorithm exceeds that of the LS solution. The performance of the RTLS-D algorithm approaches that of the TLS algorithm well for an SNR equal or higher than 15 dB. For an SNR of 10 dB the performance of the RTLS-D algorithm is worse than that of the TLS algorithm (and even worse than that of the LS solution) especially if the parameter $b = 6$. The figures show that an increase of M enlarges the performance gap between the (R)TLS and LS

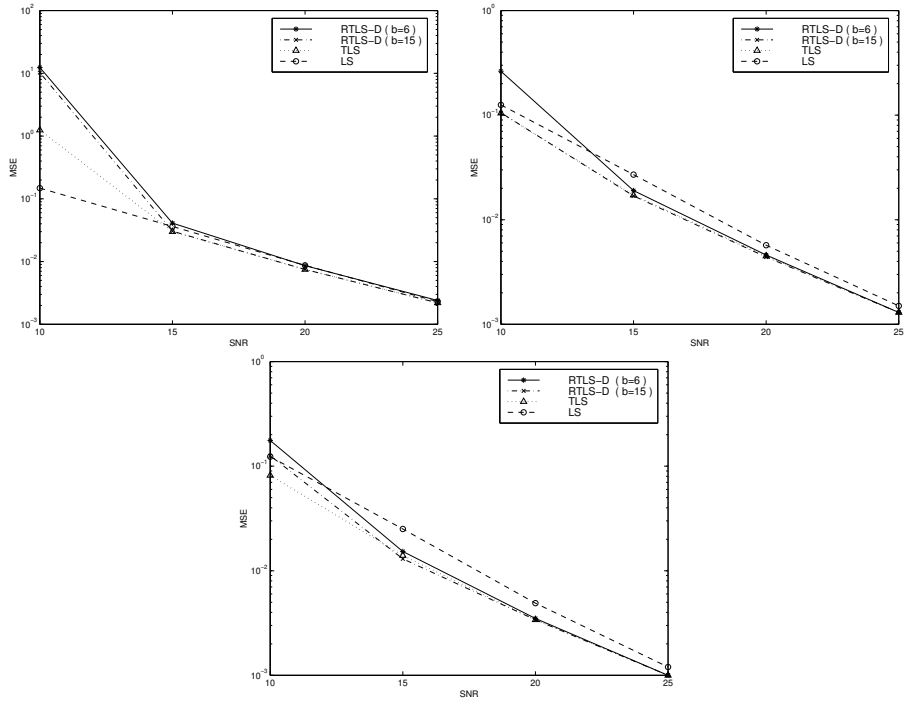


Figure 5.11: MSE for varying SNR $M = 10$ (upper left) $M = 15$ (upper right) and $M = 20$ (bottom).

algorithms. Other simulations (results not depicted) showed that an increase of P (number of iterations in the inverse iteration step of the RTLS-D algorithm) has very little influence on performance. When the SNR is decreased below 10 dB, the LS solution provided more accurate results than both the TLS/RTLS-D algorithms (see also remark 4.1.2).

In conclusion, the performance of the RTLS-D algorithm closely approximates that of the TLS algorithm provided the SNR is sufficiently high and this at a much lower computational complexity. Hence the RTLS-D algorithm might be useful in deconvolution problems whenever a TLS type approach is expected to give better results than a LS type approach.

5.5 DSP implementation

In the previous chapters we presented a number of single-user adaptive blind equalization algorithms. Their performance was investigated through com-

puter simulations. These showed that the RTLS algorithms balance complexity against performance. This section demonstrates how the RTLS-QR algorithm can be integrated into a complete modem structure, which we implemented on a DSP platform.

Many wireless test-beds have already been built for spatio-temporal channel characterization and algorithm evaluation. Examples are the TSUNAMI II stand-alone test-bed [93], the RACE Real Time Testbed [92], the Ericsson GSM testbed [7], the Lucent Technologies and ATT Labs testbed [23] and the Stanford TDMA wireless testbed [108]. The TSUNAMI II testbed for example operates in uplink mode and uses an eight antenna base-station receiver and two mobile units for transmitting modified GSM multi-frame TDMA bursts. Our DSP implementation is less ambitious but aims at an integration of the blind equalization algorithm as a building block in the transmitter-receiver chain. An exact simulation of the wireless environment and fulfilling all GSM burst format rules are not aimed at in the first place.

Nevertheless, a DSP implementation reveals some algorithmic aspects which may remain undiscovered in a straightforward MATLAB implementation.

When deriving the complex baseband data model in section 2.1 we assumed the carrier frequency to be known perfectly. However, in practice this requirement might only be fulfilled approximately. Transmitter and receiver clock are based on oscillation crystals which are specified up to a tolerance. Hence when the passband signal is downconverted at the receiver side, one obtains

$$y(t) \cdot e^{2\pi(f_{ct}-f_{cr})t}$$

where f_{ct} and f_{cr} represent the carrier frequency at transmitter, respectively receiver side. The complex baseband signal at the receiver will be a rotated version of the true baseband signal, and furthermore the rotation varies with time. Therefore, if no correction is applied for this frequency offset, it must be taken into account in a performance analysis. Traditional methods based on training sequences are not insensitive to errors in the carrier frequency estimate. For the adaptive algorithms developed in previous chapters the rotation factor has to be small only over the window length of the algorithm, a condition which will always be met in practice.

Another problem is the detection of the beginning of a burst, i.e. transmitter and receiver should be synchronized such that the receiver can estimate the beginning of a burst to within a fraction of the symbol period T. In GSM the training sequence serves the purpose of both channel estimation and synchronization. In blind methods a few training symbols could be used just for synchronization purposes. However, this issue is usually overlooked when testing blind algorithms. The DSP implementation revealed that this problem is indeed important.

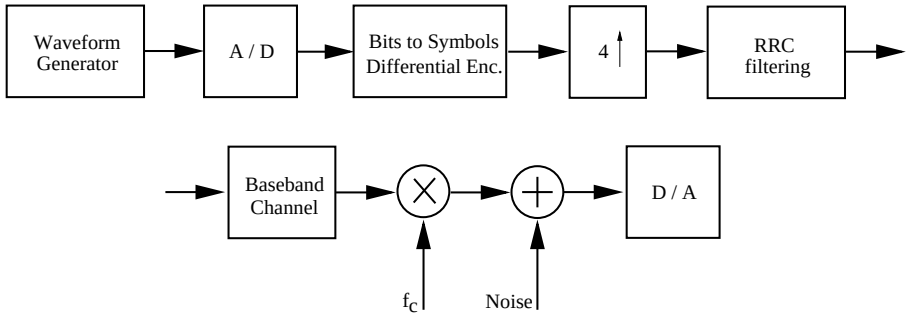


Figure 5.12: Transmitter block diagram.

A third problem is the detection of the symbol period T. Due to tolerances on transmitter and receiver crystals, transmitter and receiver will sample data at slightly different paces, i.e. T_t and T_r. Assume $T_r > T_t$, then there exists an s such that:

$$s \cdot T_r \geq T_t \cdot (s+1).$$

The receiver assumes only s symbols have been transmitted while actually $(s+1)$ symbols have been transmitted. This might cause word synchronization problems. If no adjustment of the symbol period is done at the receiver side, one should keep bursts short enough such that the above situation does not occur. Secondly, word synchronization bits could be periodically inserted in the transmitted bit stream to reinitialize synchronization.

Further a DSP implementation allows to simulate sampling artifacts (e.g. a DC offset of A/D converters) and (timing) phase offsets. From the above examples it should be clear that a DSP implementation reveals algorithmic aspects which indeed might be overlooked in a straightforward MATLAB implementation. Furthermore simulation speed is increased by a few orders of magnitude with respect to a MATLAB simulation.

The outline of this section is as follows. Section 5.5.1 gives an overview of the transmitter and receiver structure by means of a block diagram. Section 5.5.2 discusses the actual DSP implementation. Finally section 5.5.3 concludes with a practical simulation example.

5.5.1 Transmitter and receiver

The block diagrams of figures 5.12 and 5.13 give an overview of the transmitter and receiver structure. We shortly describe the function of each of the building blocks.

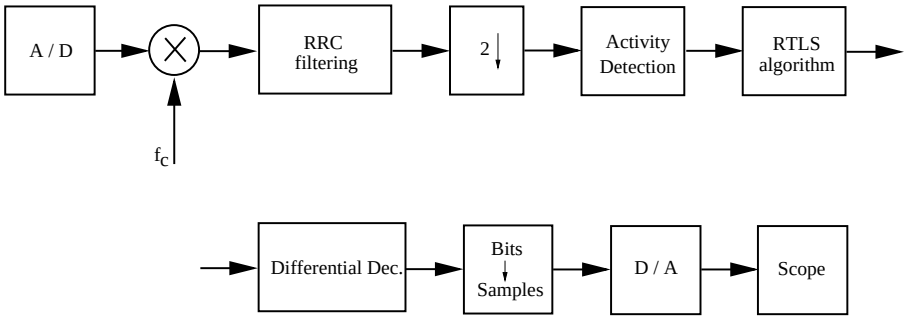


Figure 5.13: Receiver block diagram.

At the transmitter side, a waveform generator produces the signal to be transmitted. This signal is sampled by the DSP's A/D converter and each sample is quantized into 5 bits. The bits are subsequently mapped to QPSK symbols. Gray encoding is used such that adjacent phases differ by one binary digit (see also section 2.2). A symbol error in the demodulation will then most likely only result in a single bit error. Subsequently, data symbols are differentially encoded in order to avoid phase recovery problems at the receiver side (see section 2.2). An example of this process is given below:

Example 5.5.1

	1	0	1	1	1	0	0	0	0	1	binary input
	$-i$		-1		$-i$		1		i		QPSK symbols
1	$-i$		i		1		1		i		diff. encoded QPSK symbols.

□

The differentially encoded QPSK symbols are upsampled and root raised cosine (RRC) filtered. The root raised cosine filter together with its matched filter (in the receiver) form the raised cosine filter:

$$P_{RRC}(f) \cdot P^*_{RRC}(f) = P_{RC}(f)$$

and

$$\begin{aligned} p_{RRC}(t) &= \frac{\sin(\frac{\pi t}{T}(1-\alpha)) + \frac{4\alpha t}{T}\cos(\frac{\pi t}{T}(1+\alpha))}{2 \cdot \frac{\pi t}{T}(1-(\frac{4\alpha t}{T})^2)} \\ p_{RC}(t) &= \frac{\cos(\pi\alpha\frac{t}{T})}{1-(2\alpha\frac{t}{T})^2} \cdot \frac{sin(\pi\frac{t}{T})}{\pi\frac{t}{T}}. \end{aligned}$$

Figure 5.14 shows the time and frequency domain response of the raised cosine filter. The rolloff factor α determines the bandwidth of the filter. The smaller α,

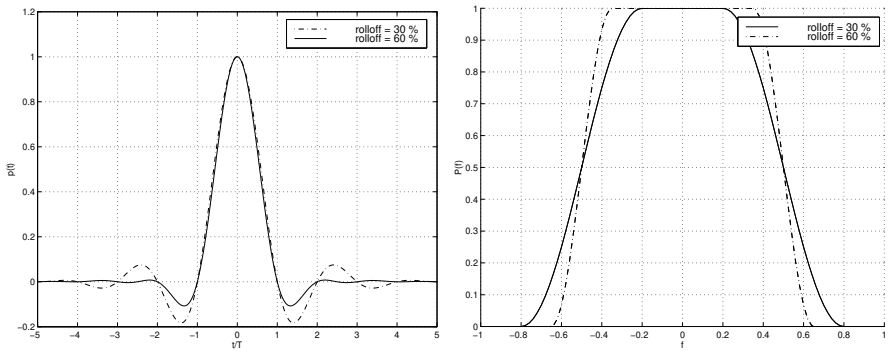


Figure 5.14: Time domain (left) and frequency domain (right) response of a raised cosine (RC) filter. The rolloff factor determines the bandwidth of the filter and the magnitude of the fading tails in the time domain. In both figures the symbol period T is normalized to 1.

the smaller the bandwidth, but the more prominent the fading tails in the time domain. These fading tails can cause order detection problems and performance degradation for subspace algorithms.

The baseband channel mimics the multi-path components of the wireless environment. The complex baseband signal is then upconverted to a real passband signal, and noise is added to the signal. Finally the signal is sent to the D/A converter which puts the signal on coax.

The receiver architecture (figure 5.13) is quite similar to the transmitter architecture. The A/D converter samples the signal at 4 times the symbol rate. This signal is then downconverted to baseband and root raised cosine filtered (matched filtering with respect to the transmitter filter). The resulting signal is downsampled with a factor 2, so we retain 2 samples per transmitted symbol (M=2). The activity detection block that follows, detects the start of a burst. Once the beginning of the burst is detected the actual blind equalization algorithm can be executed. It is followed by a differential decoding operation. The symbols are then mapped to bits and finally the bits are grouped into samples, ready for transmission to the D/A converter. The final output can then be observed on a scope.

Remark 5.5.2 Obviously, for the implementation of other equalizer algorithms, only the RTLS-QR algorithm has to be removed and replaced.

Remark 5.5.3 So far we have defined "blind" as a channel estimation/equalization without training sequence. The receiver could however be augmented

with building blocks for blind detection of an unknown baud rate, blind detection of an unknown carrier frequency or blind detection of an unknown constellation, see also [123].

5.5.2 DSP implementation

Two DSP boards, each placed in a VME-rack, represent respectively transmitter and receiver. These boards are accessible through a local network via a Sun Sparc station. Transmitter and receiver are each implemented on a 25-MIPS TMS320C44, clocked at 50 MHz.

All algorithms were first developed in MATLAB, later converted to C and finally ported to DSP with the aid of the parallel C compiler of 3L Ltd. [86]. The results obtained on DSP equal the outcome of the MATLAB programs up to machine precision. For the implementation of the subspace decomposition step in the RTLS-QR algorithm, a modified version of the adaptive SVD algorithm [95] was implemented, see also section 3.1. Table 5.2 depicts the parameters that are used in the current demo setup. The channel order is variable and is

M	i	j	b	L
2	2	8	8	1 or 2

Table 5.2: Parameters for the demo setup.

estimated at runtime by the algorithm from the SVD spectrum computed in the subspace decomposition step. The receiver is updated whenever the A/D buffer of the receiver contains 4 new samples. In this way the execution time of all update cycles is approximately equal. For a smaller number of input samples per update this would not be the case, due to the multi rate structure of the receiver, and the M=2 new samples required for each SVD update. When the program was divided over three processors the maximum achievable symbol rate was approximately 1.8 ksymbol/sec [28]. In the near future, transmitter and receiver will be augmented with an antenna front-end, such that wireless simulations become possible. The aim is to send data between transmitter and receiver at a realistic high speed and to do the signal processing at the receiver at a lower rate. The DSP setup will than act as a data acquisition system rather than a real time prototype.

5.5.3 Simulation example

This section illustrates the operation of the DSP program by means of a simulation example. The baseband channel consists of the three path model shown

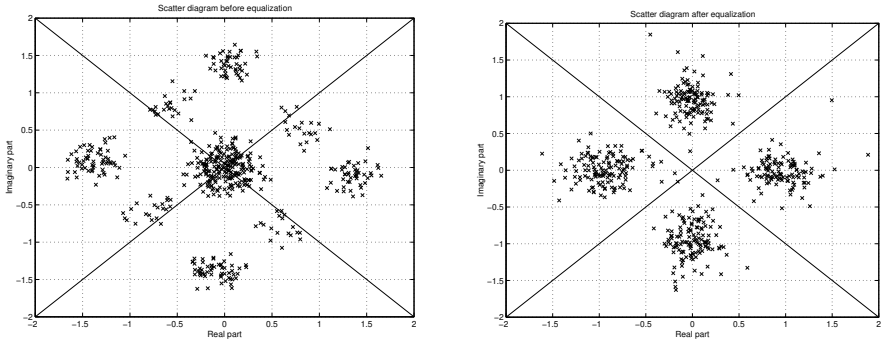


Figure 5.15: Scatter diagram before (left) and after equalization (right).

in table 5.3. Figure 5.15 shows a scatter diagram before and after equaliza-

path	gain (in dB)	delay (in symbol periods)
1	0	0
2	-2.5	0.50
3	-6	0.75

Table 5.3: Characteristics of the multi-path.

tion. The RTLS-QR algorithm clearly has a favorable impact on the QPSK scatter diagram. Before equalization a lot of symbol estimates are near the decision boundaries (full lines), while after equalization the estimates are clustered around the four correct symbol values: $\{-1, 1-i, i\}$. The reconstructed input signals are shown in figure 5.16. On the left hand side the sinusoidal nature of the input signal has completely disappeared while the signal is well reconstructed on the right hand side (to within the quantization error that was introduced at the transmitter side).

5.6 Conclusions

In this chapter we have compared the complexity and performance of all algorithms developed in previous chapters. We have shown that the complexity of the subspace + Viterbi algorithm is high, but that the algorithm performs very well (even better than the block processing algorithm of [80]), both in a time-invariant and a time varying environment. In chapter 6 we will show how this algorithm is generalized to the multi-user scenario. The RTLS-QR and Kalman filter algorithm are viable alternatives to lower the computational

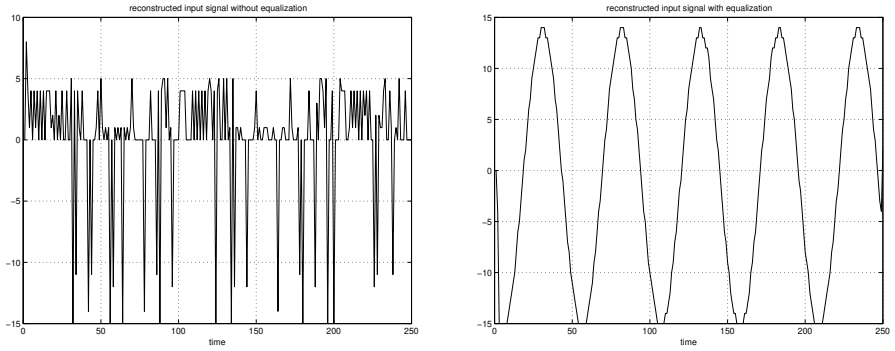


Figure 5.16: Reconstructed input without (left) and with equalization (right).

complexity and memory requirements of the block processing algorithm of [80] (and the Viterbi algorithm): for burst lengths between 100 and 200 symbol periods the computational complexity was reduced by a factor 5 to 20. We have further demonstrated that these algorithms are able to track a time-varying environment. The RTLS-QR algorithm has been shown to have superior performance compared to the Kalman filter algorithm while the two algorithms have approximately the same computational cost.

We have also investigated the complexity and performance of the RTLS-D algorithm. Simulation results have shown that its performance approaches that of the block TLS algorithm at a sufficiently high SNR and this at a much lower computational cost.

Finally in the last section of this chapter we have presented a DSP implementation of the RTLS-QR algorithm.

Part II

Multi-user blind channel equalization algorithms

In the previous chapters we have presented a number of adaptive single user blind symbol estimation algorithms. In chapter 6 and chapter 7, we show how these ideas can be generalized to solve multi-user blind equalization problems. Multi-user blind equalization generalizes single user blind equalization in the sense that two problems are solved. As for the single user scenario *inter symbol interference (ISI)* corrupts the received messages. Secondly when multiple users simultaneously occupy the same frequency band, only a linear mixture of these signals can be observed at the receiver. This is called *multiple access interference (MAI)* or *co-channel interference (CCI)*.

Multi-user blind equalization then refers to digital signal processing algorithms which remove both ISI and MAI. *Blind signal separation* refers to signal processing techniques which remove MAI from an instantaneous linear mixture of signals. Both techniques are based only on the output signals at the receiver, possibly combined with some deterministic or stochastic property of the input signals, but without relying on training sequences.

Most multi-user second order blind equalization algorithms developed up till now, e.g. [81] and [133], are two step procedures. The first step removes ISI and transforms the multi-user blind equalization problem into a blind signal separation problem. This can be done by directly estimating an instantaneous linear mixture of the signals [81, 133] or by first estimating the channel matrix (up to a mixing matrix, see also section 2.7), and then applying channel inversion [3]. The second step removes MAI and reconstructs the symbol sequences of the distinct users. Several algorithms have been proposed to accomplish MAI removal. In [115], two iterative algorithms ILSP and ILSE are proposed, which recover the input signals based on their finite alphabet property. These algorithms need good initializations and may converge to local minima. Non-iterative methods can be found in [130] (for constant modulus signals) and [6, 128] (for BPSK signals). Although near-optimal these approaches are computationally expensive. These two step algorithms require an accurate estimate of the channel lengths of individual users (the estimation usually entails a cumbersome iterative procedure, see e.g. [81]) and depend on an accurate system order estimate.

The approach taken in chapter 6 and 7 differs from previous methods in that it applies *multi-user coding* at the transmitter side to facilitate decoding at the receiver. The application of coding at the transmitter side "automatically" removes MAI, and hence immediately allows to solve d one-dimensional problems in parallel (for d users) instead of solving (iteratively) a d-dimensional problem. This leads to computationally simple, non-iterative algorithms without initialization or convergence problems. Additionally, coding improves the robustness of the algorithms to order detection problems, i.e. the proposed algorithms still provide reliable parameter estimates even if the system order is largely overestimated. Coding further enables parameter estimation without requiring the exact knowledge of the channel lengths of the individual users,

instead only a lower bound on the channel length needs to be known. Order detection and channel length estimation robustness are features usually shared by stochastic algorithms (see section 2.5). The algorithms developed here are deterministic and besides the robustness properties they exhibit the finite sample convergence property: i.e. in the absence of noise the algorithms provide perfect parameters estimates with a finite number of data.

By applying different coding strategies in the algorithms of chapter 6 and 7 we obtain algorithms with different trade-offs. As will become clear we can trade transmission efficiency (chapter 6 and section 7.2) for a restriction on the modulation format (section 7.3) for hardware (extra transmitter and receiver antennas, section 7.4). Rather than developing the "best" algorithm for a specific application, we leave the reader with a battery of techniques that be applied (and combined) under different performance and complexity constraints.

Remark 5.6.1 Coding at the transmitter side has already been proposed in the context of transmitter induced cyclostationarity, see e.g. [19, 49], for improving the robustness of single user identification algorithms. In [19] the input signal is multiplied by a superposition of complex exponentials. This implies modulus variations of the transmitted signal, which is not always desirable. In [49], the input sequence is multiplied by a deterministic and periodic sequence. However, there are two important differences with the algorithms developed here. First, the coding schemes presented in chapter 6 and chapter 7 are intended for multi-user situations while the codes of [19, 49] are used in single user scenarios. Secondly, both the algorithms of [19] and [49] are stochastic algorithms based on statistical cyclostationarity properties whereas the focus of chapter 6 and chapter 7 is on deterministic algorithms (which can be applied to much smaller data blocks).

Remark 5.6.2 When the multi-user coding schemes developed in chapter 6 and chapter 7 would be applied to currently employed (single-user) communication protocols, they should not be considered as replacing existing coding schemes. Rather they form an extra (outer) layer of coding, specifically designed to enhance multi-user detection. The "classical" (inner layer of) coding could then still provide error protection/correction.

Simulation results in the previous chapter have shown that (in the single user case) a finite alphabet non-triviality constraint provides good performance and that a quadratic non-triviality constraint might be preferred over a monic constraint. Therefore we will only consider the finite alphabet and quadratic constraint from now on and discard the monic constraint.

In chapter 6 it is shown how the single user subspace + Viterbi algorithm

presented in chapter 3 can be extended to solve multi-user blind equalization problems under a finite alphabet non-triviality constraint.

In chapter 7 we develop a number of blind signal separation algorithms based on linear coding which estimate the transmitted symbols under a quadratic non-triviality constraint.

Chapter 6

A subspace + Viterbi algorithm for multi-user blind equalization

In this chapter we return to the subspace + Viterbi algorithm presented in chapter 3 and show how it can be generalized to the multi-user scenario. In chapter 7 we demonstrate how the (R)TLS algorithms of chapter 4 can be adapted to the multi-user scenario.

The algorithm presented in this chapter comprises two steps. As for the single user Viterbi algorithm of chapter 3, the first step hinges on an adaptive subspace decomposition for a (virtual) channel identification type operation. In the second step the Viterbi algorithm is used for subsequent symbol detection. *Unlike other blind multi-user detection schemes that have recently been proposed in literature, the present algorithm removes multiple access interference (MAI) by exploiting the finite alphabet property of the input signals together with channel coding redundancy. The latter is believed to be a crucial new ingredient for performance in the context of MAI suppression. It will be demonstrated that a proper selection of the codes enhances the removal of MAI by providing an initial separation of the symbol sequences and increases robustness against channel length estimation problems. Furthermore, the algorithm is adaptive, which allows time varying channels to be tracked.*

In section 6.1 we present the multi-user subspace+Viterbi algorithm for the case where all users have equal channel lengths. In section 6.2 we generalize the results of section 6.1 to the scenario where users have different channel lengths. Finally in section 6.3 some conclusions are drawn. A performance

analysis is delayed till chapter 8 when all multi-user algorithms will have been presented.

6.1 Viterbi algorithm

We start from the multi-user data model of equation 2.13, which is repeated here for convenience

$$Y_{k|k+i-1} = \underbrace{\begin{bmatrix} H & 0 & 0 & \cdots \\ 0 & H & 0 & \cdots \\ & & \ddots & \\ 0 & 0 & \cdots & H \end{bmatrix}}_{\mathcal{H}_i} \cdot X_{k-L|k+i-1} \tag{6.1}$$

with,

$$Y_{k|k+i-1} = \begin{bmatrix} \mathbf{y_k} & \cdots & \mathbf{y_{k+j-1}} \\ \mathbf{y_{k+1}} & \cdots & \mathbf{y_{k+j}} \\ \vdots & & \vdots \\ \mathbf{y_{k+i-1}} & \cdots & \mathbf{y_{k+i+j-2}} \end{bmatrix}, X_{k-L|k+i-1} = \begin{bmatrix} \mathbf{x_{k-L}} & \cdots & \mathbf{x_{k-L+j-1}} \\ \mathbf{x_{k-L+1}} & \cdots & \mathbf{x_{k-L+j}} \\ \vdots & & \vdots \\ \mathbf{x_{k+i-1}} & \cdots & \mathbf{x_{k+i+j-2}} \end{bmatrix}.$$

We also define the $d \times N$ matrix X where each row contains the complete symbol sequence of one user:

$$X = \begin{bmatrix} \mathbf{x}^{(1)} \\ \vdots \\ \mathbf{x}^{(d)} \end{bmatrix}. \tag{6.2}$$

The first step of the algorithm is identical to that of the single user subspace + Viterbi algorithm, i.e. we compute an SVD in each symbol period.

$$\begin{aligned} Y_{k|k+i-1} &= U_k \cdot \Sigma_k \cdot V_k^H \\ &= \begin{bmatrix} U_k^s & U_k^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma_k^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V_k^{sH} \\ V_k^{\perp H} \end{bmatrix} \\ X_{k-L|k+i-1} \cdot V_k^\perp &= 0 \end{aligned}$$

where it is assumed that $\mathcal{H}_i$ is of full column rank and hence $M \cdot i \geq d \cdot (L+i)$ (see also section 2.7).

In the multi-user scenario both the number of users and their channel lengths have to be estimated from the spectrum of singular values Σ_k. Without noise, there are $r_Y = d \cdot i + \sum_{l=1}^{d} L_l$ non-zero singular values. With noise the number of

large singular values has to be determined from the gap in the SVD spectrum. As explained in [133], the number of users d can be detected by increasing the smoothing factor i by 1. The rank of $Y_{k|k+i-1}$, i.e. r_Y, then increases by d. This mechanism works well even when the noise level is high, since it is independent of the observable channel length. The property further holds even if users have different channel lengths. Further on, we will assume that the number of cochannel users is known or has been correctly estimated using the above mechanism. If all users have equal channel lengths, L can be estimated as $(\frac{r_Y}{d} - i)$. For the case where users have unequal channel lengths we refer to section 6.2. In section 6.2 we will demonstrate that if appropriate coding is applied, only a lower bound on the channel lengths is needed: the symbol sequences of the different users can be identified without estimating their exact channel lengths.

Next we construct a set of homogeneous equations in the unknown symbols in the same way as for the single user algorithm:

$$X(:, k : k+j-1) \cdot W_k = 0 \tag{6.3}$$

$$W_k \stackrel{\triangle}{=} \left[\begin{array}{cccccc} V_{k-i+1}^{\perp} & \cdots & V_k^{\perp} & \cdots & V_{k+L}^{\perp} \end{array} \right]. \tag{6.4}$$

Remark 6.1.1 Note that we define W_k without transpose operator (contrary to chapter 3) for simplicity of notation further on.

In the single user scenario, W_k defined a one-dimensional null space, while in the multi-user case it defines a d-dimensional null space. The theorem below identifies a *sufficient* condition under which W_k determines the rows of $X(:, k : k+j-1)$ in the case of BPSK modulation.

Theorem 6.1.2 *If*

1. *the matrices* $X_{k-L-i+1|k+1}, \cdots, X_{k-L|k+i}, \cdots, X_{k-1|k+i+L-1}$ *are of full row rank (which implies* $j \geq d \cdot (L+i+1)$ *)*
2. *the columns of* $X(:, k : k+j-1)$ *contain all* $2^{(d-1)}$ *possible symbol sequences up to a sign ambiguity (which implies* $j \geq 2^{d-1}$*)*

then the d *rows of* $X(:, k : k+j-1)$ *are the unique solutions (up to a sign ambiguity) belonging to the* $\{-1, 1\}$*-alphabet that lie in the* d*-dimensional null space defined by:*

$$S \cdot W_k = 0. \tag{6.5}$$

Proof: The first rank condition assures that W_k has a d-dimensional left null space and that $X(:, k : k+j-1)$ belongs to that null space. The proof is a

straightforward extension of the single user case (Theorem 3.1.1). The second condition ensures that there are only d symbol sequences (up to a sign ambiguity) that belong to the $\{-1, 1\}$-alphabet and lie in the subspace generated by equation 6.5, see [115]. □

Remark 6.1.3 If the modulation format is different from BPSK, the first rank condition of Theorem 6.1.2 remains identical. The second rank condition is modified in the following way [115]:

- real modulation format with alphabet $\Omega = \{\pm 1, \pm 3, \ldots, \pm(m-1)\}$: the columns of $X(:, k : k+j-1)$ include all $\eta^d/2$ possible symbol vectors (up to a sign), with $\eta = \#(\Omega) = m$.
- complex modulation format with alphabet $\Omega = \{\pm 1, \pm 3, \ldots, \pm(m-1)\} \oplus \{\pm 1i, \pm 3i, \ldots, \pm(m-1)i\}$: the columns of $X(:, k : k+j-1)$ include all $\eta^d/2$ possible symbol vectors (up to a sign), with $\eta = \#(\Omega) = m^2$.

For a real modulation format, identification of the symbol sequences is guaranteed up to a sign change $\{+1, -1\}$, for a complex modulation format up to a factor $\{+1, -1, i, -i\}$.

At this point W_k could again be applied in a Viterbi algorithm, with a branch metric identical to the single user case:

$$\min_{\hat{X}(l,:)\in\Omega^{1\times N}} \sum_{k=1}^{N-j+1} \left\| \hat{X}(l, k : k+j-1) \cdot W_k \right\|_2^2 \qquad l = 1, \ldots, d. \tag{6.6}$$

The d transmitted symbol sequences can then be detected as the d paths through the trellis with the lowest path metric. However, a straightforward application of the Viterbi algorithm is hampered by two problems as presented in the next remark.

Remark 6.1.4

P1 If the symbol sequences of two users are identical over a length equal or greater than the memory length of the Viterbi algorithm $j-1$, then the paths of these two users will visit the same state in the Viterbi trellis at the same time instant. Hence they will be indiscernible when the optimal path(s) through the Viterbi trellis is determined since each node has only one survivor path (as explained in section 3.2.1).

Example 6.1.5 The problem is illustrated in figure 6.1 for two users sharing a four-state trellis. The path of each user through the trellis is

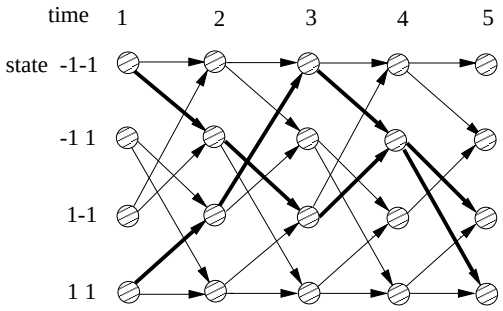


Figure 6.1: A Viterbi trellis with 4 states extending over 5 time steps. The paths of 2 users in the trellis are marked in bold. Both users visit state -11 at time instant 4.

marked in bold. At time 4 both paths merge in state 2. At this point the Viterbi algorithm only keeps track of the path with the lowest path metric, information about the other path is lost. □

A merger of two paths at time k can be detected from a rank drop of $Y_{k|k+i-1}(:, 1 : j-1)$. This is however of little practical use since we only compute the SVD of $Y_{k|k+i-1}$. A way around the problem is to choose j large enough such that the probability of a path crossing becomes negligible. However the complexity of the Viterbi algorithm is exponential in j (see section 3.2.1).

P2 A second problem is that of "phantom" solutions. When the d optimal paths through the Viterbi trellis are determined, typically the following situation will occur. Some solutions will correspond to particular users, while the other solutions are "phantom" solutions which are almost identical to the symbol sequences of the already detected users, but bear no relationship to the symbol sequences of undetected users. For examples of the phenomenon we refer to [138]. In this situation, the Viterbi algorithm is unable to recover all symbol sequences and the undetected users should be considered as interferers rather than as desired users.

To solve the above two problems, we apply *channel coding* at the transmitter. This will allow to split the detection of the different users into different trellises and enhances signal separation.

Assume that d users are synchronized up to a symbol period, and that each sequence of n symbols is extended with p code symbols, then the total trans-

mitted sequences are:

$$\left\{\begin{array}{lllll} \text{user 1:} & \text{n symbols} & \text{p code symbols} & \text{n symbols} & \text{p code symbols} \quad \ldots \\ \text{user 2:} & \text{n symbols} & \text{p code symbols} & \text{n symbols} & \text{p code symbols} \quad \ldots \\ \vdots & & & & \vdots \\ \text{user d:} & \text{n symbols} & \text{p code symbols} & \text{n symbols} & \text{p code symbols} \quad \ldots \end{array}\right. \tag{6.7}$$

We assume a so-called systematic block code [104, p 381]. The theorem below explains how to select p and n.

Theorem 6.1.6 *For d users, choose p such that $d \leq \eta^p$ and n such that $n < \frac{(j-1)}{2} - p + 1$, then code words can be selected such that no two users arrive in the same state in the Viterbi trellis at the same iteration step. Here η represents the alphabet size, i.e. $\eta = \#(\Omega)$.*

Proof: Each user selects η^n different codewords from the η^{n+p} possible codewords. Equation 6.7 then says that two users will have at most $2(n+p-1)$ subsequent equal symbols, i.e. $n-1$ data symbols followed by p code symbols, n data symbols and $p-1$ code symbols. If we choose $2(n+p-1) < (j-1)$, the memory of the Viterbi trellis, then we arrive at the condition $n < \frac{(j-1)}{2} - p + 1$ which assures that no two users will visit the same state at the same iteration step. □

The number of users d together with the modulation format determine the number of code symbols p. The parameter n primarily depends on j, i.e. on the affordable complexity of the Viterbi algorithm. The higher j, the higher n can be chosen and the higher the code rate $\frac{n}{n+p}$. *Theorem 6.1.6 provides a minimum value for p and a maximum value for n.* Obviously, if more coding is added (larger p or smaller n), a better signal separation can be obtained. It is clear that Theorem 6.1.6 provides a solution for problem P1.

With the above coding scheme, the symbol detection procedure may be split into distinct trellises since different users do not visit the same state at the same time instant. The symbol sequences of different users may then be detected in parallel: each user has its own Viterbi trellis with its unique possible states at each iteration step. Since we search only for one symbol sequence in each trellis, problem P2 is solved automatically.

Example 6.1.7 We illustrate the decoding process using a small example with downscaled dimensions. Assume $j = 4$, $d = 2$. According to the coding rules of Theorem 6.1.6 we select $n = 1$, $p = 1$, so for each information symbol we add a code symbol. For the first user, the code symbol is a copy of the information symbol, for the second user we add the sign reversed information

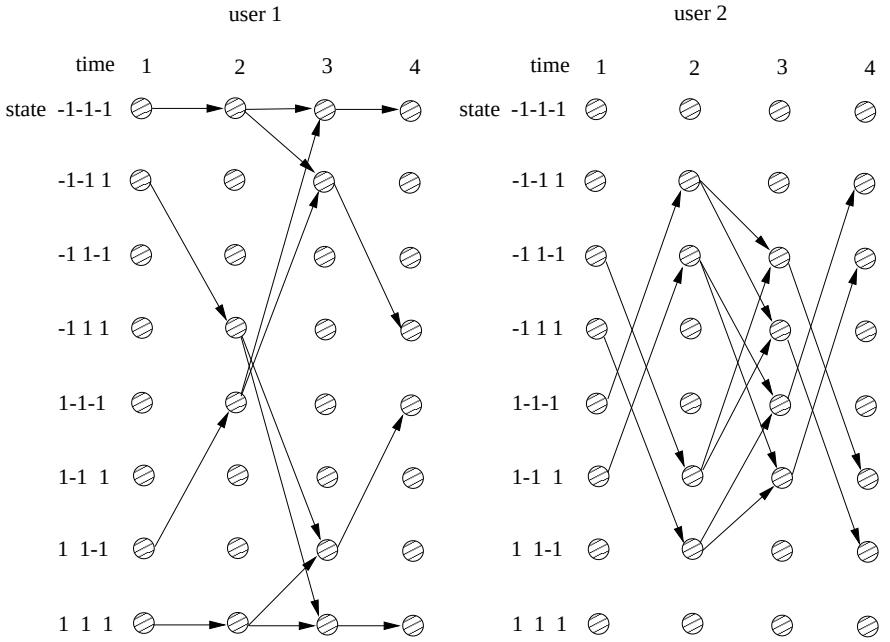


Figure 6.2: Viterbi trellis for the first and second user.

symbol. Figure 6.2 shows the first part of the corresponding Viterbi trellises. Each trellis contains $2^{j-1} = 8$ states where each state represents three BPSK symbols (at time k, the state represents the symbols $\mathbf{v}(k)$ up to $\mathbf{v}(k+2)$). Due to the coding, only 4 states are allowed for each user at each time instant. State transitions are indicated by arrows. The branch metric associated with a state transition from time k to $k+1$ is then computed as $\|\mathbf{v}(k:k+3) \cdot W_k\|_2^2$, see also equation 6.6. In each trellis, the path through the trellis with the lowest path metric provides an estimate of the transmitted symbol sequence. □

The coding thus provides a separation of the symbol sequences of distinct users into different trellises. Furthermore it automatically links each detected sequence with the corresponding user and therefore removes the permutation uncertainty that is present in e.g. the ILSP algorithm [115] (which will be presented in section 8.1). Note that the multi-user subspace + Viterbi algorithm is almost identical to the single user subspace + Viterbi algorithm 3.2.1 except for the fact that branch metrics are now only evaluated for user-specific state transitions.

The separation capability of the coding together with the observation that each symbol is estimated j times in the Viterbi trellis (i.e. it forms part of the

solution of equation 6.5 during j subsequent time steps) makes the algorithm also more robust against the rank conditions of Theorem 6.1.2. The next theorem generalizes the results of Theorem 3.2.3 derived in chapter 3 for the single user case.

Theorem 6.1.8 *Assume that at the beginning of iteration step k, there are d states with a zero path metric, one in each of the d parallel trellises ($k \geq 1$), i.e. the symbol sequence segments $\mathbf{x}^{(r)}(1 : k+j-2), r = 1, \ldots, d$ can be determined uniquely. If the rank conditions of Theorem 6.1.2 are violated less than $j-1$ consecutive iteration steps, then the coded input streams can still be recovered correctly by each of the d Viterbi algorithms.*

Proof: If the rank conditions of Theorem 6.1.2 are not satisfied at iteration step k, there will be more than d solutions to the equation $S \cdot W_k = 0$. However, the exact d solutions still belong to the null space of W_k. Assume a worst case scenario, i.e. when the rank conditions of Theorem 6.1.2 are not satisfied all branch metrics are zero. If the rank conditions of Theorem 6.1.2 are violated during l subsequent iteration steps, then the rank conditions hold again at iteration step $k+l$, and the optimal paths (the paths corresponding to $\mathbf{x}^{(r)}(1 : k+l+j-2), r = 1, \ldots, d$) still have zero cost. At iteration $k+l$, equation 6.5 has d unique solutions (up to a scaling ambiguity) for the symbols $k+l$ up to $k+l+j-1$, one in each Viterbi trellis. The scaling ambiguity can be removed by observing that l is at most $j-2$. Consequently, at least one of the symbols estimated at iteration $k+l+1$ has already been estimated before iteration k (where there were d unique solutions), which resolves the ambiguity. □

Remark 6.1.9 For a reduction of the complexity of the Viterbi algorithm, the same remarks hold as in the single user case, see section 3.2.2. Although the complexity of the Viterbi algorithm is exponential in j, the algorithm is not iterative in contrast with the multi-user ILSP algorithm of [115] (see section 8.1).

Remark 6.1.10 The conceptually simple signal separation technique at the receiver side does not come without any cost. First, the symbol sequences of all users must be synchronized. Secondly, the coding has some impact on transmission efficiency (for figures we refer to the simulation examples of chapter 8).

Remark 6.1.11 The subspace decomposition step (i.e. the ISI removal step) is common for all users. Only the MAI removal is different for each user. Furthermore, when only the symbol sequence of a specific user is desired, only the coding scheme of that user needs to be known.

Remark 6.1.12 In a CDMA system, spreading sequences automatically provide a different cost function for symbol recovery of different users. Hence extra coding is not necessary when a similar subspace + Viterbi algorithm is used for symbol recovery [75].

Remark 6.1.13 The above algorithm is also readily applied to the blind signal separation problem, i.e. we just replace L by 0 in all steps of the algorithm.

Remark 6.1.14 In a usual setting, block coding is used for error protection purposes. Based on hard (or soft) decisions, coding information is used to prevent symbol errors. Most well known binary block coding techniques are linear in Galois field GF(2) $= \{0,1\}$[1], where multiplication and addition are defined modulo-2 [104]. The code is said to be linear if for two code words $\mathbf{t_1}$, $\mathbf{t_2}$, the linear combination $\alpha_1 \cdot \mathbf{t_1} + \alpha_2 \cdot \mathbf{t_2}$ is also a code word, with $\alpha_1, \alpha_2 \in$ GF(2). This implies that a linear code contains the all zero code word. For the multi-user coding scheme presented above, this implies that at most one user can use a linear coding scheme. Otherwise at least two users would share the all zero code word and the coding requirements of Theorem 6.1.6 are not fulfilled. This shows that the extension of classical block coding techniques to the multi-user coding scheme presented here is not straightforward. Hence in an actual implementation, the multi-user coding scheme presented above will not replace existing (single user error protecting) coding schemes. Rather the multi-user coding scheme forms an extra (outer) layer of coding, specifically designed to enhance multi-user detection. The "classical" (inner layer of) coding could then still provide error protection/correction.

We conclude this section by presenting some coding schemes that fulfill the requirements of Theorem 6.1.6. These coding schemes will then also be used in the simulation examples of chapter 8. For simplicity we only consider the case $p = \log_\eta d$, where the number of users d is such that p in an integer number.

Example 6.1.15 Define the rth component of an arbitrary code word of user l as $\mathbf{t}^{(l)}(r)$ and define even and odd parity as follows:

$$\begin{array}{rcrl} \mathcal{P}^e(\mathbf{v}) & = & 1 & \text{if the vector } \mathbf{v} \text{ contains an odd number of ones} \\ & = & -1 & \text{if the vector } \mathbf{v} \text{ contains an even number of ones} \\ \mathcal{P}^o(\mathbf{v}) & = & 1 & \text{if the vector } \mathbf{v} \text{ contains an even number of ones} \\ & = & -1 & \text{if the vector } \mathbf{v} \text{ contains an odd number of ones.} \end{array}$$

Z1 $d = 2$ and BPSK modulation, $p = 1$. The first user employs even parity coding: when the first n positions of a code word contain an odd number of ones, the parity bit is 1, otherwise it is -1:

$$\mathbf{t}^{(1)}(n+1) = P^e(\mathbf{t}^{(1)}(1:n)).$$

[1] A code symbol 0 corresponds to a transmitted -1, a 1 corresponds to a 1.

The second user uses odd parity coding:

$$\mathbf{t}^{(2)}(n+1) = P^o(\mathbf{t}^{(2)}(1:n)).$$

Z2 $d = 2^r$, $r \in \mathbb{N}$, BPSK coding and $p = r$. We generalize parity coding in the following way. Associate with each of the $d = 2^r$ users a different binary code word of length $p = r$, e.g.

$$\begin{array}{lrcccccc} \text{user } 1\text{:} & \mathbf{c}^{(1)} & = & -1 & \dots & -1 & -1 & \\ \text{user } 2\text{:} & \mathbf{c}^{(2)} & = & -1 & \dots & -1 & 1 & \\ \vdots & & & & & \vdots & & \\ \text{user } d\text{:} & \mathbf{c}^{(d)} & = & 1 & \dots & 1 & 1 & . \end{array}$$

For a user l, when the first n positions of a code word contain an odd number of ones, the code word is extended with $\mathbf{c}^{(l)}$, otherwise with $-\mathbf{c}^{(l)}$:

$$\mathbf{t}^{(l)}(n+1:n+p) = P^e(\mathbf{t}^{(1)}(1:n))\mathbf{c}^{(l)}$$

Z3 $d = 4^r$, $r \in \mathbb{N}$, QPSK coding ($\Omega = \{-1, 1, -i, i\}$) and $p = r$. Replace the first n symbols of a code word by $2 \cdot n$ digits by making the following substitution:

$$\begin{array}{rcrr} 1 & \rightarrow & -1 & -1 \\ i & \rightarrow & -1 & 1 \\ -1 & \rightarrow & 1 & 1 \\ -i & \rightarrow & 1 & -1 \quad . \end{array} \tag{6.8}$$

Add $2 \cdot p$ code symbols using coding scheme Z2. Finally translate the code words of length $2 \cdot (n+p)$ back to length $(n+p)$ QPSK code words by doing a reverse substitution (with respect to equation 6.8). The extension to other modulation formats can be done similarly by replacing code words of length $(n+p)$ by binary representations of length $\log_2(\eta) \cdot (n+p)$.

□

Remark 6.1.16 The coding examples presented above fulfill the deterministic coding requirements of Theorem 6.1.6 and were designed by a rather ad hoc generalization of parity coding. A statistical approach for optimal code design in the presence of noise has not yet been developed. The performance of the above coding schemes will be assessed by simulation in chapter 8.

6.2 Unequal channel lengths

Thus far, we have assumed the (unrealistic) case where all users have equal channel lengths. In this section we show how the algorithm is modified when

channel lengths are unequal. We show how coding can provide robustness against residual ISI. Further we also demonstrate that coding provides robustness against an overestimation of the system order. Before presenting the general case, we introduce the problem using an example that will also be used in the simulation examples of chapter 8.

Example 6.2.1 Assume two users ($d = 2$), with channel lengths $L_1 = 4$ and $L_2 = 2$. Analog to equation 2.13 we obtain the following data model:

$$Y_{k|k+i-1} = \left[\begin{array}{c|c} \mathcal{H}_i^{(1)} & \mathcal{H}_i^{(2)} \end{array}\right] \left[\begin{array}{ccc} x^{(1)}(k-L_1) & \cdots & x^{(1)}(k-L_1+j-1) \\ \vdots & & \vdots \\ x^{(1)}(k+i-1) & \cdots & x^{(1)}(k+i+j-2) \\ \hline x^{(2)}(k-L_2) & \cdots & x^{(2)}(k-L_2+j-1) \\ \vdots & & \vdots \\ x^{(2)}(k+i-1) & \cdots & x^{(2)}(k+i+j-2) \end{array}\right]$$

where $\mathcal{H}_i^{(1)}$ and $\mathcal{H}_i^{(2)}$ are block Toeplitz matrices of size $M \cdot i \times (L_1 + i)$ and $M \cdot i \times (L_2 + i)$ which contain the channel parameters of the first respectively second user. Note that:

$$\left[\begin{array}{ccc} x^{(1)}(k-2) & \cdots & x^{(1)}(k+j-3) \\ x^{(1)}(k-1) & \cdots & x^{(1)}(k+j-2) \\ x^{(1)}(k) & \cdots & x^{(1)}(k+j-1) \\ \hline x^{(2)}(k) & \cdots & x^{(2)}(k+j-1) \end{array}\right] \cdot \left[\begin{array}{ccc} V_{k-i+1}^{\perp} & \cdots & V_{k+L_2}^{\perp} \end{array}\right] = 0 \quad (6.9)$$

$$\left[\begin{array}{ccc} x^{(1)}(k) & \cdots & x^{(1)}(k+j-1) \end{array}\right] \cdot \left[\begin{array}{ccc} V_{k-i+1}^{\perp} & \cdots & V_{k+L_1}^{\perp} \end{array}\right] = 0. \quad (6.10)$$

Contrary to the case of equal channel lengths, we are unable to formulate a $d = 2$-dimensional problem (coding information not taken into account so far). If we apply $(i + L_2)$ shifts (equation 6.9), we obtain a 4-dimensional solution space, where two solutions are undesired shifts. When we apply $(i + L_1)$ shifts (equation 6.10) we only obtain a one-dimensional solution space. The problem is further complicated by the fact that the channel lengths L_1 and L_2 are not known a priori. □

Consider now the general case:

$$Y_{k|k+i-1} = \mathcal{H}_i^{(1)} \cdot X_{k-L_1|k+i-1}^{(1)} + \ldots + \mathcal{H}_i^{(d)} \cdot X_{k-L_d|k+i-1}^{(d)}$$

and define:

$$W_k^{i+r} \triangleq \left[\begin{array}{ccc} V_{k-i+1}^{\perp} & \cdots & V_{k+r}^{\perp} \end{array}\right] \quad (6.11)$$

$$L_{min} = \min_{l=1}^{d} L_l$$

$$L_{max} \quad = \quad \max_{l=1}^{d} L_l.$$

W_k^{i+r} is defined similarly as W_k in equation 6.4 but now we introduce an extra superscript to denote the number of shifts. We assume that L_{min} is known, the extension to the case where this bound is not known is straightforward as will become clear further on, see remark 6.2.3. Note that we do not identify the channel length of each user separately, instead we only require a lower bound on the channel order to be known.

Now apply $(i + L_{min})$-shifts and construct $W_k^{i+L_{min}}$. The matrix $W_k^{i+L_{min}}$ defines a $(d + \sum_{l=0}^{d}(L_l - L_{min}))$-dimensional null space. Only d solutions are desired, the other solutions are shifts of the transmitted symbol sequences. The aim of the algorithm is now to *exclude the shifts based on code information.* Part of the ISI is thus rejected by including $i + L_{min}$ shifts in $W_k^{i+L_{min}}$. The remaining ISI and MAI is rejected based on coding information. The question then rises how to construct coding matrices such that additional ISI is removed.

Example 6.2.2 Resuming example 6.2.1, when the Viterbi algorithm exploits subspace information only, four symbol sequences are detected (have a zero path metric):

$$\begin{array}{ccc} x^{(1)}(-1) & \dots & x^{(1)}(N-2) \\ x^{(1)}(0) & \dots & x^{(1)}(N-1) \\ x^{(1)}(1) & \dots & x^{(1)}(N) \\ x^{(2)}(1) & \dots & x^{(2)}(N) \end{array}$$

The aim is now to exclude the first two symbol sequences based on coding information, i.e. the first two rows should not consist of a sequence of valid code words. Consider the first $n+p$ components of the above symbol sequences:

$$\left.\begin{array}{cccc|ccc} \star & \star & \cdots & x^{(1)}(n-2) & x^{(1)}(n-1) & \cdots & x^{(1)}(n+p-2) \\ \star & x^{(1)}(1) & \cdots & x^{(1)}(n-1) & x^{(1)}(n) & \cdots & x^{(1)}(n+p-1) \\ x^{(1)}(1) & x^{(1)}(2) & \cdots & x^{(1)}(n) & x^{(1)}(n+1) & \cdots & x^{(1)}(n+p) \\ x^{(2)}(1) & x^{(2)}(2) & \cdots & x^{(2)}(n) & x^{(2)}(n+1) & \cdots & x^{(2)}(n+p) \end{array}\right. \tag{6.12}$$

In order for the Viterbi algorithm to work properly, code words should be selected such that the first two rows above can never be valid code words. To ensure this, at least one of the last p components of the first row is still to contain (shifted) code information and hence $p > (L_{max} - L_{min}) = 2$ [2]. We explain the coding mechanism using a practical coding example. We assume $p = L_{max} - L_{min} + 1$, for the example above $p = 4 - 2 + 1 = 3$, then code words can be constructed as:

$$\mathbf{t}^{(1)}(n+1) \quad = \quad \mathcal{P}^e(\mathbf{t}^{(1)}(1:n-2))$$

[2] Note that if coding were only to remove MAI, $p = 1$ is sufficient (see Theorem 6.1.6).

$$\begin{aligned}\mathbf{t}^{(1)}(n+2) &= \mathcal{P}^o(\mathbf{t}^{(1)}(2:n-1))\\ \mathbf{t}^{(1)}(n+3) &= \mathcal{P}^o(\mathbf{t}^{(1)}(3:n)).\end{aligned}$$

Straightforward calculation yields that when a code word of the first user is shifted over one or two positions, it does not form a new valid code word. Hence the coding scheme provides the ISI protection needed for example 6.2.1. □

Remark 6.2.3 If L_{min} is not known, $L_{min} = 0$, the coding method remains unaltered but ISI protection has to be done over L_{max} symbol periods, i.e. $p > L_{max}$.

The joint use of subspace information, the finite alphabet property and code information allows to identify the symbol sequences of all users without identifying their channel lengths. In the previous section, we searched for d symbol sequences in a d-dimensional space. Theorem 6.1.8 already indicated that exploiting coding information together with the finite alphabet property allows to the relax rank conditions of Theorem 6.1.2. With the modifications of this section we search d symbol sequences in a $d + \sum_{l=1}^{d}(L_l - L_{min})$-dimensional space. We rely more on coding and the finite alphabet property to recover the symbol sequences. Note that coding information is known *exactly* (even with noise) while subspace information is known only *approximately* (due to noise).

However, as was explained in the remarks following Theorem 6.1.6, the coding rate $\frac{n}{n+p}$ is primarily determined by the parameter j (the complexity of the Viterbi algorithm). For a fixed complexity (i.e. a fixed j), a large p (as can be required in the case of large channel length differences, $p > L_{max} - L_{min}$) implies a lot of coding overhead. In that case, *the amount of coding needed by the Viterbi algorithm might be prohibitive* (see also the simulation examples in chapter 8). As we will see in the next chapter, there are other coding techniques that provide better and more efficient (residual) ISI protection than the coding scheme presented here. These coding schemes can equally well be combined with a Viterbi algorithm as will be explained.

Finally, so far we assumed that the null space dimension $r_{V_k^\perp}$ (i.e. the number of columns of $V_k^\perp$) was detected correctly as $r_{V_k^\perp} = j - d\cdot i - \sum_{l=1}^{d} L_l$. Although several order detection techniques based on the SVD spectrum do exist, correct order detection remains difficult especially when signal-to-noise ratios are low. In the same spirit as above, it might therefore be useful to underestimate $r_{V_k^\perp}$ (overestimate the system order) and to rely on coding (and the finite alphabet property) to detect the correct symbol sequences. $W_k^{i+L_{min}}$ is a $j \times (i+L_{min})\cdot r_{V_k^\perp}$ matrix and forms a set of homogeneous equations in j unknown transmitted symbols.

When j is large $(i + L_{min}) \cdot r_{V_k^\perp} \gg j$ and $W_k^{i+L_{min}}$ is largely overdetermined. The null space of $W_k^{i+L_{min}}$ will be $d + \sum_{l=1}^{d}(L_l - L_{min})$-dimensional even

when $r_{V_k^\perp}$ is underestimated. Furthermore, the coding information additionally protects against phantom solutions.

However, for moderate j, $W_k^{i+L_{min}}$ may be underdetermined when $r_{V_k^\perp}$ is chosen too low and performance degradations may occur. As will be explained in the next chapter and in the simulation results of chapter 8 block processing algorithms (that use a large j) will suffer less from the problem of an underestimated $r_{V_k^\perp}$.

6.3 Conclusions

In this chapter we have extended the single user subspace + Viterbi algorithm of chapter 3 to the multi-user scenario. We have presented a fully adaptive algorithm for multi-user blind channel equalization. The algorithm is based on an adaptive subspace decomposition for a virtual channel identification type operation and the Viterbi algorithm for subsequent symbol detection. MAI is removed by exploiting both the finite alphabet property of the digital input signals and coding redundancy. Coding has allowed to split symbol detection for different users into distinct trellises which may be searched in parallel. Furthermore, coding has been used to increase robustness against channel length estimation problems and has been shown to increase the robustness of subspace algorithms against an overestimation of the system order. However, for the last two purposes the window size (the parameter j) plays an important role and the Viterbi algorithm may not provide enough flexibility. The amount of coding used in the Viterbi algorithm may be prohibitive (see also the simulation results in chapter 8). A block processing + coding approach, as presented in the next chapter, can be more appropriate to provide robustness against channel length and system order estimation problems.

Chapter 7

Linear coding based blind signal separation

In the previous chapter we have presented a multi-user blind equalization algorithm based on a subspace + Viterbi approach using a non-linear coding scheme and a finite alphabet constraint. In this chapter we restrict ourselves to linear[1] codes because we envisage linear decoding techniques based on a quadratic non-triviality constraint. While the algorithms developed in previous chapters were based on direct symbol recovery, in this chapter we also study algorithms which aim at direct equalizer design. The output of the equalizer then forms an estimate of the transmitted signal. As for the subspace + Viterbi algorithm of chapter 6 the application of coding at the transmitter side "automatically" removes MAI and differentiates the algorithms presented here from existing signal separation approaches [115, 6, 128] (see also the more elaborate discussion in the introduction of part two).

Remark 7.0.1 As already indicated, coding at the transmitter side has been proposed in the context of transmitter induced cyclostationarity [19, 49]. However the algorithms of [19, 49] are stochastic, single user algorithms while the algorithms developed here are deterministic and multi-user.

We present three linear coding schemes.

The *first* algorithm is based on block coding and is applicable for any modulation scheme. The price paid for the general applicability is some loss in transmission efficiency due to the coding. The amount of coding overhead can be balanced against the separation capabilities of the algorithm.

[1] Linear in the field of real or complex values.

The *second* coding scheme is only applicable when the data symbols belong to a real constellation, e.g. BPSK. The coding scheme maintains the information rate without increasing the bandwidth.

In the *third* coding scheme we exploit transmit diversity: we assume that each transmitter is equipped with at least two antennas. Transmit diversity combined with coding will then allow for an easy signal separation at the receiver side. The coding scheme again maintains the information rate without increasing the bandwidth and does not impose any restriction on the modulation format.

The aim of this chapter is not only to develop conceptually simple decoding algorithms, but also to illustrate that algorithms can be designed with flexible trade-offs. One can trade transmission efficiency (coding overhead in the first algorithm) for a restriction on the modulation format (second algorithm) or for hardware (extra transmitter and receiver antennas in the third algorithm). Furthermore it is demonstrated how coding increases the robustness of subspace algorithms to the common problems of order estimation and channel length detection, see e.g. [81].

For the presentation of the algorithms we will take a different approach than in the previous chapter. In order to make the presentation not to cumbersome, we start from a *blind source separation* problem in a block processing context (i.e. first we consider the problem of MAI only (no ISI)). In section 7.1 we present the data model for blind signal separation. In the sections 7.2, 7.3 and 7.4 we present blind signal separation algorithms C1, C2 and C3 based on three different coding techniques. After presenting the main ideas, we extend the results in two directions: we show how to modify the algorithms to additionally remove ISI (section 7.5) and we explain how to make them adaptive (section 7.6). In section 7.7 we present some conclusions.

7.1 Data model

Assume d sources transmitting digital symbols at the same symbol rate to an M element antenna array (we assume 1 antenna per transmitter). If the sources are synchronized and if the multi-path delay spread is negligible, then the sampled received (vector) signal can be expressed as:

$$\mathbf{y_k} = A \cdot \mathbf{x_k} + \mathbf{n_k}. \tag{7.1}$$

Here

$$\begin{aligned} \mathbf{y_k} &= \left[\begin{array}{ccc} y_1(k) & \cdots & y_M(k) \end{array} \right]^T \\ \mathbf{x_k} &= \left[\begin{array}{ccc} x^{(1)}(k) & \cdots & x^{(d)}(k) \end{array} \right]^T \end{aligned}$$

$$\mathbf{n_k} = \left[\begin{array}{ccc} n_1(k) & \cdots & n_M(k) \end{array} \right]^T .$$

The mixing matrix $A \in \mathbb{C}^{M \times d}$ is assumed to have full column rank d.

Remark 7.1.1 One can also think of this data model as being obtained after an initial ISI removal step.

If each user transmits N symbols (for the time being we omit the noise), the above data model can be rewritten as:

$$\left[\begin{array}{ccc} \mathbf{y_1} & \cdots & \mathbf{y_N} \end{array} \right] = A \cdot \left[\begin{array}{ccc} \mathbf{x_1} & \cdots & \mathbf{x_N} \end{array} \right] \quad (7.2)$$

$$Y = A \cdot X. \quad (7.3)$$

where the $d \times N$ matrix X is defined as in equation 2.14.

7.2 Algorithm C1

The first algorithm is based on a block coding scheme and can be applied for any modulation format. In contrast with the two coding schemes that are presented in the next sections this first coding scheme does require a transmission overhead in order to properly separate the different sources. However, the minimum amount of overhead increases only linearly with the number of sources.

At the transmitter side, each input symbol stream $\mathbf{z}^{(l)}$ is block coded as:

$$\mathbf{x}^{(l)} = \mathbf{z}^{(l)} \cdot \underbrace{\left[\begin{array}{cc} I_{N-p} & D_l' \end{array} \right]}_{D_l} .$$

Here $\mathbf{x}^{(l)}$ is the sequence of symbols which is actually transmitted to the receiver. The data model of equation 7.1 thus remains valid. $\mathbf{x}^{(l)}$ is a $1 \times N$ vector. If there are p code symbols then $\mathbf{z}^{(l)}$ is a $1 \times (N-p)$ vector, I_{N-p} is an $(N-p) \times (N-p)$ identity matrix and D_l' is an $(N-p) \times p$ coding matrix.

The aim of the algorithm is to recover the symbol sequences $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(d)}$ *from the output matrix* Y *and the knowledge of the matrices* $D_1, \ldots, D_d$. Section 7.2.1 presents the algorithm. Section 7.2.2 focuses on the design of the matrices D_l.

7.2.1 Algorithm C1

The algorithm consists of two steps. The first step is based on an SVD of Y. The second step exploits the knowledge of the matrices $D_1, \ldots, D_d$ and

removes MAI in a non-iterative way. For the derivations to follow, we first omit the noise, i.e. we assume $\mathbf{n_k} = 0$, next we show how noise can be incorporated into the problem formulation.

The *first step* is based on an SVD of the matrix Y:

$$\begin{aligned} Y &= U \cdot \Sigma \cdot V^H \\ &= \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}. \end{aligned} \tag{7.4}$$

If A and X have respectively full column and row rank, then $V^\perp$ has exactly a d-dimensional left null space and X spans that null space:

$$X \cdot V^\perp = 0.$$

Now we are left to remove MAI in the *second step*. Define for each user l:

$$V_l^\perp = D_l \cdot V^\perp \qquad l = 1, \ldots, d \tag{7.5}$$

such that:

$$\mathbf{z}^{(l)} \cdot V_l^\perp = 0 \qquad l = 1, \ldots, d. \tag{7.6}$$

The matrices D_l should now be selected such that $V_l^\perp$ has a one-dimensional left null space to which $\mathbf{z}^{(l)}$ belongs. In this way we obtain d one-dimensional problems.

With noise, we search for a matrix of coded symbol sequences $\hat{X}$ such that:

$$\min_{\hat{X}} \|\hat{X} \cdot V^\perp\|_F^2 = \min_{\hat{\mathbf{z}}^{(1)}, \ldots, \hat{\mathbf{z}}^{(d)}} \| \begin{bmatrix} \hat{\mathbf{z}}^{(1)} \cdot D_1 \\ \vdots \\ \hat{\mathbf{z}}^{(d)} \cdot D_d \end{bmatrix} \cdot V^\perp \|_F^2. \tag{7.7}$$

We now review three properties of the Frobenius norm, which will also prove useful during the derivation of the algorithm C2 (section 7.3) and algorithm C3 (section 7.4).

Property 7.2.1 *Assume $A \in \mathbb{C}^{m \times n}$ and $B \in \mathbb{C}^{n \times p}$ then:*

$$\begin{aligned} \|A \cdot B\|_{\mathrm{F}}^2 &= \sum_{k=1}^{m} \|A(k,:) \cdot B\|_{\mathrm{F}}^2 \\ \|A \cdot B\|_{\mathrm{F}}^2 &= \left\| \begin{bmatrix} A(1,:) \cdot B & \ldots & A(m,:) \cdot B \end{bmatrix} \right\|_{\mathrm{F}}^2 \\ \|A \cdot Q\|_{\mathrm{F}}^2 &= \|A\|_{\mathrm{F}}^2 \end{aligned}$$

where $Q \in \mathbb{C}^{m \times m}$ is a unitary matrix, $Q^H \cdot Q = I$ and $Q \cdot Q^H = I$.

Algorithm 7.2.1
Algorithm C1

1. *Compute the SVD of* Y*:*

$$Y = \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & \Sigma^\perp \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}.$$

2. *for* $l = 1, \ldots, d$

$$\begin{aligned} D_l \cdot V^\perp &= \widetilde{U} \cdot \widetilde{\Sigma} \cdot \widetilde{V}^H \\ \hat{\mathbf{z}}^{(l)} &= \mathcal{D}(\alpha \cdot \widetilde{U}(:, N-p)^H) \end{aligned}$$

end

The first property allows to split equation 7.7 into d subproblems:

$$\min_{\hat{\mathbf{z}}^{(l)}} \left\| \hat{\mathbf{z}}^{(l)} \cdot V_l^\perp \right\|_{\mathrm{F}}^2 \qquad l = 1, \ldots, d. \tag{7.8}$$

Under a unit norm constraint, the symbol sequence of each user can be determined as the left singular vector corresponding to the smallest singular value of $V_l^\perp$. Using block coding, we have thus split a d-dimensional problem into d one-dimensional problems, which may be solved in parallel. The algorithm exploits subspace and coding information for symbol recovery. The finite alphabet property is not used and in this sense the algorithm is suboptimal. The algorithm is summarized as algorithm 7.2.1. The operator $\mathcal{D}(\cdot)$ denotes a mapping on the closest constellation symbol and α is an appropriate scaling factor.

Remark 7.2.1 Algorithm C1 can be made adaptive, but then the coding information has to be spread across the burst (omitting exact details) instead of being grouped at the end of the burst. Notation however, becomes cumbersome and more coding will be needed than for the block processing version of the algorithm presented above. The algorithm C2 (section 7.3) and algorithm C3 (section 7.4) lend themselves better to an adaptive implementation.

Remark 7.2.2 Several variants of the algorithm C1 can be constructed, e.g. using the signal subspace instead of the noise subspace. These variants will be discussed in section 7.3.3 where we present similar algorithmic variations for algorithm C2.

The next section explains how to construct the coding matrices D_l.

7.2.2 Code selection

First we establish some general conditions on the matrices D_l, then we present a specific strategy for code selection.

We want the matrices $V_l^{\perp} \in \mathbb{C}^{(N-p)\times(N-d)}$ to have a one-dimensional left null space, this requires

$$p \geq d - 1.$$

The minimum overhead, necessary to separate the different users, increases linearly with the number of users. Of course, if more coding is allowed, a better signal separation can be obtained.

To get some insight into the code construction process it is useful to rewrite the rank condition on $V_l^{\perp}$ as a function of the coding matrices D_l and the coded input matrix X. We start from equation 7.5:

$$\begin{aligned} \text{rank}(V_l^{\perp}) &= N - p - 1 \\ &\Updownarrow \\ \text{rank}(D_l \cdot V^{\perp}) &= N - p - 1. \end{aligned} \tag{7.9}$$

The code matrices D_l should be such that the rank of $D_l \cdot V^{\perp}$ can never drop below $N - p - 1$. To rewrite equation 7.9, we make use of the following matrix rank property.

Property 7.2.2 *Assume $A \in \mathbb{C}^{m\times n}$ and $B \in \mathbb{C}^{n\times p}$ then:*

$$\textit{rank}\ (A \cdot B) = \textit{rank}\ (A) - \textit{dim}(\textit{span}_{\textit{row}}(A) \cap \textit{span}_{\textit{col}}(B)^{\perp}). \tag{7.10}$$

Applying this property to equation 7.9 gives:

$$\begin{aligned} \dim(\text{span}_{\text{row}}(D_l) \cap \text{span}_{\text{col}}(V^{\perp})^{\perp}) &= (N-p) - (N-p-1) \\ &= 1 \\ &\Updownarrow \\ \dim(\text{span}_{\text{row}}(D_l) \cap \text{span}_{\text{row}}(X)) &= 1 \end{aligned}$$

Now applying the dimension law of vector spaces (equation 3.6), we get:

$$\dim(\text{span}_{\text{row}}(\begin{bmatrix} X \\ D_l \end{bmatrix})) = N - p + d - 1.$$

Again, the matrices D_l should be such that the rank of $\begin{bmatrix} X \\ D_l \end{bmatrix}$ can never drop below $N - p + d - 1$. Note that $\mathbf{x}^{(l)}$ lies in the row space of D_l by construction

($\mathbf{x}^{(l)} = \mathbf{z}^{(l)} \cdot D_l$). Now define X_{-l} as X in which the lth row is omitted. Then the above equation is equivalent to

$$\text{rank}(\begin{bmatrix} X_{-l} \\ D_l \end{bmatrix}) = N - p + d - 1. \tag{7.11}$$

Stated in a different way, *the matrix* $\left[\begin{smallmatrix} X_{-l} \\ D_l \end{smallmatrix}\right]$ *should have full row rank.* We now present some examples of very simple coding schemes fulfilling rank condition (7.11).

Example 7.2.3

Z1 If $p = d$, we can use the following coding matrices, $D_l = \begin{bmatrix} I_{N-p} & D'_l \end{bmatrix}$ and:

$$\begin{aligned} D'_1 &= \begin{bmatrix} -\mathbf{e_1} & \mathbf{e_2} & \cdots & \mathbf{e_p} \end{bmatrix} \\ D'_2 &= \begin{bmatrix} \mathbf{e_1} & -\mathbf{e_2} & \cdots & \mathbf{e_p} \end{bmatrix} \\ \vdots &= \vdots \\ D'_d &= \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} & \cdots & -\mathbf{e_p} \end{bmatrix} \end{aligned}$$

where $\mathbf{e_i}$ is the ith column of an $(N-p) \times (N-p)$ identity matrix. We now prove that this coding scheme fulfills equation 7.11: we prove that $\left[\begin{smallmatrix} X_{-l} \\ D_l \end{smallmatrix}\right]$ has full rank, $\forall l$.

Proof: The proof is ex absurdo.

Assume that $\left[\begin{smallmatrix} X_{-l} \\ D_l \end{smallmatrix}\right]$ is not of full row rank. Since D_l has full row rank by construction, a row $\mathbf{x}^{(k)}$ can be written as a linear combination of the other rows, i.e.

$$\mathbf{x}^{(k)} = \underbrace{\begin{bmatrix} a_1 & \cdots & a_{(d-2)+(N-p)} \end{bmatrix}}_{\mathbf{a}^T} \cdot \underbrace{\begin{bmatrix} X_{-k,-l} \\ D_l \end{bmatrix}}_{B},$$

where $X_{-k,-l}$ denotes the matrix X in which both the kth and lth row have been removed. We now select the equations in $\mathbf{x}^{(k)}(k)$ and $\mathbf{x}^{(k)}(N-p+k)$:

$$\begin{aligned} \mathbf{x}^{(k)}(k) &= \mathbf{a}^T \cdot B(:,k) \\ \mathbf{x}^{(k)}(N-p+k) &= \mathbf{a}^T \cdot B(:,N-p+k). \end{aligned}$$

The kth and $(N-p+k)$th row of B are equal by construction such that:

$$\mathbf{x}^{(k)}(k) = \mathbf{x}^{(k)}(N-p+k).$$

However, from the coding scheme for user k, it follows that:

$$\mathbf{x}^{(k)}(N-p+k) = -\mathbf{x}^{(k)}(k).$$

Hence $\mathbf{x}^{(k)}(k)$ should be zero, which is not true. Consequently, $\begin{bmatrix} X_{-l} \\ D_l \end{bmatrix}$ has full rank $\forall l$. □

Z2 If $p = 2^{(d-1)}$ one could use the following coding scheme, here shown for the case $d = 3$:

$$\begin{aligned} D_1' &= \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} & \mathbf{e_3} & \mathbf{e_4} \end{bmatrix} \\ D_2' &= \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} & -\mathbf{e_3} & -\mathbf{e_4} \end{bmatrix} \\ D_3' &= \begin{bmatrix} \mathbf{e_1} & -\mathbf{e_2} & \mathbf{e_3} & -\mathbf{e_4} \end{bmatrix}. \end{aligned}$$

The proof that this coding scheme is valid is easily constructed along the same lines as for coding scheme Z1.

Z3 If $p = b \cdot 2^{(d-1)}$ with $b \geq 1$ we can easily extend coding scheme Z2, e.g. for $d = 3$, $b = 2$:

$$\begin{aligned} D_1' &= \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} & \mathbf{e_3} & \mathbf{e_4} & \mathbf{e_5} & \mathbf{e_6} & \mathbf{e_7} & \mathbf{e_8} \end{bmatrix} \\ D_2' &= \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} & \mathbf{e_3} & \mathbf{e_4} & -\mathbf{e_5} & -\mathbf{e_6} & -\mathbf{e_7} & -\mathbf{e_8} \end{bmatrix} \\ D_3' &= \begin{bmatrix} \mathbf{e_1} & \mathbf{e_2} & -\mathbf{e_3} & -\mathbf{e_4} & \mathbf{e_5} & \mathbf{e_6} & -\mathbf{e_7} & -\mathbf{e_8} \end{bmatrix}. \end{aligned}$$

Remark 7.2.4 As a rule of thumb, in a set of good coding matrices, the matrices D_l' differ in many diagonal positions. Obviously, when two matrices D_l' are identical signal separation fails.

The optimal codes are those that maximize the ratio of the one but smallest to the smallest singular value of the matrices $V_l^{\perp}$, $l = 1, \ldots, d$. Clearly, when a $V_l^{\perp}$ has more than one zero singular value, the null space of $V_l^{\perp}$ is more than one-dimensional and signal separation has failed. The matrix $V_l^{\perp}$ is the product of a matrix D_l, which can be selected, and a matrix $V^{\perp}$, which is a function of the input sequences and the noise, which are only known in statistical sense (viewing the input sequences as deterministic quantities does not make the problem easier). The development of optimal codes, should thus proceed in a statistical framework and remains an open problem. The optimality of the codes can also be interpreted as a function of the angles between the subspaces [53] spanned by D_l and $V^{\perp H}$, $0 \leq \theta_1 \leq \ldots \leq \theta_{N-p-1} \leq \theta_{N-p} = \frac{\pi}{2}$. The largest principal angle between the row spaces of D_l and $V^{\perp H}$, θ_{N-p}, equals $\frac{\pi}{2}$ because $\mathbf{x}^{(l)}$ lies in the row space of D_l and also lies in the null space of $V^{\perp}$. Significant performance degradations can be expected when there is another principal angle close to $\frac{\pi}{2}$.

7.3 Algorithm C2

The algorithm C1 presented in the previous section and the subspace + Viterbi algorithm of chapter 6 are generally applicable but induce a coding overhead. This section presents an algorithm that does not impose any overhead but the algorithm is only applicable when the modulation format is (real) one-dimensional, e.g. BPSK. In section 7.4 we present an algorithm C3 that is applicable for any modulation format and has no coding overhead. The algorithm C3 can then be viewed as an alternative for the algorithm C2 when the modulation format is complex two-dimensional.

The classical way to exploit the knowledge that the signal constellation is real, is to split the received signal in its real and imaginary part and hence to double the number of observations prior to any other operation (see [69], [133], [128] or [115]). Equation 7.1 is then rewritten as:

$$\begin{bmatrix} \Re(\mathbf{y_k}) \\ \Im(\mathbf{y_k}) \end{bmatrix} = \underbrace{\begin{bmatrix} \Re(A) \\ \Im(A) \end{bmatrix}}_{\widetilde{A}} \cdot \mathbf{x_k} + \begin{bmatrix} \Re(\mathbf{n_k}) \\ \Im(\mathbf{n_k}) \end{bmatrix}. \tag{7.12}$$

Here we will follow a different approach. We show that for a real constellation, MAI interference can be removed efficiently by multiplying each input sequence by a different complex coding sequence (information rate is maintained without increasing the bandwidth).

The coding scheme multiplies each data symbol sequence $\mathbf{z}^{(l)} \in \mathbb{R}^{1\times N}$ (with symbols in a real alphabet Ω, i.e. $\Omega \subset \mathbb{R}$), by a different complex diagonal coding matrix D_l, with symbols in some complex alphabet $\Pi \subset \mathbb{C}$, i.e. $\mathbf{x}^{(l)} = \mathbf{z}^{(l)} \cdot D_l$. The result $\mathbf{x}^{(l)}$ is a complex coded data sequence with symbols in $\Omega \times \Pi \subset \mathbb{C}$, which is transmitted to the receiver. Note that this coding maintains the information rate without increasing the bandwidth. *The aim of the algorithm is to recover the symbol sequences* $\mathbf{z}^{(1)}$,...,$\mathbf{z}^{(d)}$ *from the output matrix* Y *and the knowledge of the coding matrices* $D_1, \ldots, D_d$.

The outline of this section is as follows. In section 7.3.1 we explain the algorithmic steps underlying symbol recovery. The algorithm greatly parallels the developments of algorithm C1 of the previous section. In section 7.3.2 we discuss code design rules for algorithm C2. Finally, section 7.3.3 presents some algorithmic alternatives based on the same coding scheme.

7.3.1 Algorithm C2

For the presentation of the algorithm we omit the noise, i.e. $\mathbf{n_k} = 0$, later on we show how to account for noise effects. The two-step algorithm is similar to

algorithm C1. The *first step* computes an SVD of the matrix Y, see equation 7.4. We select the right null space to obtain:

$$X \cdot V^{\perp} = 0.$$

Now it remains to remove MAI in the *second step*. For each user l we define $V_l^{\perp} = D_l \cdot V^{\perp}$, such that

$$\mathbf{z}^{(l)} \cdot V_l^{\perp} = 0.$$

Since the input sequence is known to be real, we can double the number of constraints by imposing

$$\mathbf{z}^{(l)} \cdot \left[\begin{array}{cc} \Re(V_l^{\perp}) & \Im(V_l^{\perp}) \end{array} \right] = 0. \tag{7.13}$$

The coding matrices D_l should now be selected such that $\left[\begin{array}{cc} \Re(V_l^{\perp}) & \Im(V_l^{\perp}) \end{array} \right]$ has a *one-dimensional* left null space, which defines $\mathbf{z}^{(l)}$.

With noise we can write a number of equivalent minimization problems, much in the same way as we did for algorithm C1 (see equation 7.7):

$$\begin{aligned} & \min_{\hat{X}} \|\hat{X} \cdot V^{\perp}\|_F^2 \\ = & \sum_{l=1}^{d} \min_{\hat{\mathbf{z}}^{(l)}} \left\| \hat{\mathbf{z}}^{(l)} \cdot V_l^{\perp} \right\|_{\mathrm{F}}^2 \\ = & \sum_{l=1}^{d} \min_{\hat{\mathbf{z}}^{(l)}} \left\| \hat{\mathbf{z}}^{(l)} \cdot \left[\begin{array}{cc} \Re(V_l^{\perp}) & \Im(V_l^{\perp}) \end{array} \right] \right\|_{\mathrm{F}}^2 . \end{aligned}$$

The second equation exploits property 7.2.1 of the Frobenius norm and the third equation uses the fact that the symbol sequences $\mathbf{z}^{(l)}$ are real. The symbol sequence of each user can then be determined as the left singular vector corresponding to the smallest singular value of $\left[\begin{array}{cc} \Re(V_l^{\perp}) & \Im(V_l^{\perp}) \end{array} \right]$. The procedure C2 is summarized in algorithm 7.3.1.

The next section explains how to select the coding matrices D_l.

7.3.2 Code selection

In this section we show how to derive coding matrices D_l such that $\left[\begin{array}{cc} \Re(V_l^{\perp}) & \Im(V_l^{\perp}) \end{array} \right]$ has a one-dimensional left null space (no noise). First note that if $\left[\begin{array}{cc} \Re(V_l^{\perp}) & \Im(V_l^{\perp}) \end{array} \right] \in \mathbb{R}^{N \times 2(N-d)}$ has a one-dimensional null space then:

$$N \geq 2 \cdot d - 1.$$

Algorithm 7.3.1
Algorithm C2

1. *Compute the SVD of* Y*:*

$$Y = \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & \Sigma^\perp \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}.$$

2. *for* $l = 1, \ldots, d$

$$\begin{aligned} V_l^\perp &= D_l \cdot V^\perp \\ \begin{bmatrix} \Re(V_l^\perp) & \Im(V_l^\perp) \end{bmatrix} &= \widetilde{U} \cdot \widetilde{\Sigma} \cdot \widetilde{V}^T \\ \hat{\mathbf{z}}^{(l)} &= \mathcal{D}(\alpha \cdot \widetilde{U}(:, N)^T) \end{aligned}$$

end

This condition is negligible for a block processing algorithm (where N denotes the burst length), but also quite feasible for an adaptive approach (where N should be replaced by the window size).

To rewrite the rank conditions on $\begin{bmatrix} \Re(V_l^\perp) & \Im(V_l^\perp) \end{bmatrix}$ as a function of X and D_l first note that:

$$\begin{bmatrix} \Re(V_l^\perp) & \Im(V_l^\perp) \end{bmatrix} = \begin{bmatrix} \Re(D_l) & \Im(D_l) \end{bmatrix} \cdot \begin{bmatrix} \Re(V^\perp) & \Im(V^\perp) \\ -\Im(V^\perp) & \Re(V^\perp) \end{bmatrix}. \quad (7.14)$$

Now $\begin{bmatrix} \Re(V_l^\perp) & \Im(V_l^\perp) \end{bmatrix}$ has a one-dimensional left null space, if and only if this product has rank $N-1$. Applying matrix rank property 7.2.2 gives:

$$\begin{aligned} \dim(\mathrm{span}_{\mathrm{row}}(\begin{bmatrix} \Re(D_l) & \Im(D_l) \end{bmatrix}) \cap \\ \mathrm{span}_{\mathrm{col}}(\begin{bmatrix} \Re(V^\perp) & \Im(V^\perp) \\ -\Im(V^\perp) & \Re(V^\perp) \end{bmatrix})^\perp) = N - (N-1). \quad (7.15) \end{aligned}$$

Now to further rewrite this expression, we make use of the following theorem.

Theorem 7.3.1 *The row space of* X *spans the left null space of* $V^\perp$ *if and only if the row space of* $\begin{bmatrix} \Re(X) & \Im(X) \\ -\Im(X) & \Re(X) \end{bmatrix}$ *spans the left null space of* $\begin{bmatrix} \Re(V^\perp) & \Im(V^\perp) \\ -\Im(V^\perp) & \Re(V^\perp) \end{bmatrix}$.

Proof: Assume that the row space of X spans the left null space of $V^\perp$. We then show that the row space of $\begin{bmatrix} \Re(X) & \Im(X) \\ -\Im(X) & \Re(X) \end{bmatrix}$ spans the left null space of $\begin{bmatrix} \Re(V^\perp) & \Im(V^\perp) \\ -\Im(V^\perp) & \Re(V^\perp) \end{bmatrix}$. We use shorthand notation as follows

$$c_r = \Re(c),$$

$$c_i \quad = \quad \Im(c).$$

The proof consists of two parts, first we show that $\text{span}_{\text{row}}(\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]^\perp) \subset \text{span}_{\text{row}}(\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right])$, and subsequently we show that $\text{span}_{\text{row}}(\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right]) \subset \text{span}_{\text{row}}(\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]^\perp)$.

We first show that any vector in the null space of $\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]$, lies in the row space of $\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right]$. Take any vector $\left[\begin{array}{cc} \mathbf{b_1} & \mathbf{b_2} \end{array}\right]$ in the left null space of $\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]$:

$$\left[\begin{array}{cc} \mathbf{b_1} & \mathbf{b_2} \end{array}\right] \left[\begin{array}{cc} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{array}\right] = 0$$

such that

$$(\mathbf{b_1} + i\mathbf{b_2})V^\perp = 0.$$

Hence, the vector $\mathbf{b_1} + i\mathbf{b_2}$ lies in the null space of $V^\perp$. But since X spans the left null space of $V^\perp$, $\mathbf{b_1} + i\mathbf{b_2}$ lies in the row space of X. So, we can write

$$\mathbf{b_1} + i\mathbf{b_2} = a_1 \cdot \mathbf{x_1} + a_2 \cdot \mathbf{x_2} + \cdots + a_d \cdot \mathbf{x_d}$$

or

$$\begin{aligned} \mathbf{b_1} &= a_{1r}\mathbf{x_{1}}_r - a_{1i}\mathbf{x_{1}}_i + a_{2r}\mathbf{x_{2}}_r - a_{2i}\mathbf{x_{2}}_i + \ldots + a_{dr}\mathbf{x_{d}}_r - a_{di}\mathbf{x_{d}}_i, \\ \mathbf{b_2} &= a_{1r}\mathbf{x_{1}}_i + a_{1i}\mathbf{x_{1}}_r + a_{2r}\mathbf{x_{2}}_i + a_{2i}\mathbf{x_{2}}_r + \ldots + a_{dr}\mathbf{x_{d}}_i + a_{di}\mathbf{x_{d}}_r, \end{aligned}$$

such that

$$\left[\begin{array}{cc} \mathbf{b_1} & \mathbf{b_2} \end{array}\right] = \left[\begin{array}{cccccc} a_{1r} & \ldots & a_{dr} & a_{1i} & \ldots & a_{di} \end{array}\right] \left[\begin{array}{cc} X_r & X_i \\ -X_i & X_r \end{array}\right].$$

Hence, $\left[\begin{array}{cc} \mathbf{b_1} & \mathbf{b_2} \end{array}\right]$ lies in the row space of $\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right]$.

We now show that any vector in the row space of $\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right]$, lies in the null space of $\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]$. Take any vector $\left[\begin{array}{cc} \mathbf{b_1} & \mathbf{b_2} \end{array}\right]$ in the row space of $\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right]$. Then the vector $\mathbf{b_1} + i\mathbf{b_2}$ lies in the row space of X (similar derivation as above). But since X spans the left null space of $V^\perp$, $\mathbf{b_1} + i\mathbf{b_2}$ lies in the null space of $V^\perp$. So, $\left[\begin{array}{cc} \mathbf{b_1} & \mathbf{b_2} \end{array}\right]$ lies in the null space of $\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]$ (similar derivation as above).

In a similar way it can be shown that if the row space of $\left[\begin{smallmatrix} X_r & X_i \\ -X_i & X_r \end{smallmatrix}\right]$ equals the left null space of $\left[\begin{smallmatrix} V_r^\perp & V_i^\perp \\ -V_i^\perp & V_r^\perp \end{smallmatrix}\right]$ then the row space of X will equal the null space of $V^\perp$. □

Equation 7.15 can thus be rewritten as

$$\dim(\text{span}_{\text{row}}(\begin{bmatrix} \Re(D_l) & \Im(D_l) \end{bmatrix}) \cap \text{span}_{\text{row}}(\begin{bmatrix} \Re(X) & \Im(X) \\ -\Im(X) & \Re(X) \end{bmatrix}) = 1. \tag{7.16}$$

We can further rewrite this equation using the dimension law of vector spaces (equation 3.6):

$$1 = N + 2 \cdot d - \text{rank}(\underbrace{\begin{bmatrix} \Re(X) & \Im(X) \\ -\Im(X) & \Re(X) \\ \Re(D_l) & \Im(D_l) \end{bmatrix}}_{F^{(l)}}) \tag{7.17}$$

or

$$\text{rank}(F^{(l)}) = N + 2 \cdot d - 1. \tag{7.18}$$

Equation 7.18 is equivalent with the requirement that the matrix $F^{(l)}$ in which the row $\begin{bmatrix} \Re(\mathbf{x}^{(l)}) & \Im(\mathbf{x}^{(l)}) \end{bmatrix}$ has been removed $F^{(l)}_{-l}$, has full row rank.

Example 7.3.2 To illustrate the code selection, assume a complex constant modulus coding with two BPSK users ($d = 2$) having the following coding matrices ($N = 3$):

$$D_1 = \begin{bmatrix} e^{i\theta_{11}} & & \\ & e^{i\theta_{12}} & \\ & & e^{i\theta_{13}} \end{bmatrix} \qquad D_2 = \begin{bmatrix} e^{i\theta_{21}} & & \\ & e^{i\theta_{22}} & \\ & & e^{i\theta_{23}} \end{bmatrix}$$

One should thus check the rank of the matrices:

$$F^{(1)}_{-1} = \begin{bmatrix} \pm\cos\theta_{21} & \pm\cos\theta_{22} & \pm\cos\theta_{23} & \pm\sin\theta_{21} & \pm\sin\theta_{22} & \pm\sin\theta_{23} \\ \mp\sin\theta_{11} & \mp\sin\theta_{12} & \mp\sin\theta_{13} & \pm\cos\theta_{11} & \pm\cos\theta_{12} & \pm\cos\theta_{13} \\ \mp\sin\theta_{21} & \mp\sin\theta_{22} & \mp\sin\theta_{23} & \pm\cos\theta_{21} & \mp\cos\theta_{22} & \mp\cos\theta_{23} \\ \cos\theta_{11} & & & \sin\theta_{11} & & \\ & \cos\theta_{12} & & & \sin\theta_{12} & \\ & & \cos\theta_{13} & & & \sin\theta_{13} \end{bmatrix}$$

$$F^{(2)}_{-2} = \begin{bmatrix} \pm\cos\theta_{11} & \pm\cos\theta_{12} & \pm\cos\theta_{13} & \pm\sin\theta_{11} & \pm\sin\theta_{12} & \pm\sin\theta_{13} \\ \mp\sin\theta_{11} & \mp\sin\theta_{12} & \mp\sin\theta_{13} & \pm\cos\theta_{11} & \pm\cos\theta_{12} & \pm\cos\theta_{13} \\ \mp\sin\theta_{21} & \mp\sin\theta_{22} & \mp\sin\theta_{23} & \pm\cos\theta_{21} & \mp\cos\theta_{22} & \mp\cos\theta_{23} \\ \cos\theta_{21} & & & \sin\theta_{21} & & \\ & \cos\theta_{22} & & & \sin\theta_{22} & \\ & & \cos\theta_{23} & & & \sin\theta_{23} \end{bmatrix}$$

Calculation yields that these two matrices will have full rank if:

$$\pm\sin(-\theta_{21}+\theta_{23}+\theta_{11}-\theta_{13})\pm\sin(-\theta_{21}+\theta_{22}+\theta_{11}-\theta_{12}) \\ \pm\sin(\theta_{23}-\theta_{22}+\theta_{12}-\theta_{13})\neq 0 \quad (7.19)$$

for all eight possible sign combinations. From this expression it can be derived that mapping BPSK symbols onto QPSK symbols i.e. $\theta_{kl}\in\{0,\frac{\pi}{2},\pi,\frac{3\cdot\pi}{2}\}$, will not yield a valid coding scheme. I.e. for each selection of θ_{kl} at least one of the sign combinations will lead to a zero sum of the terms in equation 7.19. This means that BPSK input sequences can be found which make it impossible to obtain one-dimensional problems after exploiting the coding information. On the other hand, when the code phase angles θ_{kl} are selected in a *random* way the rank condition of equation 7.19 is very likely to be satisfied. □

For $d>2$ and/or $N>3$, it becomes infeasible to obtain analytic expressions as the one presented above. For a specific code, rank conditions could be checked by enumeration, but even that strategy becomes infeasible for increasing d and/or N. However, motivated by the two user case we can state that random codes will very likely satisfy the rank condition imposed by equation 7.18. This will be corroborated by the simulation results of chapter 8, where we select *random complex constant modulus codes.*

As for algorithm C1, optimal code design should proceed in a statistical framework, which is beyond the scope of this work.

7.3.3 Further discussion

In this section we present a number of different formulations of the algorithm C2 of section 7.3.1. These alternative formulations are also readily applicable to the algorithms C1 and C3. The key observation is that the row space of X can be estimated from the signal subspace of Y: V^{sH} (see equation 7.4). Then analog to Theorem 7.3.1 we have that:

Theorem 7.3.3 *X spans the same row space as V^{sH} if and only if $\left[\begin{smallmatrix}\Re(X) & \Im(X)\\ -\Im(X) & \Re(X)\end{smallmatrix}\right]$ spans the same row space as $\left[\begin{smallmatrix}\Re(V^{sH}) & \Im(V^{sH})\\ -\Im(V^{sH}) & \Re(V^{sH})\end{smallmatrix}\right]$.*

Proof: The proof is analog to the proof of Theorem 7.3.1. □

The above theorem now leads to the following alternative problem formulations. Equation 7.16 reads:

$$\dim(\mathrm{span}_{\mathrm{row}}(\begin{bmatrix}\Re(D_l) & \Im(D_l)\end{bmatrix})\cap\mathrm{span}_{\mathrm{row}}(\begin{bmatrix}\Re(V^{sH}) & \Im(V^{sH})\\ -\Im(V^{sH}) & \Re(V^{sH})\end{bmatrix})=1 \quad (7.20)$$

and the vector $\left[\begin{array}{cc} \Re(\mathbf{x}^{(l)}) & \Im(\mathbf{x}^{(l)}) \end{array}\right]$ lies in the intersection of the two row spaces. Without noise, an algorithm based on equation 7.20 will give perfect symbol estimates. With noise, it will give results identical to that of algorithm C2. However, this alternative problem formulation is computationally not more attractive than algorithm C2, therefore we discard this approach further on.

Next, we approach the problem from an equalizer design point of view. Equalizer design could start from the following minimization problem:

$$\min_{\hat{G},\hat{X}} \left\| \hat{G} \cdot V^{sH} - \hat{X} \right\|_{\mathrm{F}}^2 \tag{7.21}$$

where $\hat{G}$ denotes a matrix equalizer. We take the convention that the lth row of $\hat{G}$ is an equalizer for the lth user, i.e. $\hat{G}(l,:) = \hat{\mathbf{g}}^{(l)}$ and $\hat{\mathbf{g}}^{(l)} \cdot V^{sH} = \hat{\mathbf{x}}^{(l)}$. Equation 7.21 can then be rewritten into the following set of equivalent minimization problems:

$$\begin{aligned}
&\min_{\hat{G},\hat{\mathbf{z}}^{(l)},l=1,\ldots,d} \| \hat{G} \cdot V^{sH} - \left[\begin{array}{c} \hat{\mathbf{z}}^{(1)} \cdot D_1 \\ \vdots \\ \hat{\mathbf{z}}^{(d)} \cdot D_d \end{array}\right] \|_F^2 \\
&= \sum_{l=1}^{d} \min_{\hat{\mathbf{g}}^{(l)},\hat{\mathbf{z}}^{(l)}} \left\| \left[\begin{array}{cc} \hat{\mathbf{g}}^{(l)} & \hat{\mathbf{z}}^{(l)} \end{array}\right] \left[\begin{array}{c} V^{sH} \\ -D_l \end{array}\right] \right\|_{\mathrm{F}}^2 \\
&= \sum_{l=1}^{d} \min_{\hat{\mathbf{g}}^{(l)},\hat{\mathbf{z}}^{(l)}} \| \left[\begin{array}{ccc} \Re(\hat{\mathbf{g}}^{(l)}) & \Im(\hat{\mathbf{g}}^{(l)}) & \hat{\mathbf{z}}^{(l)} \end{array}\right] \left[\begin{array}{cc} \Re(V^{sH}) & \Im(V^{sH}) \\ -\Im(V^{sH}) & \Re(V^{sH}) \\ -\Re(D_l) & -\Im(D_l) \end{array}\right] \|_F^2.
\end{aligned} \tag{7.22}$$

In the last equation we exploited the fact that the symbol sequences to be estimated belong to a real alphabet. Note that the last equation is similar to the requirement that the matrix $F^{(l)}$ of equation 7.18 has a one-dimensional left null space (without noise). This null space is formed by $\left[\begin{array}{ccc} \Re(\mathbf{g}^{(l)}) & \Im(\mathbf{g}^{(l)}) & \mathbf{z}^{(l)} \end{array}\right]$, where $\mathbf{g}^{(l)}$ can be interpreted as an equalizer for the lth user. The equalizer and/or symbol sequence can be estimated as the left singular vector corresponding to smallest singular value of the above matrix. Further note this equalizer is reminiscent of the deterministic equalizer of [84], developed for DS CDMA systems. The above cost function can be transformed, to a computationally more attractive scheme in the following way.

$$\sum_{l=1}^{d} \min_{\hat{\mathbf{g}}^{(l)},\hat{\mathbf{z}}^{(l)}} \left\| \hat{\mathbf{g}}^{(l)} \cdot V^{sH} - \hat{\mathbf{z}}^{(l)} \cdot D_l \right\|_{\mathrm{F}}^2 \tag{7.23}$$

$$= \sum_{l=1}^{d} \min_{\hat{\mathbf{g}}^{(l)},\hat{\mathbf{z}}^{(l)}} \left\| \hat{\mathbf{g}}^{(l)} \cdot V^{sH} \cdot D_l^{-1} - \hat{\mathbf{z}}^{(l)} \right\|_{\mathrm{F}}^2 \tag{7.24}$$

$$= \sum_{l=1}^{d} \min_{\hat{\mathbf{g}}^{(l)}} \left\| \Im(\hat{\mathbf{g}}^{(l)} \cdot V^{sH} \cdot D_l^{-1}) \right\|_{\mathrm{F}}^2 \tag{7.25}$$

$$= \sum_{l=1}^{d} \min_{\hat{\mathbf{g}}_l} \| \begin{bmatrix} \Re(\hat{\mathbf{g}}^{(l)}) & \Im(\hat{\mathbf{g}}^{(l)}) \end{bmatrix} \cdot \begin{bmatrix} \Im(V^{sH} \cdot D_l^{-1}) \\ \Re(V^{sH} \cdot D_l^{-1}) \end{bmatrix} \|_F^2 \qquad (7.26)$$

In 7.24 we used the fact that D_l is unitary for constant modulus coding. In equation 7.25 we exploited that $\mathbf{z}^{(l)}$ is real. In the last equation we only need to estimate the equalizer and have hence strongly reduced problem dimensions: the equalizer has only length $2 \cdot d$. Once the equalizer coefficients are estimated, the symbol sequence $\hat{\mathbf{z}}^{(l)}$ can be estimated as $\hat{\mathbf{z}}^{(l)} = \Re(\hat{\mathbf{g}}^{(l)} \cdot V^{sH} \cdot D_l^{-1})$.

A final variant can be derived in the following way. Assume $\hat{\mathbf{g}}^{(l)}$ known, then we obtain from equation 7.22:

$$\begin{aligned} \hat{\mathbf{z}}^{(l)} &= \begin{bmatrix} \Re(\hat{\mathbf{g}}^{(l)}) & \Im(\hat{\mathbf{g}}^{(l)}) \end{bmatrix} \cdot \begin{bmatrix} \Re(V^{sH}) & \Im(V^{sH}) \\ -\Im(V^{sH}) & \Re(V^{sH}) \end{bmatrix} \cdot \begin{bmatrix} \Re(D_l) & \Im(D_l) \end{bmatrix}^{\dagger} \\ &= \begin{bmatrix} \Re(\hat{\mathbf{g}}^{(l)}) & \Im(\hat{\mathbf{g}}^{(l)}) \end{bmatrix} \cdot \begin{bmatrix} \Re(V^{sH}) & \Im(V^{sH}) \\ -\Im(V^{sH}) & \Re(V^{sH}) \end{bmatrix} \cdot \begin{bmatrix} \Re(D_l)^H \\ \Im(D_l)^H \end{bmatrix} \end{aligned}$$

where we used the fact that for constant modulus coding D_l is unitary. Substituting this expression in equation 7.22 gives the following optimization problem in $\hat{\mathbf{g}}^{(l)}$:

$$\min_{\hat{\mathbf{g}}^{(l)}} \| \begin{bmatrix} \Re(\hat{\mathbf{g}}^{(l)}) & \Im(\hat{\mathbf{g}}^{(l)}) \end{bmatrix} \cdot \begin{bmatrix} \Re(V^{sH}) & \Im(V^{sH}) \\ -\Im(V^{sH}) & \Re(V^{sH}) \end{bmatrix} \cdot (I - \begin{bmatrix} \Re(D_l)^H \\ \Im(D_l)^H \end{bmatrix} \cdot \begin{bmatrix} \Re(D_l) & \Im(D_l) \end{bmatrix}) \|_F^2.$$

Remark 7.3.4 All equalizer algorithms derived above will have a performance identical to that of the algorithm C2 but may be more attractive from a computational point of view. The modifications involved are also straightforwardly applied to the algorithms C1 and C3 (except for the algorithm based on equation 7.26).

Remark 7.3.5 A rough approximation of the row space of X can also by obtained from the row space of Y (this approximation being exact only in the noiseless case). Equation 7.26 e.g. could then be replaced by:

$$\sum_{l=1}^{d} \min_{\tilde{\mathbf{g}}^{(l)}} \| \begin{bmatrix} \Re(\tilde{\mathbf{g}}^{(l)}) & \Im(\tilde{\mathbf{g}}^{(l)}) \end{bmatrix} \cdot \begin{bmatrix} \Im(Y \cdot D_l^{-1}) \\ \Re(Y \cdot D_l^{-1}) \end{bmatrix} \|_F^2.$$

The last equation has the advantage of not longer requiring an (computationally expensive) SVD. Note that when $M > d$, one should avoid choosing an equalizer $\tilde{\mathbf{g}}^{(l)}$ in the null space of $Y \cdot D_l^{-1}$, because that gives $\tilde{\mathbf{z}}^{(l)} = \Re(\tilde{\mathbf{g}}^{(l)} \cdot Y \cdot D_l^{-1}) = 0$.

7.4 Algorithm C3

In this section we present a third linear coding scheme. It leads to an algorithm C3 which is applicable for any modulation format and which does not require any coding overhead. We assume transmit diversity, i.e. each user is equipped with at least two transmit antennas. Through each antenna the same symbol sequence is transmitted but coded in a different way. This coding does not require any extra bandwidth and allows to remove MAI in a non-iterative way. So in some sense the algorithm presented here generalizes the results of the previous section where a coding based signal separation algorithm was proposed for BPSK constellations (without exploiting transmit diversity).

The outline of this section is as follows. In section 7.4.1 the data model of section 7.1 is extended to incorporate transmit diversity. Section 7.4.2 presents the algorithm and section 7.4.3 details code selection criteria.

7.4.1 Data model

Assume that d sources each transmit digital symbols at the same symbol rate from M_t transmit antennas to an M_r element antenna array. Retaining the assumptions that sources are synchronized and that the multi-path delay spread is negligible, the sampled received (vector) signal can be expressed as:

$$\mathbf{y_k} = A \cdot \mathbf{x_k} + \mathbf{n_k}. \tag{7.27}$$

Here

$$\begin{aligned} \mathbf{y_k} &= \left[\begin{array}{ccc} y_1(k) & \cdots & y_{M_r}(k) \end{array} \right]^T \\ \mathbf{x_k} &= \left[\begin{array}{ccc} x^{(1)}(k) & \cdots & x^{(dM_t)}(k) \end{array} \right]^T \\ \mathbf{n_k} &= \left[\begin{array}{ccc} n_1(k) & \cdots & n_{M_r}(k) \end{array} \right]^T. \end{aligned}$$

The mixing matrix $A \in \mathbb{C}^{M_r \times dM_t}$ is assumed to have full column rank dM_t (which implies $M_r \geq dM_t$).

Remark 7.4.1 Note that with respect to the data model of equation 7.1 we have more stringent constraints both for the number of transmitter antennas M_t and the number of receiver antennas M_r: $M_t \geq 2$ and $M_r \geq d \cdot M_t$. *We need M_t times more receiver antennas with $M_t \geq 2$.*

Analog to the data model of section 7.1, we obtain the following (noiseless) data model:

$$\left[\begin{array}{ccc} \mathbf{y_1} & \cdots & \mathbf{y_N} \end{array} \right] = A \cdot \left[\begin{array}{ccc} \mathbf{x_1} & \cdots & \mathbf{x_N} \end{array} \right]$$

$$Y = A \cdot X.$$

We further define the $d \cdot M_t$ transmitted sequences $\mathbf{x}^{(1)},\cdots,\mathbf{x}^{(dM_t)}$ as the $d \cdot M_t$ rows of X:

$$X = \begin{bmatrix} \boxed{\mathbf{x}^{(1)}} \\ \vdots \\ \boxed{\mathbf{x}^{(dM_t)}} \end{bmatrix}.$$

7.4.2 Algorithm C3

Assume that d sources each want to transmit a symbol sequence $\mathbf{z}^{(l)}$. The question rises how to construct the transmitted sequences $\mathbf{x}^{(l)}$ from the symbol sequences $\mathbf{z}^{(l)}$ such that decoding is facilitated at the receiver side. For a user l we define M_t coding matrices D_{lk} as follows:

$$\mathbf{x}^{((l-1)\cdot M_t+k)} = \mathbf{z}^{(l)} \cdot D_{lk} \qquad k = 1,\ldots,M_t$$

where the coding matrices are complex diagonal matrices. On each of the M_t transmit antennas for user l we thus transmit the same symbol sequence but coded by a different coding matrix. Additionally, we put the following power constraint on the coding matrices:

$$|D_{lk}(r,r)|^2 = 1/M_t \qquad \text{for} \qquad \begin{array}{l} r = 1,\ldots,N \\ l = 1,\ldots,d \\ k = 1,\ldots,M_t \end{array} \tag{7.28}$$

such that the coding does not require any extra power. The total transmitted power equals the power that would have been transmitted using a single antenna without coding. *The aim of the algorithm is to recover the symbol sequences* $\mathbf{z}^{(1)},\ldots,\mathbf{z}^{(d)}$ *from the output matrix* Y *and the knowledge of the coding matrices* $D_{11}, \ldots, D_{dM_t}$.

For the presentation of the algorithm we will omit the noise, i.e. $\mathbf{n_k} = 0$, later on we will then show how to account for noise effects. Analog to the algorithms C1 and C2, the first step consists of an SVD of Y:

$$\begin{aligned} Y &= U \cdot \Sigma \cdot V^H \\ &= \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}. \end{aligned}$$

We select the noise space $V^\perp$. If A and X have respectively full column (which requires $M_r \geq M_t \cdot d$) and row rank (which requires $N \geq M_t \cdot d$), then $V^\perp$ has exactly a $M_t \cdot d$-dimensional left null space and X spans that null space:

$$X \cdot V^\perp = 0.$$

Now it remains to remove MAI in the *second step*. For each user l we define

$$V_l^{\perp} = \left[\begin{array}{ccc} D_{l1} \cdot V^{\perp} & \cdots & D_{lM_t} \cdot V^{\perp} \end{array} \right] \tag{7.29}$$

such that

$$\mathbf{z}^{(l)} \cdot V_l^{\perp} = 0. \tag{7.30}$$

The coding matrices should now be selected such that $V_l^{\perp}$ has a one-dimensional left null space, which defines $\mathbf{z}^{(l)}$.

With noise, we want to minimize

$$\begin{aligned} & \min_{\hat{X}} \| \hat{X} \cdot V^{\perp} \|_F^2 \\ = & \min_{\hat{\mathbf{z}}^{(1)}, \ldots, \hat{\mathbf{z}}^{(d)}} \| \left[\begin{array}{c} \hat{\mathbf{z}}^{(1)} \cdot D_{11} \\ \vdots \\ \hat{\mathbf{z}}^{(1)} \cdot D_{1M_t} \\ \hline \vdots \\ \hline \hat{\mathbf{z}}^{(d)} \cdot D_{d1} \\ \vdots \\ \hat{\mathbf{z}}^{(d)} \cdot D_{dM_t} \end{array} \right] \cdot V^{\perp} \|_F^2. \end{aligned}$$

Using Frobenius norm properties 7.2.1, the last equation is separable into d subproblems:

$$\begin{aligned} & \sum_{l=1}^{d} \min_{\hat{\mathbf{z}}^{(l)}} \| \left[\begin{array}{c} \hat{\mathbf{z}}^{(l)} \cdot D_{l1} \\ \vdots \\ \hat{\mathbf{z}}^{(l)} \cdot D_{lM_t} \end{array} \right] \cdot V^{\perp} \|_F^2 \\ = & \sum_{l=1}^{d} \min_{\hat{\mathbf{z}}^{(l)}} \| \hat{\mathbf{z}}^{(l)} \cdot \left[\begin{array}{ccc} D_{l1} \cdot V^{\perp} & \cdots & D_{lM_t} \cdot V^{\perp} \end{array} \right] \|_F^2 \\ = & \sum_{l=1}^{d} \min_{\hat{\mathbf{z}}^{(l)}} \left\| \hat{\mathbf{z}}^{(l)} \cdot V_l^{\perp} \right\|_{\mathrm{F}}^2. \end{aligned}$$

The symbol sequence of each user can then be determined as the left singular vector corresponding to the smallest singular value of $V_l^{\perp}$. The algorithm is summarized as algorithm 7.4.2.

Algorithm 7.4.2
Algorithm C3

1. *Compute the SVD of* Y*:*

$$Y = \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & \Sigma^\perp \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}.$$

2. *for* $l = 1, \ldots, d$

$$\begin{aligned} V_l^\perp &= \begin{bmatrix} D_{l1} \cdot V^\perp & \cdots & D_{lM_t} \cdot V^\perp \end{bmatrix} \\ V_l^\perp &= \widetilde{U} \cdot \widetilde{\Sigma} \cdot \widetilde{V}^H \\ \hat{\mathbf{z}}^{(l)} &= \mathcal{D}(\alpha \cdot \widetilde{U}(:, N)^H) \end{aligned}$$

end

7.4.3 Code selection

If $V_l^\perp \in \mathbb{C}^{N \times (M_t \cdot (N - M_t \cdot d))}$ (as defined in equation 7.29) has a one-dimensional null space, then:

$$(M_t - 1) \cdot N \geq M_t^2 \cdot d - 1.$$

We again rewrite the rank condition as a function of X and D_{lk}. First $V_l^\perp$ can be rewritten as

$$V_l^\perp = \begin{bmatrix} D_{l1} & \cdots & D_{lM_t} \end{bmatrix} \cdot \underbrace{\begin{bmatrix} V^\perp & & & \\ & V^\perp & & \\ & & \ddots & \\ & & & V^\perp \end{bmatrix}}_{\overline{V}^\perp}. \tag{7.31}$$

From this point on we take the convention that if A is an $m \times n$ matrix then $\overline{A}$ is an $M_t \cdot m \times M_t \cdot n$ matrix obtained by stacking copies of A as in the above example.

Now requiring that $V_l^\perp$ has rank $N - 1$ is equivalent to:

$$\dim(\mathrm{span}_{\mathrm{row}}(\begin{bmatrix} D_{l1} & \cdots & D_{lM_t} \end{bmatrix}) \cap \mathrm{span}_{\mathrm{col}}(\overline{V}^\perp) = N - (N - 1) \tag{7.32}$$

where we applied matrix rank property 7.2.2. To further rewrite this expression, we make use of the following theorem.

Theorem 7.4.2 *The row space of X spans the left null space of $V^{\perp}$ if and only if the row space of $\overline{X} = \begin{bmatrix} X & & \\ & \ddots & \\ & & X \end{bmatrix}$ spans the left null space of $\overline{V}^{\perp}$.*

Proof: The proof is straightforward. □

Equation (7.32) can then be rewritten as

$$\dim(\text{span}_{\text{row}}(\begin{bmatrix} D_{l1} & \cdots & D_{lM_t} \end{bmatrix}) \cap \text{span}_{\text{row}}(\overline{X}) = 1. \tag{7.33}$$

We can further rewrite this equation by using the dimension law of vector spaces (equation 3.6):

$$1 = N + M_t^2 \cdot d - \text{rank}(\begin{bmatrix} & \overline{X} & \\ D_{l1} & \cdots & D_{lM_t} \end{bmatrix}) \tag{7.34}$$

or

$$\text{rank}(\begin{bmatrix} & \overline{X} & \\ D_{l1} & \cdots & D_{lM_t} \end{bmatrix}) = N + M_t^2 \cdot d - 1. \tag{7.35}$$

As for algorithm C2, complex *random* codes are very likely to satisfy the above rank conditions. This will be corroborated by the simulation results of chapter 8, where we select random complex constant modulus codes.

Remark 7.4.3 If the modulation format is real, it might be advantageous to split the received signal into its real and imaginary component as in equation 7.12. In that case real coding matrices are used instead of complex codes. The real coding matrices are still chosen in such a way that $\sum_{k=1}^{M_t} D_{lk}(r,r)^2 = 1$, $r = 1, \ldots, N$, $l = 1, \ldots, d$, such that the same amount of power is transmitted as in the case where only one antenna is used.

Remark 7.4.4 Equalizer versions of algorithm C3 are constructed in the same way as for algorithm C2 (see section 7.3.3).

7.5 Algorithm C1, C2 and C3 with ISI

In the previous sections we have presented the algorithms C1, C2 and C3 in a blind signal separation scenario. In this section we show how these algorithms are modified in the presence of ISI. We have deliberately delayed the presentation of this scenario until all algorithms have been presented because similar

modifications are involved for the three algorithms. We show that coding enables the identification of the symbol sequences of the different users without identifying their individual channel lengths (which usually involves a cumbersome iterative procedure, see e.g. [81]). Furthermore it is demonstrated how coding provides robustness against system order overestimation (underestimation of the dimension of the null space).

If all users have equal channel lengths then the algorithms C1, C2 and C3 are easily augmented with a preprocessing step removing ISI as in the algorithms [80, 81] (see also section 5.1). However, when users have different channel lengths the same problems occur as described in section 6.2 for the multi-user subspace + Viterbi algorithm. As for the subspace + Viterbi algorithm, we can also rely on coding information to remove ISI.

We introduce the problem via a small example (that will also be used in the simulation examples of chapter 8):

Example 7.5.1 Assume $d = 2$ users, with channel lengths $L_1 = 4$, $L_2 = 2$ and $M_t = 1$ (the extension to $M_t > 1$ and algorithm C3 is straightforward). We again use the multi-user data model of equation 2.13. In a first step we compute the SVD of $Y_{L_2+1|L_2+i}$:

$$
\begin{aligned}
Y_{L_2+1|L_2+i} &= \left[\begin{array}{cc} \mathcal{H}_i^{(1)} & \mathcal{H}_i^{(2)} \end{array} \right] \cdot \left[\begin{array}{ccc} x^{(1)}(L_2-L_1+1) & \cdots & x^{(1)}(L_2-L_1+j) \\ \vdots & & \vdots \\ x^{(1)}(L_2+i) & \cdots & x^{(1)}(L_2+i+j-1) \\ \hline x^{(2)}(1) & \cdots & x^{(2)}(j) \\ \vdots & & \vdots \\ x^{(2)}(L_2+i) & \cdots & x^{(2)}(L_2+i+j-1) \end{array} \right] \\
&= \left[\begin{array}{cc} U^s & U^\perp \end{array} \right] \cdot \left[\begin{array}{cc} \Sigma^s & 0 \\ 0 & \Sigma^\perp \end{array} \right] \cdot \left[\begin{array}{c} V^{sH} \\ V^{\perp H} \end{array} \right] .
\end{aligned}
$$

Now define the matrix G^r as:

$$
G^r \triangleq \left[\begin{array}{ccccc} \boxed{V^\perp} & 0 & \cdots & 0 \\ 0 & \boxed{V^\perp} & \ddots & \\ & & & \boxed{V^\perp} \end{array} \right] . \tag{7.36}
$$

Here the superscript r denotes the number of shifts that are applied, i.e. r is

the number of block columns of G^r. For the example above we obtain:

$$\left[\begin{array}{ccc} x^{(1)}(-1) & \cdots & x^{(1)}(N-2) \\ x^{(1)}(0) & \cdots & x^{(1)}(N-1) \\ x^{(1)}(1) & \cdots & x^{(1)}(N) \\ \hline x^{(2)}(1) & \cdots & x^{(2)}(N) \end{array}\right] \cdot G^{i+L_2} = 0$$

$$\left[\begin{array}{ccc} x^{(1)}(-1) & \cdots & x^{(1)}(N) \end{array}\right] \cdot G^{i+L_1} = 0.$$

The matrix G^{i+L_2} has a four-dimensional null space with two desired and two undesired solutions. The matrix G^{i+L_1} has a one-dimensional null space where all information on the second symbol sequence is lost. If L_1 were known, $\mathbf{x}^{(1)}$ could be detected from G^{i+L_1}. In a second step $\mathbf{x}^{(2)}$ could be detected from G^{i+L_2} using the already estimated $\mathbf{x}^{(1)}$. We will take a different approach here and identify both symbol sequences from G^{i+L_2}. □

From here on we assume $L_{min} = \min_l L_l$ to be known (the extension to $L_{min} = 0$, no lower bound is known, is straightforward). The symbol detection strategy proceeds as follows.

- construct $G^{i+L_{min}}$
- depending on the coding scheme, construct for each user l:

  C1 $K_l = D_l \cdot G^{i+L_{min}}$

  C2 $K_l = \left[\begin{array}{cc} \Re(D_l \cdot G^{i+L_{min}}) & \Im(D_l \cdot G^{i+L_{min}}) \end{array}\right]$

  C3 $K_l = \left[\begin{array}{ccc} D_{l1} \cdot G^{i+L_{min}} & \cdots & D_{lM_t} \cdot G^{i+L_{min}} \end{array}\right]$
- estimate the lth transmitted symbol sequence as the left null vector of K_l

The coding matrices should thus be constructed such that K_l has a one-dimensional left null space. Coding is then used both for (partial) ISI and MAI rejection. In the construction of $G^{i+L_{min}}$ the Hankel structure of the input matrices is exploited. In the second step the problem is considered as an MAI problem where the remaining ISI is considered as MAI without exploiting the remaining Hankel structure. Hence in the second step we obtain a blind source separation problem with $d + \sum_{l=1}^{d}(L_l - L_{min})$ sources (for algorithm C3 we have $dM_t + \sum_{l=1}^{dM_t}(L_l - L_{min})$ sources). Coding matrices should be designed to cope with these additional interferers. Rank conditions are easily constructed as in equations 7.11, 7.18 and 7.35.

Example 7.5.2 Resuming example 7.5.1, for algorithm C2, the symbol sequences in the null space of G^{i+L_2} have the following coding matrices: D_1, D_2, $\left[\begin{array}{cc} 0 & D_1(:, 1:N-1) \end{array}\right]$,$\left[\begin{array}{ccc} 0 & 0 & D_1(:, 1:N-2) \end{array}\right]$. Due to the pseudo random

character of the coding matrices no corrections have to be applied to provide additional ISI protection. The same applies to algorithm C3.

For algorithm C1 the coding matrices of the symbol sequences in the null space of the matrix G^{i+L_2} are $\left[\begin{array}{cc} I & D_1' \end{array}\right]$, $\left[\begin{array}{cc} I & D_2' \end{array}\right]$, $\left[\begin{array}{ccc} 0 & I & D_1'(:1:p-1) \end{array}\right]$, $\left[\begin{array}{ccc} 0 & 0 & I & D_1'(:,1:p-2) \end{array}\right]$. Careful code design should be done in order to solve the ISI problem. As in the source separation problem, it is useful to make the diagonal entries of the coding matrices to differ in many positions. For the channels of example 7.5.1 the following coding matrices can be used:

$$\begin{aligned} D_1' &= \left[\begin{array}{cccccc} \mathbf{e_1} & -\mathbf{e_2} & -\mathbf{e_3} & \mathbf{e_4} & -\mathbf{e_5} & -\mathbf{e_6} \end{array}\right] \\ D_2' &= -D_1' \\ D_l &= \left[\begin{array}{cc} I & D_l' \end{array}\right] \qquad l = 1, 2, \end{aligned}$$

where $\mathbf{e_i}$ is the ith column of the $(N-p)\times(N-p)$ identity matrix. The coding pattern repeats itself every three coding symbols. This coding provides MAI protection and ISI protection up to order two. In the simulation examples of the next chapter we will use a generalized form of the above coding scheme where the coding pattern is repeated several times. □

As will be shown in the simulation results of chapter 8, relying on coding (instead of subspace information) to additionally remove ISI does not necessarily lead to a performance loss: coding information is known exactly while subspace information is only known approximately (due to noise). Also note that in the above procedure the symbol sequences of the different users are estimated based only on a lower bound for the channel lengths, without identifying the individual channel lengths for each user separately.

Finally, we spend a few words on robustness against order detection problems. So far we have assumed that the dimension of the null space $V^\perp$, $r_{V^\perp}$ was detected correctly. The dimension of this null space is $r_{V^\perp} = j - d\cdot i - \sum_{l=1}^{d} L_l$ (for algorithm C3 $r_{V^\perp} = j - dM_t i - \sum_{l=1}^{dM_t} L_l$). If $r_{V^\perp}$ is underestimated, the null space of $V^\perp$ will be larger than the desired one but the correct solution still lies in the null space of $V^\perp$. When constructing $G^{i+L_{min}}$ we impose a Hankel structure on the solution. Simulation results (without noise) show that imposing the Hankel structure is mostly enough to reduce the null space dimension of $G^{i+L_{min}}$ to $d+\sum_{l=1}^{d}(L_l - L_{min})$ (for algorithm C3 $dM_t + \sum_{l=1}^{dM_t}(L_l - L_{min})$) even when $r_{V^\perp}$ is underestimated. $G^{i+L_{min}}$ contains $(i+L_{min})\cdot r_{V^\perp}$ equations in N unknowns and is usually largely overdetermined. Even when the null space dimension of $G^{i+L_{min}}$ is larger than $d+\sum_{l=1}^{d}(L_l - L_{min})$ (for algorithm C3 $dM_t + \sum_{l=1}^{dM_t}(L_l - L_{min})$), the algorithms can still perform successfully by considering the extra solutions as interferers which are canceled by the coding. As will be shown in the simulation results of chapter 8 the algorithms C1, C2 and C3 are fairly robust against an underestimation of the null space dimension (overestimation of the system order).

7.6 Adaptive algorithms based on C2 and C3

Algorithm C1 can be made adaptive but requires quite some modifications. We will discard this approach and instead we discuss an adaptive implementation based on the coding scheme of algorithm C2 (algorithm C3 is straightforwardly modified in a similar way). As an example, we explain how the RTLS-QR algorithm of section 4.2 is modified to provide multi-user separation based on the coding scheme of algorithm C2.

- in each symbol period, compute the SVD of:

$$Y_{k|k+i-1} = \begin{bmatrix} U_k^s & U_k^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma_k^s & 0 \\ 0 & \Sigma_k^\perp \end{bmatrix} \cdot \begin{bmatrix} V_k^{sH} \\ V_k^{\perp H} \end{bmatrix}$$

  where $Y_{k|k+i-1}$ was defined in equation 2.13.

- construct $\overline{W}_k^l$ as:

$$\begin{aligned} Z_k^l &\triangleq \begin{bmatrix} V_{k-i+1}^{\perp T} \\ \vdots \\ V_{k+L_{min}}^{\perp T} \end{bmatrix} \cdot D_l(k : k+j-1, k : k+j-1) \\ \overline{W}_k^l &\triangleq \begin{bmatrix} \Re(Z_k^l) \\ \Im(Z_k^l) \end{bmatrix} \end{aligned}$$

- construct d parallel versions of the RTLS-QR algorithm (algorithm 4.2.2) and for user l replace W_k by $\overline{W}_k^l$.

The resulting adaptive algorithm removes both MAI and ISI. The matrix $\overline{W}_k^l$ defines a different cost function for each user at each time instant. Each of the matrices $\overline{W}_k^l$ has a one-dimensional null space. A similar combination of the RTLS-QR algorithm with the coding scheme of algorithm C3 is easily derived.

Furthermore instead of combining coding with the RTLS-QR algorithm one could also employ the Kalman or Viterbi algorithm. When combined with a Viterbi algorithm we construct d Viterbi trellises each using a different cost function $\overline{W}_k^l$. Hence, the coding schemes of algorithms C2 and C3 form an alternative for the non-linear block coding presented in the previous chapter. This procedure has the additional advantage not requiring any coding overhead. Instead the coding scheme of algorithm C2 restricts the modulation format, while the coding scheme of algorithm C3 imposes more stringent constraints on the number of antennas.

7.7 Conclusions

In this chapter we have presented three new deterministic non-iterative blind signal separation algorithms. The three algorithms entail different trade-offs: transmission efficiency (coding overhead) versus modulation format or hardware (number of antennas) and illustrate that multi-user coding is a flexible concept.

Algorithm C1 is based on block coding, requires coding overhead but does not impose any other restrictions. Algorithm C2 is only applicable when the modulation format is one-dimensional, but requires no coding overhead. Algorithm C3 is based on transmit diversity, is applicable for any modulation format but imposes stronger conditions on the number of transmitter and receiver antennas. For the three algorithms, coding allows to split a d-dimensional problem (for d users) into d one-dimensional problems which may be solved in parallel. We have derived theoretical deterministic requirements on coding matrices to provide proper signal separation. A theoretical framework for the derivation of statistically optimal coding matrices has not yet been developed and remains an open problem.

In section 7.5 we have shown how these algorithms can be extended to remove ISI. We have indicated how coding provides robustness against the problem of unknown channel lengths and order detection (a property usually only shared by stochastic algorithms, see section 2.5). These favorable properties will be corroborated by the simulation results of the next chapter. In section 7.6 we have shown how the single user RTLS-QR algorithm of chapter 4 can be combined with the coding scheme of algorithm C2 to provide a fully adaptive multi-user algorithm. Other adaptive multi-user algorithms can be derived in a similar way.

Chapter 8

Simulation results

This chapter presents simulation examples analyzing the performance of the multi-user algorithms developed in chapter 6 and chapter 7. In a blind source separation context we show that the proposed coding based algorithms have a performance comparable or slightly worse than that of existing source separation algorithms [115, 128]. The assets of the new algorithms are their simplicity, their lack of convergence problems and their implicit parallelism. When MAI is present, this chapter illustrates some interesting robustness properties of the proposed algorithms. First, simulation results show that the transmitted symbol sequences can be accurately detected even if the system order is largely overestimated, which corroborates the theoretical derivations of previous chapters. Secondly, we demonstrate that the proposed algorithms allow symbol estimation without the exact identification of the channel lengths of individual users (which usually involves a cumbersome iterative procedure [81]).

Section 8.1 summarizes the ILSP algorithm [115] which we will use for comparison further on. Section 8.2 presents simulation results and finally in section 8.3 some conclusions are drawn.

8.1 ILSP algorithm [115]

This section reviews the iterative least squares with projection (ILSP) algorithm [115] that will be used in the simulation examples of section 8.2. We assume the data model of equation 7.3:

$$\underbrace{\begin{bmatrix} \mathbf{y_1} & \cdots & \mathbf{y_N} \end{bmatrix}}_{Y} = A \cdot \underbrace{\begin{bmatrix} \mathbf{x_1} & \cdots & \mathbf{x_N} \end{bmatrix}}_{X} + \begin{bmatrix} \mathbf{n_1} & \cdots & \mathbf{n_N} \end{bmatrix}$$

Algorithm 8.1
ILSP algorithm

1. *initialization:* A_0, $k = 0$

2. *until convergence:*
   - $k = k + 1$
   - $\hat{X}_k = (A_{k-1}^H \cdot A_{k-1})^{-1} \cdot A_{k-1}^H \cdot Y$
   - $\hat{\hat{X}}_k = \mathcal{D}(\hat{X}_k)$
   - $A_k = Y\hat{\hat{X}}_k^H(\hat{\hat{X}}_k \cdot \hat{\hat{X}}_k^H)^{-1}$

with the output matrix $Y \in \mathbb{C}^{M \times N}$, the mixing matrix $A \in \mathbb{C}^{M \times d}$ and $X \in \Omega^{d \times N}$, with Ω the finite alphabet to which the symbols belong. The ILSP algorithm estimates A and X by minimizing the cost function $\|Y - A \cdot X\|_{\mathrm{F}}^2$. The algorithm assumes that an initial estimate of A is available (denoted by $\hat{A}$) and consists of the following steps:

- Estimate $\hat{X}$ by solving a least squares problem in Y and $\hat{A}$.
- Map each element of $\hat{X}$ onto the closest finite alphabet element $\hat{\hat{X}} = \mathcal{D}(\hat{X})$.
- Obtain a better estimate of A by solving a least squares problem involving the variables Y and $\hat{\hat{X}}$.

This process is repeated until convergence, as described in algorithm 8.1. The ILSP algorithm is computationally simple (it involves two least squares problems in each iteration step). However, the algorithm is not proved to converge and may be trapped in local minima requiring reinitialization of the algorithm (see also the simulation results of figures 8.1 and 8.2). As the constellation size increases, it becomes important to have reliable initialization procedures to avoid too many restarts.

Note that the matrix $\hat{\hat{X}}$ is estimated in two steps: the first step computes a least squares solution $\hat{X}$ and the second step is a projection onto the closest constellation symbols. A more optimal approach is to directly optimize the estimated X in the finite alphabet (without projection step). When $\hat{X}$ is enumerated over all possible symbol sequences, this leads to the ILSE algorithm [115], however we will not consider ILSE here.

	modulation	ISI	algorithms
simulation 1	BPSK	no	C1,C2,ILSP,Vit.,RACMA
simulation 2	QPSK	no	C1,C3,ILSP
simulation 3	BPSK	no	C2,C3
simulation 4	BPSK	yes	C1,C2,Vit.
simulation 5	BPSK	yes	Vit.
simulation 6	BPSK	yes	C1
simulation 7	BPSK	yes	C2
simulation 8	QPSK	yes	C3
simulation 9	BPSK	no	C2, optimal receiver

Table 8.1: The different simulation scenarios studied in this section.

8.2 Simulation results

In this section we analyze the performance of the multi-user algorithms developed in chapters 6 and 7. We compare the algorithms both in situations without ISI and with ISI. Table 8.1 summarizes all simulation scenarios. In each simulation bursts consist of $N = 101$ DBPSK (or DQPSK) symbols. Hence, we have 100 information symbols per source. Results are averaged over 2000 trials and in each trial a different realization for the data and noise sequences is generated.

In a *first simulation* we compare the performance of the algorithms C1, C2, Viterbi, ILSP [115] and RACMA[1] [128] for an instantaneous linear mixture and BPSK modulation. For algorithm C2 we use random complex constant modulus codes (Π is a complex constant modulus alphabet). For all other algorithms we split the received signal in its real and imaginary part as in equation 7.12. For the ILSP algorithm we show results without reinitialization and with 2 reinitializations (for initialization we choose $\hat{A} = \left[\begin{smallmatrix} I \\ 0 \end{smallmatrix}\right]$, in the reinitializations we choose a random $\hat{A}$ matrix). For algorithm C1, we use coding scheme Z3 with $b = 1, 2$, see example 7.2.3. For the Viterbi algorithm $j = 14$, $n = 5$, $p = 2$ and coding scheme Z2 is applied, see section 6.1.

We assume $d = 4$ equal power users, and $M = 6$ receiver antennas. The matrix A was generated as a random complex matrix, in which the power of each column is normalized. The SNR is defined as:

$$SNR = 10 \cdot \log_{10} \frac{E\{\|A \cdot \mathbf{x_k}\|^2\}}{d \cdot E\{\|\mathbf{n_k}\|^2\}}.$$

Figure 8.1 depicts the average BER per user as a function of SNR for all

[1] For a description of the RACMA algorithm we refer to [128].

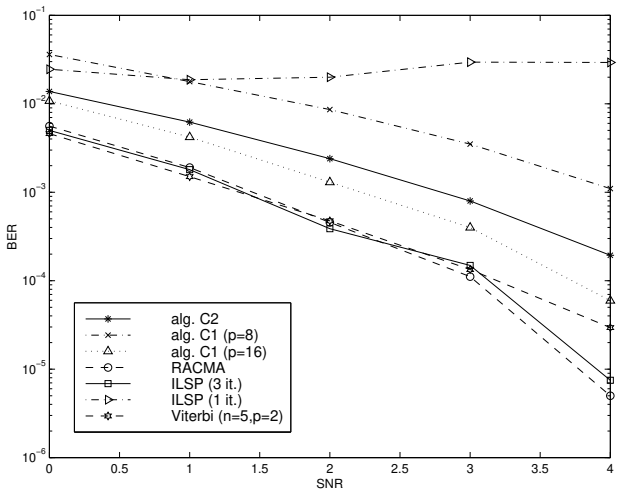


Figure 8.1: Average BER per user for varying SNR, BPSK modulation, $d = 4$, $M = 6$.

algorithms. The performance of the ILSP algorithm depends heavily on (re)initialization conditions. When only one initialization is used, the performance of the ILSP algorithm levels out for increasing SNR. The performance of algorithm C1 is flexible and improves when the amount of coding overhead is increased. Algorithm C2 has a performance in between that of algorithm C1 (p=8) and C1 (p=16). The Viterbi algorithm outperforms the other algorithms developed in previous chapters due to the finite alphabet constraint and the large amount of coding that is used. However, it does not exploit the fact that the mixing matrix is constant over the whole burst. The best performance is given by the ILSP algorithm with 3 initializations and the RACMA algorithm which both exploit the time invariance of the channel and the finite alphabet property of the signal. In conclusion, the algorithms developed in previous chapters have simple decoding schemes and do not suffer from convergence problems. However, algorithms C1 and C2 do not impose a finite alphabet constraint while the Viterbi algorithm does not exploit the time invariance of the channel. This leads to some performance loss with respect to existing algorithms.

In a *second simulation* the modulation format is QPSK. We now compare the performance of algorithms C1, C3 and ILSP. Algorithm C2 and the RACMA algorithm cannot be applied for QPSK modulation. For the subspace + Viterbi algorithm simulation time becomes excessive because the number of states in the Viterbi trellis equals 4^{j-1}. We now assume $d = 3$ users and $M_r = 8$ antennas at the receiver side. For algorithm C3 we assume $M_t = 2$ transmit antennas per user. The magnitude of each coding element is $1/\sqrt{2}$ while its phase is ran-

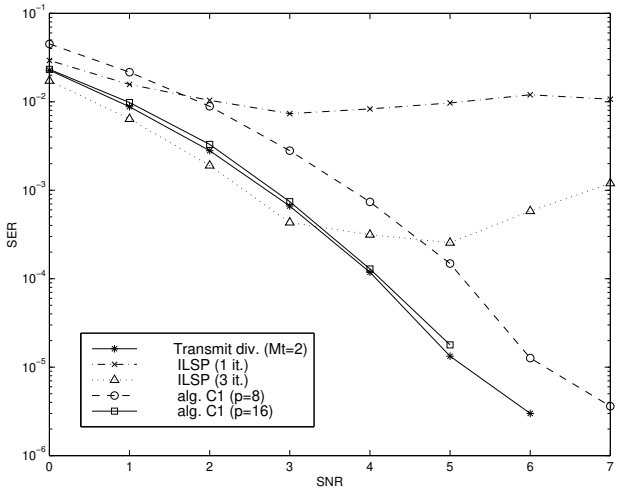


Figure 8.2: Average SER per user for varying SNR, QPSK modulation, $d = 3$, $M = 8$.

dom, as in equation 7.28. For algorithm C3 the A matrix is thus a 8×6 matrix A_1 while for the other algorithms it is a 8×3 matrix A_2. A_1 and A_2 were constructed such that each column of both matrices has unit norm and the condition number of both matrices is equal. Figure 8.2 shows the symbol error rate (SER) as a function of SNR. For the ILSP algorithm even three initializations are not sufficient to escape from local minima. For larger constellation sizes, the ILSP algorithm requires reliable initialization procedures to avoid too many reinitizalizations. For the other algorithms performance improves for increasing SNR. The performance of algorithm C1 improves when more coding is applied. Algorithm C3 (not requiring any coding overhead) performs equally well as algorithm C1 requiring 16 % overhead. As the constellation size further increases the cost function of the ILSP algorithm exhibits too many local minima to ensure good convergence while algorithms C1 and C3 do not suffer from this problem.

In a *third simulation* we compare the performance of algorithm C2 (only applicable for BPSK modulation) and algorithm C3 (exploiting transmit diversity). To compare their performance we take the same scenario as the previous simulation ($d = 3$, $M_r = 8$ and same A_1 and A_2 matrices) but the modulation format is BPSK. For algorithm C3 we use real coding matrices (see remark 7.4.3) and split the received signal into its real and imaginary part before processing (equation 7.12). As is seen from figure 8.3 both algorithms have almost identical performance. In algorithm C2 the first step detects a 3-dimensional complex signal subspace in a 8-dimensional complex space, while for algorithm

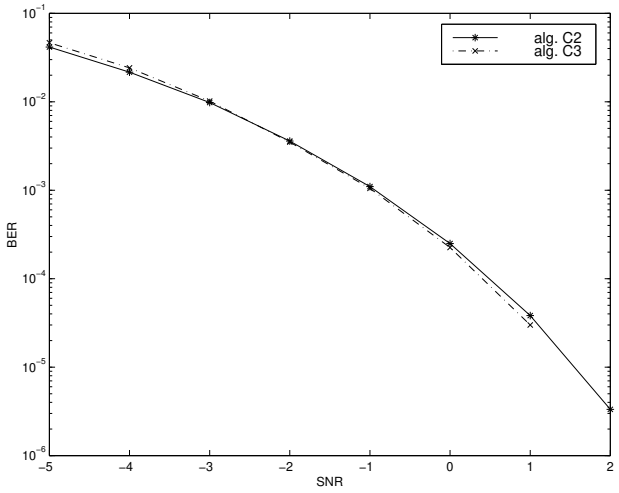


Figure 8.3: Average BER per user for varying SNR, BPSK modulation, $d = 3$, $M = 8$.

C3 the first step estimates a 6-dimensional real subspace in a 16-dimensional real space. The second step in both algorithms exploits coding information. The simulation results shows that the constellation diversity exploited in algorithm C2 and the spatial diversity exploited in algorithm C3 are quite similar.

Next we investigate *the influence of ISI on performance*. We assume $d = 2$ users with channel lengths $L_1 = 4$ and $L_2 = 2$ and $M = 6$ antennas. The channel matrices are complex random, but power is normalized such that $\|H_1\|_{\mathrm{F}}^2 = \|H_2\|_{\mathrm{F}}^2$. First we compare the performance of the Viterbi algorithm, algorithm C1 and algorithm C2 in the case where a lower bound on the channel length $L_{min} = 2$ is known. For the Viterbi algorithm $j = 14$, $i = 2$ and the coding scheme of example 6.2.2 is used ($n = 3$, $p = 3$). For algorithm C1 we assume the coding scheme presented in example 7.5.2 and $i = 4$. The coding scheme for algorithm C2 remains constant modulus, random phase with smoothing factor $i = 4$. For the Viterbi algorithm we use $W_k^{i+L_{min}}$ as defined in equation 6.11. For algorithms C1 and C2 we use $G^{i+L_{min}}$ as defined in equation 7.36. Figure 8.4 depicts the results and reveals important differences compared to the blind source separation simulation results (figure 8.1). Although the Viterbi algorithm now requires 50 % coding overhead, performance is worse than that of algorithms C1 and C2 especially for higher SNR. Also note that algorithm C2 performs almost equally well as algorithm C1 while not requiring any coding overhead.

Next, we want to quantify *the influence of the number of shifts in the construc-*

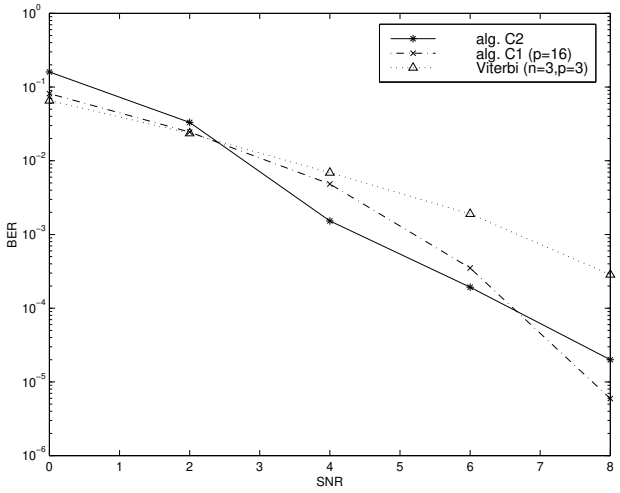


Figure 8.4: Average BER per user for varying SNR, BPSK modulation, $d = 2$, $M = 6$, $L_1 = 4$, $L_2 = 2$ and $i + L_{min}$ shifts.

tion of W_k^l and G^l and the robustness against order detection problems. We define the following simulation scenarios.

1. The channel length of each user is known and G^{i+L_l} is applied for user l (or for the Viterbi algorithm $W_k^{i+L_l}$).
2. Only $L_{min} = 2$ is known, $G^{i+L_{min}}$ is used for each user (or for the Viterbi algorithm $W_k^{i+L_{min}}$).
3. Nothing is known about the channel lengths of the individual users $L_{min} = 0$, we use G^i for each user.
4. Identical to scenario 2, but additionally the dimension of $V^\perp$ is underestimated.
5. Identical to scenario 3, but additionally the dimension of $V^\perp$ is underestimated.

We compare these scenarios for each of the algorithms (Viterbi, C1, C2 and C3) separately.

Figure 8.5 shows results for *the Viterbi algorithm* ($n = 3$, $p = 3$). The bit error rate of scenario 2 is lower than that of scenario 1. Although only a lower bound on the channel length is known in scenario 2, performance is better than when all channel lengths are known exactly (scenario 1). In scenario 2 we rely less on

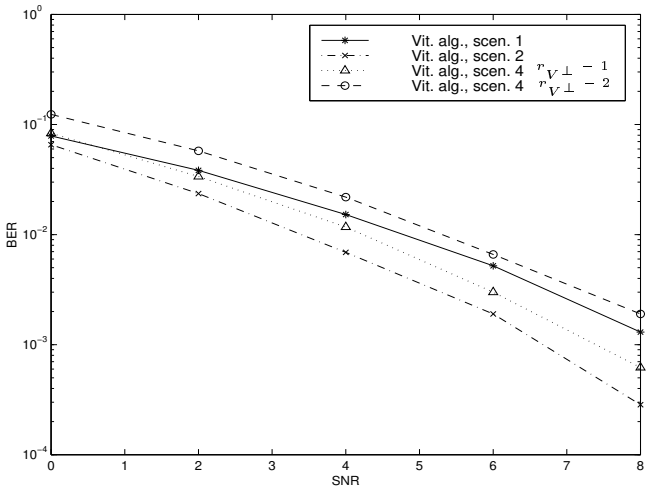


Figure 8.5: Average BER per user for varying SNR for the Viterbi algorithm with ISI, BPSK modulation, $d = 2$, $M = 6$, $L_1 = 4$, $L_2 = 2$ and $i = 2$.

subspace information, but more on the coding and the finite alphabet property to recover the symbol sequences. Results for scenario 3 are not shown because the Viterbi algorithm requires too much coding when only i shifts are applied. The Viterbi algorithm shows to be quite sensitive to an underestimation of the null space dimension (scenario 4), even when it is only underestimated by 1 or 2. This is primarily explained by the small window size ($j = 14$). The true value of the null space dimension $r_{V^\perp} = j - d \cdot i - L_1 - L_2 = 14 - 4 - 4 - 2 = 4$. $W_k^{i+L_{min}}$ is then a $j \times r_{V^\perp} \cdot (i + L_{min}) = 14 \times 16$ matrix. When $\hat{r}_{V^\perp}$ is respectively 3 and 2, $W_k^{i+L_{min}}$ becomes a 14×12 and a 14×8 matrix. The set of equations becomes underdetermined. Although there is a performance loss, observe that the Viterbi algorithm still provides meaningful results. In the simulation results presented below, it is illustrated that other (block processing) algorithms are less sensitive to an incorrectly estimated $r_{V^\perp}$. We conclude that the Viterbi algorithm provides good performance in a source separation scenario (figure 8.1) but in the presence of ISI (where different users may have different channel lengths) both the complexity of the algorithm (the limited value of j) and the large amount of coding (50 % in the example above) render the algorithm impractical.

Remark 8.2.1 The multi-user + Viterbi algorithm can also be applied in situations where the channel is time varying. For simulation results we refer to [148].

Figure 8.6 shows simulation results for *algorithm C1* ($p = 16$). The performance

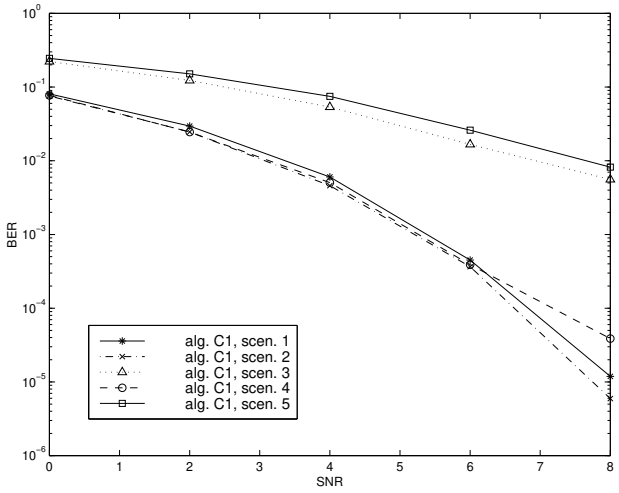


Figure 8.6: Average BER per user for varying SNR for algorithm C1 with ISI, BPSK modulation, $d = 2$, $M = 6$, $L_1 = 4$, $L_2 = 2$ and $i = 4$.

improvement for scenario 2 with respect to scenario 1 is modest but again shows that applying the 'correct' number of shifts for each user (i.e. identifying the correct channel length for each user) does not necessarily lead to the best performance. Further decreasing the number of shifts to i leads to a large performance loss. When only i shifts are applied G^i defines a $d + \sum_{l=1} L_l = 8$-dimensional null space. The coding has to remove MAI from 6 additional interferers, which leads to a performance loss. In scenario 4 and 5, we assume that the signal subspace dimension is $M \cdot i$, i.e. $V^{\perp}$ is the null space of the (full rank noise corrupted) $Mi \times j$ matrix $Y_{L_2+1|L_2+i}$ The signal subspace dimension is overestimated by $M \cdot i - (d \cdot i + \sum L_l) = 24 - 14 = 10$. The dimension of the null space $r_{V^{\perp}}$ is thus underestimated by 10. Figure 8.6 shows that scenario 4 and 5 exhibit a performance loss with respect to scenario 2 and 3. However, despite the large underestimation of the null space dimension, algorithm C1 shows good robustness against system order detection problems.

Remark 8.2.2 Other simulations (results not displayed) have shown that when the dimension of the null space is not that largely underestimated (i.e. $r_{V^{\perp}} - \hat{r}_{V^{\perp}} < 10$) performance losses are smaller and sometimes even slight performance improvements are obtained.

Figure 8.7 shows results for *algorithm C2*. As for the Viterbi algorithm and algorithm C1, scenario 1 has a worse performance than scenario 2. Additionally, there is only a moderate performance difference between scenarios 2 and 3, which indicates that algorithm C2 is fairly robust against channel order es-

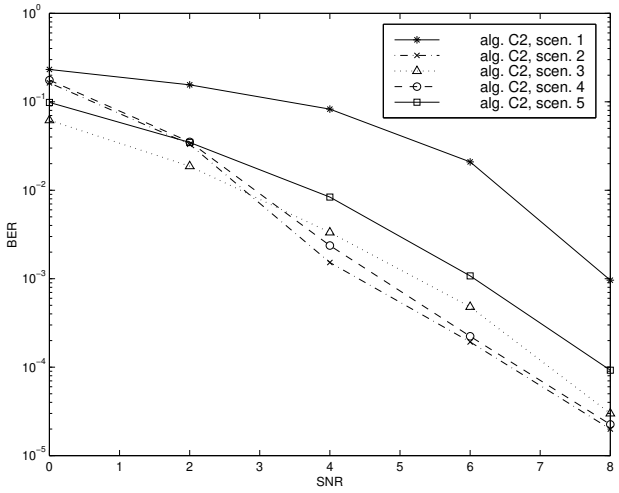


Figure 8.7: Average BER per user for varying SNR for algorithm C2 with ISI, BPSK modulation, $d = 2$, $M = 6$, $L_1 = 4$, $L_2 = 2$ and $i = 4$.

timation problems. Good performance can be obtained even in the absence of any information on the channel lengths of the individual users. Further, the figure also illustrates the robustness of the algorithm against an underestimated $r_{V^\perp}$. Again the dimension of the null space is underestimated by 10. When $(i + L_{min})$ shifts are applied the effect of underestimating the dimension of the null space is minimal. When only i shifts are applied the effect is more pronounced but again remark 8.2.2 applies.

Finally figure 8.8 shows results for *algorithm C3*. Simulation parameters are different. We assume $M_r = 8$ antennas at the receiver side and $M_t = 2$ antennas at the transmitter side. The smoothing factor $i = 6$ and the channel lengths remain $L_1 = 4$ and $L_2 = 2$. The modulation format is QPSK. Again scenario 2 outperforms scenario 1 and scenario 3, but algorithm C3 is shown to be robust against the problem of unknown channel lengths. For scenario 4 and 5 the dimension of the null space is again underestimated by 10. Simulation results show that algorithm C3 also provides good robustness against an underestimation of the null space dimension.

Remark 8.2.3 The simulation results presented above show that all algorithms provide robustness and to some extent a performance improvement when less than $i + L_l$ shifts are applied for user l. A similar positive effect of taking fewer intersections was reported in [133]. However, since coding was not applied, fewer intersections led to several phantom solutions, hence to increased problem dimensions and (sometimes) to convergence problems for the

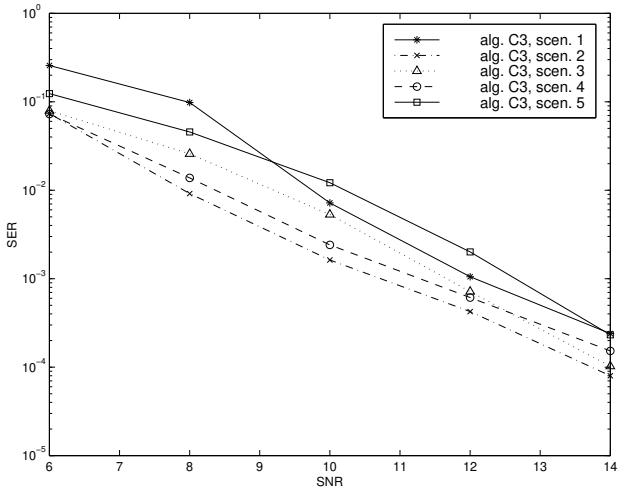


Figure 8.8: Average SER per user for varying SNR for algorithm C3 with ISI, QPSK modulation, $d = 2$, $M_t = 2$, $M_r = 8$, $L_1 = 4$, $L_2 = 2$ and $i = 6$.

ILSP algorithm.

The last simulation setup concerns algorithm C2 only. The performance of algorithm C2 is compared to that of several optimal receivers. More specifically *the performance gain of the blind multi-user algorithm C2 with respect to standard single user receivers is investigated.* This section provides an answer to the following questions. What is the optimal performance that can be obtained in a multi-user scenario? How does the performance of algorithm C2 relate to this optimal performance ? How does the performance of algorithm C2 relate to that of an optimal single-user receiver?

We compare the performance of the following transmitter receiver combinations ($k = 4$ and $k = 6$):

1. one user transmitting 2^k-QAM bursts to 1 antenna with power $k \cdot P$ at a data rate $k \cdot R$ (optimal demodulation).
2. one user transmitting 2^k-QAM bursts to $M = 6$ antennas with power $k \cdot P$ at a data rate $k \cdot R$ (optimal demodulation).
3. k users transmitting BPSK bursts to $M = 6$ antennas with power P at a data rate R (optimal demodulation).
4. k users transmitting BPSK bursts to $M = 6$ antennas with power P at a data rate R (algorithm C2).

These four scenarios have in common that:

- the same amount of information is conveyed from transmitter to receiver at each time instant, i.e. all scenarios share the same aggregate data rate.
- the energy per transmitted bit is the same in the four scenarios.

Therefore it is relevant to compare these scenarios. If $k = 4$, 16-QAM symbols are transmitted in the first two scenarios. If $k = 6$, 64-QAM bursts are transmitted. We analyze the BER performance of the four scenarios as a function of SNR per bit, which we denote as γ_b. Except for scenario 4, all scenarios represent ideal situations, where analytic expressions for the BER exist. First we review these analytic expressions, next we compare these expressions with the performance of algorithm C2 which is obtained through simulation. During the derivations of the BER expressions we assume spatially and temporally white noise at each of the antennas. The noise is assumed to be complex-valued zero mean white Gaussian with independent real and imaginary parts, each having variance $\sigma^2/2$.

Scenario 1 (one transmitter sending bursts to one receiver) is standard:

$$y(k) = a \cdot x(k) + n(k).$$

For simplicity we assume $|a| = 1$, such that the received power per bit equals P. BER formulas are readily available in literature (e.g. [104, p 282]). Define the alphabet size $\eta = 2^k$. The BER probability $P_{2^k\text{-QAM}}$ varies as function of SNR per bit $\gamma_b = \frac{P}{\sigma^2}$:

$$P_{2^k\text{-QAM}} = 2(1 - \frac{1}{\sqrt{\eta}}) \text{ erfc } (\sqrt{\frac{3}{2(\eta-1)} k\gamma_b}) \times (1 - \frac{1}{2}(1 - \frac{1}{\sqrt{\eta}}) \text{ erfc } (\sqrt{\frac{3}{2(\eta-1)} k\gamma_b}))/k.$$

The division by k allows to obtain a formula for the bit error rate from an expression for the symbol error rate. Gray encoding (remark 2.2.1) is assumed such that each symbol error will most likely only cause one bit error.

In scenario 2, one transmitter communicates with M receivers:

$$\mathbf{y_k} = \mathbf{a} \cdot x(k) + \mathbf{n_k}.$$

We assume $\|\mathbf{a}\| = 1$. The received signal power equals again $k \cdot P$, but the noise power is now $M \cdot \sigma^2$, such that $\gamma_b = \frac{P}{M \cdot \sigma^2}$. The optimal receiver filters the incoming data by a spatially matched filter, i.e.:

$$\mathbf{a}^H \cdot \mathbf{y_k} = \mathbf{a}^H \cdot \mathbf{a} \cdot x(k) + \mathbf{a}^H \cdot \mathbf{n_k}$$

$$= \quad x(k) + \mathbf{a}^H \cdot \mathbf{n_k}.$$

Matched filtering leaves the received signal power unaltered. The noise power before matched filtering equaled $M \cdot \sigma^2$, after matched filtering it equals:

$$E\{\mathbf{a}^H \cdot \mathbf{n_k} \cdot \mathbf{n_k}^H \cdot \mathbf{a}\} = \sigma^2.$$

Matched filtering has increased the signal to noise ratio by a factor M. After matched filtering we again obtain scenario 1: one transmitter, one receiver but a modified SNR. For a receiver with M antennas we can use the classical single user BER-formulas with an M times higher SNR:

$$P_{2^k\text{-QAM}} = 2(1 - \frac{1}{\sqrt{\eta}}) \text{ erfc } (\sqrt{\frac{3}{2(\eta - 1)} kM\gamma_b})$$
$$\times (1 - \frac{1}{2}(1 - \frac{1}{\sqrt{\eta}}) \text{ erfc } (\sqrt{\frac{3}{2(\eta - 1)} kM\gamma_b}))/k.$$

In scenario 3, $d = k$ users transmit BPSK symbols to $M = 6$ antennas:

$$\begin{aligned} \mathbf{y_k} &= \sum_{l=1}^{d} \mathbf{a_l} \cdot x^{(l)}(k) + \mathbf{n_k} \\ &= A \cdot \mathbf{x_k} + \mathbf{n_k}. \end{aligned}$$

We assume that the mixing matrix A is orthonormal. After matched filtering scenario 3 essentially reduces to scenario 2.:

$$\mathbf{a_l}^H \cdot \mathbf{y_k} = x^{(l)}(k) + \mathbf{a_l}^H \cdot \mathbf{n_k} \qquad l = 1, \ldots, d.$$

The optimal receiver for scenario 3 removes all MAI perfectly and increases the SNR by a factor M. In a perfect scenario adding extra co-channel users induces no penalty: capacity is increased without any performance loss. The BER performance can be expressed as

$$P_{\text{BPSK}} = \frac{1}{2}\text{erfc } (\sqrt{M \cdot \gamma_b}).$$

However, in practice (scenario 4) increasing the number of users will also increase BER because of non-ideal MAI removal. How big is the performance loss of scenario 4 with respect to scenario 3? The second question is whether there is a performance gain in scenario 4 with respect to scenario 2. If not, it might by more useful to equip one user with full power instead of allowing four low powered co-channel users. We use $M = 6$ receiver antennas and for the simulation results of algorithm C2 we averaged over 20000 runs. Figure 8.9 depicts the results. First note that in both plots, scenario 1 (only one receiver

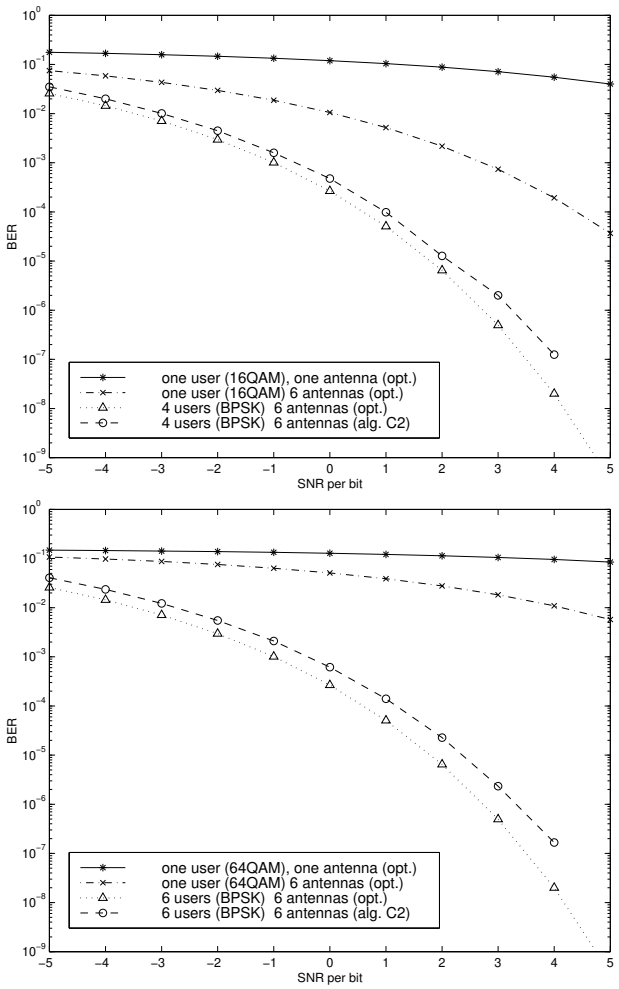


Figure 8.9: Performance comparison for the four scenarios $k = 4$ (top) and $k = 6$ (bottom).

antenna) gives the worst performance, hence illustrating the benign effect of an antenna array at the receiver side. Spreading the bit rate amongst different users has clearly a positive effect on performance. Algorithm C2 (i.e. scenario 4), although suboptimal with respect to scenario 3, still performs a lot better than the optimal single user 16-QAM or 64-QAM receiver. The lower bound on the optimal performance for the multi-user detection is the same in the left and right figure: ideally all MAI is removed without any penalty. Note however that also for algorithm C2 there is relatively little performance loss when going

from 4 to 6 users. This contrasts with the single user situation where there is a clear performance gap between the optimal 16-QAM and 64-QAM receiver.

8.3 Conclusions

In this chapter we have compared the performance of the blind multi-user receiver algorithms developed in chapters 6 and 7. The performance of the new algorithms is comparable or somewhat inferior to that of existing signal separation algorithms [115, 128]. The Viterbi algorithm has a good performance but does not exploit the fact that the channel is time-invariant. The algorithms C2 and C3 avoid the coding overhead and might be more attractive from a computational point of view but do not exploit the finite alphabet property of the signals. In a blind source separation context, the power of the algorithms lies in their simplicity: the algorithms are non-iterative, have no convergence problems and create one-dimensional problem formulations.

With ISI, we have demonstrated two important properties of the coding. First we investigated performance when not all channel lengths of individual users are known. Simulation results illustrated that all algorithms show good performance when the smallest amongst all channel lengths is available, i.e. not all channel lengths of the individual users need to be identified. Furthermore algorithms C2 and C3 show good robustness even when the smallest channel length is not known. The amount of coding needed by the Viterbi algorithm to provide good ISI protection might be prohibitive. Secondly, we have demonstrated robustness against underestimation of the dimension of the null space (overestimation of the system order). Simulation results have shown that a block processing approach may be more appropriate than an adaptive approach. For all algorithms C1, C2 and C3 it was shown that they are robust even when the dimension of the null space is largely underestimated, while the Viterbi algorithm only provided limited protection against an underestimation of the system order.

Finally, we have demonstrated the benefit of multi-user equalization with respect to (optimal) single user transmission schemes.

Part III

Subspace algorithms for blind channel estimation in noise fields with unknown spatial color

In previous chapters we have studied single and multi-user blind symbol estimation algorithms. In chapter 9 and chapter 10 we again study the single user problem. However, the approach taken in these chapters differs from the algorithms presented in chapters 3, 4, 6 and 7. We aim at estimating the channel instead of directly estimating the symbols, i.e. we present *blind channel estimation algorithms*. After the channel has been identified, the symbols can be recovered via e.g. a Viterbi algorithm. Estimating the channel can be interesting when the channel is stationary for long data bursts since the number of parameters to be estimated does not increase with the burst length and hence parameters can be estimated consistently.

Several stochastic and deterministic blind channel estimation algorithms have been proposed in literature. Stochastic algorithms have been developed based a.o. on frequency domain methods exploiting cyclostationary statistics [118, 160], linear prediction [111, 4, 41] and optimal moment matching [163, 162, 46] (see also section 2.5 and [82, 117]). The main disadvantage of stochastic algorithms is that they may suffer from performance degradations when only a limited number of output data is available. On the other hand, deterministic algorithms are very data efficient. In [97] a subspace based deterministic blind channel estimation algorithm is proposed that exploits the block Toeplitz structure of the channel matrix. Optimally weighted subspace fitting algorithms have been derived in [1, 68]. In [121] a blind channel estimation algorithm is proposed based on orthogonal projections. However, all these algorithms assume that the additive noise is spatially white or else that its spatial covariance is known. In the latter case, the known covariance is used to whiten the signal prior to channel estimation. In [2], a deterministic subspace algorithm was proposed robust against an unknown spatial color of the additive noise[2].

In chapter 9 and 10 we present three blind channel estimation algorithms that are also robust against an unknown spatial color of the noise. The new idea introduced here is to precede channel recovery by a noise filtering step based on orthogonal or oblique projections.

In chapter 9 we derive a stochastic algorithm for blind channel recovery[3]. The first step in the algorithm reformulates the blind channel estimation problem in a stochastic state space framework and is based on orthogonal projections. In a second step the channel parameters are estimated using a subspace approach as in the algorithms [97, 2]. In chapter 10 we present two deterministic algorithms for blind channel recovery. The first step of these algorithms is based on oblique projections. The second step of the first deterministic algorithm hinges on a rank one matrix approximation, while the second algorithm uses a subspace approach for channel recovery.

[2] The algorithms [97],[121] and [2] are summarized in sections 11.1, 11.2 and 11.3 respectively.

[3] This stochastic algorithm forms the starting point for the derivation of the second deterministic algorithm of chapter 10.

In chapter 11 we analyze the performance of the new algorithms and compare them with several other algorithms [97, 121, 2] that have recently appeared in literature.

Chapter 9

A stochastic subspace algorithm based on orthogonal projections

In this chapter we derive a stochastic algorithm for blind channel estimation. The algorithm exploits the signal and noise statistics. *The algorithm does not give perfect channel estimates for a finite number of data. On the other hand, it provides perfect channel estimates when the number of data tends to infinity and this independent of the noise level or color.*

The algorithm is based on a state space description of the data model and relies on orthogonal projections. It is closely related to the theory of [135], where non-blind subspace identification algorithms are presented.

The algorithm has the same identifiability conditions as the algorithm [2] (which is also robust against the spatial noise color and which is summarized in section 11.3). However, simulation results (see chapter 11) show that the new algorithm has a better performance than the algorithm of [2]. *This chapter contains two original contributions. First, it reformulates the blind channel estimation problem in a stochastic state space framework such that results of [135] can be applied to the blind identification problem at hand. Secondly, we show how two orthogonal projections are combined to recover the channel parameters.*

The outline of this chapter is as follows. Section 9.1 presents a stochastic state space data model. Section 9.2 introduces the model assumptions. Section 9.3 presents the algorithm that consists of two steps. The first step is a noise filtering preprocessing step based on two orthogonal subspace projections, derived from the theory of [135]. Using these orthogonal projections, the channel is es-

timated in the second step of the algorithm. Finally section 9.4 presents some conclusions. Simulation results analyzing the performance of the algorithm will be provided in chapter 11.

9.1 Data model

We start from the single user data model of equation 2.6

$$\mathbf{y_k} = \underbrace{\begin{bmatrix} \mathbf{h_L} & \dots & \mathbf{h_0} \end{bmatrix}}_{H} \cdot \begin{bmatrix} x(k-L) \\ \vdots \\ x(k) \end{bmatrix} + \mathbf{n_k}. \tag{9.1}$$

Further on, we make the following assumptions:

Assumption 9.1.1 *The noise* $\mathbf{n_k}$ *is temporally white but can be spatially colored, i.e.* $E\{\mathbf{n_k} \cdot \mathbf{n_l}^H\} = R_n \cdot \delta_{kl}$ *with* R_n *the spatial noise covariance matrix.*

Assumption 9.1.2 *Source symbols are uncorrelated, unit variance, i.e.* $E\{x(k) \cdot x(l)\} = \delta_{kl}$ *and noise and symbols are uncorrelated, i.e.* $E\{x(k) \cdot \mathbf{n_l}\} = 0, \forall k, l.$

Remark 9.1.3 The extension of the algorithms of chapters 9 and 10 to temporally colored noise, which is a relevant problem (see also remark 2.3.4), remains an interesting open problem.

First we rewrite the data model of equation 9.1 into a forward and backward stochastic state space model. These state space models will then be used in the next section when analyzing the orthogonal projections. Next we succinctly review how to modify equation 9.1 to obtain a data model with Hankel structure (see also section 2.3).

9.1.1 Forward state space model

Define the state vector $\mathbf{s_k}$ as:

$$\mathbf{s_k} = \begin{bmatrix} x(k-L) \\ x(k-L+1) \\ \vdots \\ x(k-1) \end{bmatrix}$$

then we obtain the following state and output equation

$$\begin{aligned}
\mathbf{s_{k+1}} &= \underbrace{\begin{bmatrix} 0 & 1 & 0 & \cdots & \\ 0 & 0 & 1 & \cdots & \\ & \ddots & & \ddots & \\ 0 & \cdots & & & 1 \\ 0 & \cdots & & & 0 \end{bmatrix}}_{A} \cdot \mathbf{s_k} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}}_{\mathbf{q_k}} \cdot x(k) \\
\mathbf{y_k} &= \underbrace{\begin{bmatrix} \mathbf{h_L} & \cdots & \mathbf{h_1} \end{bmatrix}}_{C} \cdot \mathbf{s_k} + \underbrace{(\mathbf{h_0} \cdot x(k) + \mathbf{n_k})}_{\mathbf{r_k}}
\end{aligned}$$

where the (unknown) input signal is treated as a random white noise sequence. The noise covariance matrices are defined in the following way (using assumptions 9.1.1 and 9.1.2):

$$E\{\begin{bmatrix} \mathbf{q_k} \\ \mathbf{r_k} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{q_l}^H & \mathbf{r_l}^H \end{bmatrix}\} = \begin{bmatrix} Q & S \\ S^H & R \end{bmatrix} \cdot \delta_{kl}.$$

The following identities are easily derived from the above data model:

$$\begin{aligned}
Q &= \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}, \\
S &= \begin{bmatrix} 0_{(L-1)\times M} \\ \mathbf{h_0}^H \end{bmatrix}, \\
R &= \mathbf{h_0} \cdot \mathbf{h_0}^H + R_n
\end{aligned}$$

where R_n represents the spatial covariance matrix of the additive noise $\mathbf{n_k}$.

The conditions under which this data model is minimal (i.e. both observable and controllable) will be detailed in section 9.2.

9.1.2 Backward state space model

In a similar way we can derive a backward state space model. Define the state vector $\mathbf{z_k}$ as:

$$\mathbf{z_k} = \begin{bmatrix} x(k-L+1) \\ x(k-L+2) \\ \vdots \\ x(k) \end{bmatrix}.$$

We obtain the following state and output equation

$$\mathbf{z_{k-1}} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & \cdots & \\ 1 & 0 & 0 & \cdots & \\ & \ddots & & \ddots & \\ 0 & \cdots & 1 & 0 & 0 \\ 0 & \cdots & & 1 & 0 \end{bmatrix}}_{A^H} \cdot \mathbf{z_k} + \underbrace{\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \cdot x(k-L)}_{\mathbf{q_k^b}}$$

$$\mathbf{y_k} = \underbrace{\begin{bmatrix} \mathbf{h_{L-1}} & \cdots & \mathbf{h_0} \end{bmatrix}}_{G^H} \cdot \mathbf{z_k} + \underbrace{(\mathbf{h_L} \cdot x(k-L) + \mathbf{n_k})}_{\mathbf{r_k^b}}.$$

Note that $\mathbf{z_k}$ equals $\mathbf{s_{k+1}}$.

9.1.3 Hankel matrix notation

We construct the following three block Hankel matrices, called 'past' outputs Y_{pa}, 'present' outputs Y_{pr} and 'future' outputs Y_{fu}:

$$\begin{aligned} Y_{pa} &= Y_{1|i} = \begin{bmatrix} \mathbf{y_1} & \cdots & \mathbf{y_j} \\ \vdots & & \vdots \\ \mathbf{y_i} & \cdots & \mathbf{y_{j+i-1}} \end{bmatrix} \\ Y_{pr} &= Y_{i+1|2\cdot i} = \begin{bmatrix} \mathbf{y_{i+1}} & \cdots & \mathbf{y_{j+i}} \\ \vdots & & \vdots \\ \mathbf{y_{2\cdot i}} & \cdots & \mathbf{y_{j+2\cdot i-1}} \end{bmatrix} \\ Y_{fu} &= Y_{2\cdot i+1|3\cdot i} = \begin{bmatrix} \mathbf{y_{2\cdot i+1}} & \cdots & \mathbf{y_{j+2\cdot i}} \\ \vdots & & \vdots \\ \mathbf{y_{3\cdot i}} & \cdots & \mathbf{y_{j+3\cdot i-1}} \end{bmatrix}. \end{aligned}$$

where we adopted the same notation as in previous chapters. Note that

$$Y_{pa} = \mathcal{H}_i \cdot X_{1-L|i} + N_{pa} \tag{9.2}$$
$$Y_{pr} = \mathcal{H}_i \cdot X_{i+1-L|2\cdot i} + N_{pr} \tag{9.3}$$
$$Y_{fu} = \mathcal{H}_i \cdot X_{2\cdot i+1-L|3\cdot i} + N_{fu} \tag{9.4}$$

with obvious definitions for N_{pa}, N_{pr} and N_{fu} and the Hankel input matrices as defined in equation 2.8. Here $\mathcal{H}_i$ is again an $M \cdot i \times (L+i)$ matrix.

9.2 Model assumptions

We make the following additional assumptions:

Assumption 9.2.1 *The forward and backward state space models are both observable and controllable.*

Assumption 9.2.2 $j \to \infty$. *This allows to apply stochastic properties of the data model.*

Assumption 9.2.3 $\mathcal{H}_i$ *has full column rank. This condition is satisfied if* $M \geq 2$, $i \geq L$, *the rows of* $\mathbf{H}(z)$ *have no common zeros (i.e.* $\mathbf{H}(z) \neq 0$ *for all* z*), and at least one of the rows of* $\mathbf{H}(z)$ *has degree* L, *see e.g. [97] and section 3.1.*

First we show that assumption 9.2.3 automatically implies 9.2.1. For the forward state space model we define the extended observability matrix Γ_i:

$$\begin{aligned} \Gamma_i &= \begin{bmatrix} C \\ C \cdot A \\ \vdots \\ C \cdot A^{i-1} \end{bmatrix} \\ &= \mathcal{H}_i(:, 1:L). \end{aligned} \tag{9.5}$$

The pair $\{A, C\}$ is observable if Γ_i has full column rank, which is clearly the case under assumption 9.2.3. Straightforward calculation also yields that the pair $\{A, Q^{1/2}\}$ is controllable (i.e. that all dynamical modes of the system are excited by the (unknown) input signal), i.e.

$$\text{rank}(\begin{bmatrix} A^{i-1} \cdot Q^{1/2} & \cdots & A \cdot Q^{1/2} & Q^{1/2} \end{bmatrix}) = L.$$

Similar expressions can be derived for the backward model.

9.3 Stochastic subspace algorithm

In this section we present a stochastic subspace algorithm. It consists of two steps. The first step is based on two orthogonal projections and can be considered as a noise filtering preprocessing step. In the second step the two orthogonal projections are combined and the channel parameters are identified using a subspace type approach.

9.3.1 Step 1: Orthogonal projections

Using the assumptions of the previous section, we now define two orthogonal projections. Similar to [135], we orthogonally project the present outputs Y_{pr} onto the past outputs Y_{pa}, i.e.[1][2]

$$\begin{aligned}\mathcal{O} &\triangleq Y_{pr}/Y_{pa} \\ &= \lim_{j\to\infty}(\frac{1}{j}\cdot Y_{pr}\cdot Y_{pa}^H)\cdot(\lim_{j\to\infty}(\frac{1}{j}\cdot Y_{pa}\cdot Y_{pa}^H))^\dagger\cdot Y_{pa}.\end{aligned}$$

It has been shown in [135], that under the assumptions 9.1.1, 9.1.2, 9.2.1 and 9.2.2 [3], the above equation can be rewritten as:

$$\mathcal{O} = \Gamma_i \cdot \hat{X}_{i+1-L|i} \tag{9.6}$$

which is a rank L model, as will now be explained. Equation 9.6 is valid whatever the spatial color of the noise.

Γ_i is the extended observability matrix as defined in equation 9.5:

$$\begin{aligned}\Gamma_i &= \mathcal{H}_i(:,1:L) \\ &= \begin{bmatrix} \mathbf{h_L} & \mathbf{h_{L-1}} & \cdots & \mathbf{h_1} \\ & \mathbf{h_L} & \cdots & \mathbf{h_2} \\ & & \ddots & \vdots \\ & & & \mathbf{h_L} \\ 0_{M\cdot(i-L)\times 1} & & \cdots & 0_{M\cdot(i-L)\times 1} \end{bmatrix}.\end{aligned}$$

Γ_i is thus block upper triangular and block Toeplitz.

$\hat{X}_{i+1-L|i}$ contains the non-steady state Kalman filter state estimates of the exact state sequence $\begin{bmatrix} \mathbf{s_{i+1}} & \cdots & \mathbf{s_{i+j}} \end{bmatrix}$ of the forward state space model of section 9.1.1 [135], i.e.

$$\hat{X}_{i+1-L|i} = \begin{bmatrix} \hat{\mathbf{s}}_{\mathbf{i+1}} & \cdots & \hat{\mathbf{s}}_{\mathbf{i+j}} \end{bmatrix}. \tag{9.7}$$

The matrix $\hat{X}_{i+1-L|i}$ is approximately Hankel. The Hankel structure will be better approximated whenever the noise level decreases. Forming $\hat{X}_{i+1-L|i}$ by performing an orthogonal projection can be viewed as generating a state sequence by a bank of non-steady state Kalman filters working in parallel on each of the columns of the block Hankel matrix of past outputs Y_{pa}, see [135] and figure 9.1. The bank of Kalman filters runs in a vertical direction (over

[1] In [135] all equations are expressed for real data. For complex data the transpose operator T is replaced by the Hermitian H.

[2] For a finite number of data $\mathcal{O}$ is computed as $\mathcal{O} = Y_{pr}\cdot Y_{pa}^H\cdot(Y_{pa}\cdot Y_{pa}^H)^\dagger\cdot Y_{pa}$.

[3] An additional condition is that the process noise $\mathbf{q_k}$ and the measurement noise $\mathbf{r_k}$ are not identically zero.

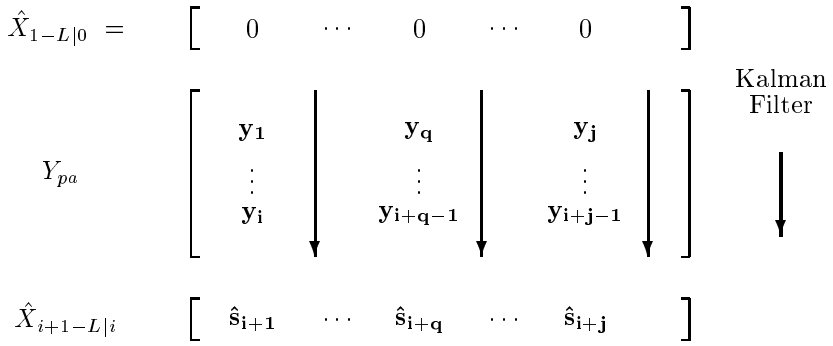


Figure 9.1: Kalman filter state estimates based upon i measurements of $\mathbf{y_k}$. If the system matrices A, C, Q, R, S were known, the state $\hat{\mathbf{s}}_{\mathbf{i+q}}$ could be determined from a Kalman filter as follows: start the filter at time q, with an initial state estimate 0. Now iterate the Kalman filter over i time steps (the vertical arrow down). The Kalman filter will then return a state estimate $\hat{\mathbf{s}}_{\mathbf{i+q}}$. This procedure could be repeated for each of the j columns, and thus we speak of a *bank* of Kalman filters. The major observation is that the system matrices A, C, Q, R, S do not have to be known to determine the state sequence $\hat{X}_{i+1-L|i}$. It can be determined directly from the output data, see [135].

the columns). They thus only use partial input-output information: i.e. the Kalman filter generating the estimate of $\hat{\mathbf{s}}_{\mathbf{i+q}}$ will only use i output measurements $\mathbf{y_q}, \ldots, \mathbf{y_{i+q-1}}$ instead of all the output measurements $\mathbf{y_1}, \ldots, \mathbf{y_{i+q-1}}$. $\hat{\mathbf{s}}_{\mathbf{i+q}}$ is thus the optimal one-step-ahead predicted state given the measurements of the outputs $\mathbf{y_q}, \ldots, \mathbf{y_{i+q-1}}$.

Similarly we project the present outputs onto the future outputs:

$$\begin{aligned}
\mathcal{B} &\stackrel{\triangle}{=} (Y_{pr}/Y_{fu}) \\
&= \lim_{j\to\infty}(\frac{1}{j}\cdot Y_{pr}\cdot Y_{fu}^H)\cdot(\lim_{j\to\infty}(\frac{1}{j}\cdot Y_{fu}\cdot Y_{fu}^H))^\dagger\cdot Y_{fu} \qquad (9.8)\\
&= \Delta_i^H\cdot\hat{\hat{X}}_{2\cdot i+1-L|2\cdot i}. \qquad (9.9)
\end{aligned}$$

In this last equation Δ_i is the reversed extended stochastic controllability matrix:

$$\begin{aligned}
\Delta_i^H &= \left[\begin{array}{cccc} A^{i-1}\cdot G & A^{i-2}\cdot G & \cdots & G \end{array}\right]^H \\
&= \mathcal{H}_i(:, i+1 : L+i)
\end{aligned}$$

$$= \begin{bmatrix} 0_{M\cdot(i-L)\times 1} & & \cdots & 0_{M\cdot(i-L)\times 1} \\ \mathbf{h_0} & & & \\ \mathbf{h_1} & \mathbf{h_0} & & \\ \vdots & & \ddots & \\ \mathbf{h_{L-1}} & \mathbf{h_{L-2}} & \cdots & \mathbf{h_0} \end{bmatrix}.$$

$\mathcal{H}_i(:, i+1 : L+i)$ is block lower triangular, block Toeplitz and contains the last L columns of $\mathcal{H}_i$ of equation 9.2.

$\hat{\hat{X}}_{2\cdot i+1-L|2\cdot i}$ equals

$$\begin{aligned} \hat{\hat{X}}_{2\cdot i+1-L|2\cdot i} &= \begin{bmatrix} \hat{\mathbf{z}}_{\mathbf{2\cdot i}} & \cdots & \hat{\mathbf{z}}_{\mathbf{2\cdot i+j-1}} \end{bmatrix} \\ &= \begin{bmatrix} \hat{\hat{\mathbf{s}}}_{\mathbf{2\cdot i+1}} & \cdots & \hat{\hat{\mathbf{s}}}_{\mathbf{2\cdot i+j}} \end{bmatrix}. \end{aligned} \tag{9.10}$$

Here $\hat{\mathbf{z}}_{\mathbf{2\cdot i+q}}$ is the optimal one-step-backward predicted state given the measured outputs $\mathbf{y_{2\cdot i+q+1}}, \ldots, \mathbf{y_{3\cdot i+q}}$. Note again that the matrix $\hat{\hat{X}}_{2\cdot i+1-L|2\cdot i}$ will only be approximately Hankel.

Remark 9.3.1 Orthogonal projections can be computed efficiently using an LQ decomposition as follows, see [135, 53]. For the computation of Y_{pr}/Y_{fu} first calculate the LQ decomposition of:

$$\begin{bmatrix} Y_{fu} \\ Y_{pr} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}.$$

The orthogonal projection is then given as:

$$Y_{pr}/Y_{fu} = L_{21} \cdot L_{11}^{\dagger} \cdot Y_{fu}.$$

When j is large compared to $M \cdot i$, the complexity is dominated by the computation of the L-factor of the LQ decomposition (i.e. the computational complexity of the pseudo-inverse of the small $(M \cdot i) \times (M \cdot i)$ matrix L_{11} can be neglected) which is of the order $(2 \cdot M \cdot i)^2 \cdot j$. This computational complexity can however further be reduced to the order $(2 \cdot M \cdot i) \cdot j$ by exploiting the block Hankel structure of the input matrix. As described in [20, 135], displacement theory can be used to obtain such fast decompositions of block Hankel matrices.

The choice of the smoothing factor i (i.e. the number of block rows in Y_{pa}, Y_{pr} and Y_{fu}) is influenced by two factors:

- Assumption 9.2.3 requires that $i \geq L$.
- Equations 9.6 and 9.9 state that it is not useful choosing i larger than L since the corresponding rows of $\mathcal{O}$ and $\mathcal{B}$ will be zero (for $j \to \infty$). For finite j the entries in these rows will be small.

In general, one should choose i larger than the maximum channel order to be expected. Inspecting the matrices $\mathcal{O}$ and $\mathcal{B}$ then allows to determine the channel order L.

After the order has been detected we drop the last $M \cdot (i - L)$ rows of $\mathcal{O}$ and the first $M \cdot (i - L)$ rows of $\mathcal{B}$ (which are approximately zero). I.e. after the orthogonal projection $\mathcal{O}$ and $\mathcal{B}$ are redefined as:

$$\begin{aligned} \underline{\mathcal{O}} &\triangleq \mathcal{O}(1 : L \cdot M, :) \\ \overline{\mathcal{B}} &\triangleq \mathcal{B}((i-L) \cdot M + 1 : i \cdot M, :). \end{aligned}$$

The second step of the algorithm then retrieves the channel parameters by combining both orthogonal projections. Both projections are needed since neither the forward nor backward projection model contains all channel parameters.

9.3.2 Step 2: Channel identification

The second step of the algorithm retrieves the channel parameters using a subspace approach. First we add $\underline{\mathcal{O}}$ and $\overline{\mathcal{B}}$:

$$\underline{\mathcal{O}} + \overline{\mathcal{B}} = \underbrace{\begin{bmatrix} \mathcal{H}_L(:, 1:L) & \mathcal{H}_L(:, L+1:L+i) \end{bmatrix}}_{\mathcal{H}_L} \cdot \begin{bmatrix} \hat{X}_{i+1-L|i} \\ \hat{X}_{2\cdot i+1-L|2\cdot i} \end{bmatrix}. \quad (9.11)$$

The matrix $\begin{bmatrix} \hat{X}_{i+1-L|i} \\ \hat{X}_{2\cdot i+1-L|2\cdot i} \end{bmatrix}$ will have full row rank. The matrices $\underline{\mathcal{O}} + \overline{\mathcal{B}}$ and $\mathcal{H}_L$ will thus share the same column space. The next theorem which comes from [2] gives the conditions under which the channel parameters can be identified from this column space:

Theorem 9.3.2 *If the rows of* $\mathbf{H}(z)$ *do not have common zeros and if* $\mathbf{H}(z)$ *does not lie in a two-dimensional subspace (which implies* $M \geq 3$, $L \geq 2$*), and*

$$U^{\perp} \cdot \mathcal{F}_L = 0$$

with $\mathcal{F}_L$ *a matrix having the same structure as* $\mathcal{H}_L$ *and* $U^{\perp}$ *the left null space of* $\mathcal{H}_L$*, then* $\mathcal{F}_L = \alpha \cdot \mathcal{H}_L$ *with* α *an unknown constant.*

From Theorem 9.3.2, it is clear that the channel parameters can be retrieved from the null space of $\underline{\mathcal{O}} + \overline{\mathcal{B}}$ by exploiting the structure of $\mathcal{H}_L$.

We now proceed in the following way. Compute the SVD of $\underline{\mathcal{O}} + \overline{\mathcal{B}}$:

$$\underline{\mathcal{O}} + \overline{\mathcal{B}} = U \cdot \Sigma \cdot V^H$$

$$= \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}.$$

Note that Σ has $2 \cdot L$ non-zero singular values. This provides a way to check whether the order was correctly detected and enough rows were removed in $\underline{\mathcal{O}}$ and $\overline{\mathcal{B}}$.

Next, the left null space $U^\perp$ of $\underline{\mathcal{O}} + \overline{\mathcal{B}}$ is extracted, such that:

$$U^{\perp H} \cdot \mathcal{H}_L = 0. \tag{9.12}$$

Now split $U^{\perp H}$ in L submatrices of equal dimensions:

$$U^{\perp H} = \begin{bmatrix} U_1^\perp & \cdots & U_L^\perp \end{bmatrix}.$$

Finally we rewrite equation 9.12 as a function of the unknown channel parameters:

$$\begin{bmatrix} U_L^\perp & & & \\ \vdots & \ddots & & \\ U_1^\perp & \cdots & U_L^\perp & \\ & U_1^\perp & \cdots & U_L^\perp \\ & & \ddots & \vdots \\ & & & U_1^\perp \end{bmatrix} \cdot \begin{bmatrix} \mathbf{h_0} \\ \vdots \\ \mathbf{h_L} \end{bmatrix} = 0.$$

Using a unit norm constraint, the channel coefficient vector can be estimated as the right singular vector corresponding to the smallest singular value of the above matrix. The algorithm is summarized as algorithm 9.3.2.

This algorithm will provide consistent estimates of the channel parameters since for $j \to \infty$, equation 9.11 will hold exactly, in spite of the possibly spatially colored noise (assumption 9.1.1). Theorem 9.3.2 then ensures the exact determination of the channel coefficients (up to an unknown scaling factor).

9.4 Conclusions

In this chapter we have presented a new blind channel estimation algorithm based on a stochastic state space description of the data model and two orthogonal subspace projections. The algorithm is robust against the spatial color of the noise: the algorithm provides consistent channel estimates even if the noise is spatially colored. As will be shown in chapter 11, the algorithm has a better performance than the algorithm of [2] (this algorithm is summarized in section 11.3), although it is computationally somewhat more complex.

Algorithm 9.3.2
Stochastic subspace algorithm

1. *compute* $\mathcal{O}$ *and* $\mathcal{B}$:

$$\begin{aligned}
\begin{bmatrix} Y_{pa} \\ Y_{pr} \end{bmatrix} &= \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} \\
\mathcal{O} &= Y_{pr}/Y_{pa} \\
&= L_{21} \cdot L_{11}^{\dagger} \cdot Y_{pa} \\
\begin{bmatrix} Y_{fu} \\ Y_{pr} \end{bmatrix} &= \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} \\
\mathcal{B} &= Y_{pr}/Y_{fu} \\
&= L_{21} \cdot L_{11}^{\dagger} \cdot Y_{fu}
\end{aligned}$$

2. *remove the (approximately) zero rows*

$$\begin{aligned}
\underline{\mathcal{O}} &= \mathcal{O}(1 : L \cdot M, :) \\
\overline{\mathcal{B}} &= \mathcal{B}((i-L) \cdot M + 1 : i \cdot M, :)
\end{aligned}$$

3. *compute the SVD of* $\underline{\mathcal{O}} + \overline{\mathcal{B}}$:

$$\begin{aligned}
\underline{\mathcal{O}} + \overline{\mathcal{B}} &= \begin{bmatrix} U^s & U^{\perp} \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix} \\
U^{\perp H} &= \begin{bmatrix} U_1^{\perp} & \cdots & U_L^{\perp} \end{bmatrix}
\end{aligned}$$

4. *estimate the channel parameters from the right singular vector corresponding to the smallest singular value of*:

$$\begin{bmatrix}
U_L^{\perp} & & & \\
\vdots & \ddots & & \\
U_1^{\perp} & \cdots & U_L^{\perp} & \\
 & U_1^{\perp} & \cdots & U_L^{\perp} \\
 & & \ddots & \vdots \\
 & & & U_1^{\perp}
\end{bmatrix}$$

Chapter 10

Two deterministic subspace algorithms based on oblique projections

In this chapter we consider the problem of deterministic blind channel estimation using *oblique projections*. Oblique projections [12, 135] allow to decompose a matrix into two non-orthogonal components (whereas the orthogonal projections used in the previous chapter decompose a matrix into two orthogonal components). This property allows to develop *two new blind channel estimation algorithms. The algorithms are robust against the spatial color of the additive noise and they are deterministic: if there is no additive noise, they allow perfect channel recovery with a finite number of samples if the model assumptions hold.*

The *first algorithm* uses row space information to recover the channel parameters. The first step in the algorithm is based on two oblique projections. The oblique projections decompose the channel estimation problem into two matrix equations. In a second step, the Hankel structure of the matrix of input symbols is exploited and the channel parameters are estimated from a rank-one matrix approximation.

The *second algorithm* recovers the channel parameters using column space information, as the algorithms [97, 2] and the stochastic algorithm presented in the previous chapter. The first step in the algorithm is again based on two oblique projections and can be considered as a noise filtering preprocessing step. In the second step, the channel parameters are estimated using a subspace approach.

The outline of this chapter is as follows. In section 10.1 we review the concept of

oblique projection. Section 10.2 presents the two deterministic subspace algorithms. Section 10.3 discusses the problem of channel order estimation. Finally section 10.4 presents some conclusions. The performance of the algorithms will be analyzed in chapter 11.

10.1 The oblique projection

In this section we review the concept of oblique projections. We adhere to the notation of [135].

Definition 10.1.1 *The oblique projection* $A/_B\ C$ *can be defined as follows:*

1. *Project the row space of a matrix* $A \in \mathbb{C}^{p\times j}$ *orthogonally on the joint row space of the matrices* $B \in \mathbb{C}^{q\times j}$ *and* $C \in \mathbb{C}^{r\times j}$*:*

$$\begin{aligned} A/\begin{bmatrix} B \\ C \end{bmatrix} &= A \cdot \begin{bmatrix} B^H & C^H \end{bmatrix} \cdot \begin{bmatrix} B\cdot B^H & B\cdot C^H \\ C\cdot B^H & C\cdot C^H \end{bmatrix}^\dagger \cdot \begin{bmatrix} B \\ C \end{bmatrix} \\ &= \begin{bmatrix} M & | & N \end{bmatrix} \cdot \left[\begin{array}{c} B \\ \hline C \end{array}\right], \end{aligned}$$

2. *and decompose the result along the row space of* B *onto the row space of* C [1]*:*

$$A/_B\ C = N \cdot C.$$

The oblique projection has a number of interesting properties which provide further insight.

Property 10.1.1 *When* $B = 0$ *or when the row space of* B *is orthogonal to the row space of* C *(*$B\cdot C^H = 0$*), the oblique projection reduces to an orthogonal projection:*

$$\begin{aligned} A/_B\ C &= A/C \\ &= A\cdot C^H \cdot (C\cdot C^H)^\dagger \cdot C. \end{aligned} \tag{10.1}$$

Property 10.1.2 *If:*

[1] For an infinite number of data the oblique projection is defined as:
$A/_B\ C = \begin{bmatrix} \phi_{AB} & \phi_{AC} \end{bmatrix} \cdot \begin{bmatrix} \phi_{BB} & \phi_{BC} \\ \phi_{CB} & \phi_{CC} \end{bmatrix}^\dagger \cdot \begin{bmatrix} 0 \\ I_r \end{bmatrix} \cdot C$, with $\phi_{XY} = \lim_{j\to\infty}(\frac{1}{j} X \cdot Y^H)$.

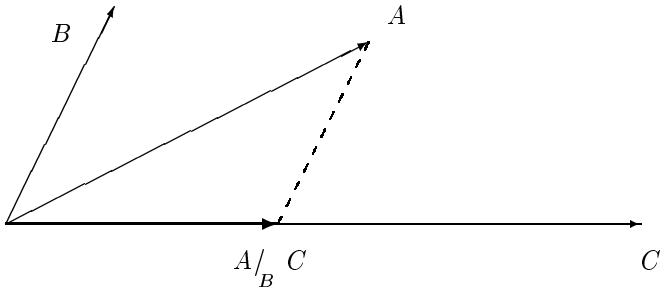


Figure 10.1: Interpretation of the oblique projection in the j-dimensional space ($j = 2$ in this case). The oblique projection is formed by projecting the row space of A along the row space of B on the row space of C.

- $A = M \cdot B + N \cdot C$
- *and the intersection of the row spaces of* B *and* C *is empty,*

then

$$A/_B\ C = N \cdot C \tag{10.2}$$
$$A/_C\ B = M \cdot B. \tag{10.3}$$

It is clear that when A lies in the joint row spaces of B and C, the projection $A/\left[\begin{smallmatrix} B \\ C \end{smallmatrix}\right] = A$ and the above result easily follows. Figure 10.1 depicts this decomposition. Note that if B and/or C do not have full row rank, A can also be written as:

$$A = (M + P_1 \cdot B^{\perp}) \cdot B + (N + P_2 \cdot C^{\perp}) \cdot C.$$

where $B^{\perp}$ and $C^{\perp}$ are the left null spaces of respectively B and C, and P_1, P_2 are random matrices of appropriate dimensions. This does however not alter the equations 10.2 and 10.3. For a rigorous proof of property 10.1.2, see the proof of theorem 2 in [135].

Property 10.1.3 *The oblique projection can be efficiently computed from an LQ decomposition, see also [135, 53]. If the LQ decomposition of* $\left[\begin{smallmatrix} B \\ C \\ A \end{smallmatrix}\right]$ *equals:*

$$\begin{bmatrix} B \\ C \\ A \end{bmatrix} = \begin{bmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix}$$

then the orthogonal projection of A onto the joint row space of B and C equals:

$$A/\begin{bmatrix} B \\ C \end{bmatrix} = \underbrace{\begin{bmatrix} L_{31} & L_{32} \end{bmatrix} \cdot \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix}^{\dagger}}_{\begin{bmatrix} L_B & L_C \end{bmatrix}} \cdot \begin{bmatrix} B \\ C \end{bmatrix}.$$

The oblique projection $A/_B\ C$ equals $L_C \cdot C$ (and $A/_C\ B$ equals $L_B \cdot B$).

It is important to note the computation of the oblique projection does not require the calculation of the (possibly large) matrix Q of the LQ decomposition.

10.2 Deterministic subspace algorithms

For the derivations below we again use the single user data model of equation 2.9. We further assume that $\mathcal{H}_i$ has full column rank, i.e. that assumption 9.2.3 holds and that:

Assumption 10.2.1 *The system input is "sufficiently excited" (see also section 2.4.3) such that the following matrix has full row rank:*

$$rank\left(\begin{bmatrix} X_{1-L|i} \\ X_{i+1|2i+L} \end{bmatrix}\right) = 2 \cdot (L+i).$$

In this section, we assume that the channel order is known. In section 10.3 we show how the channel order can be estimated. We assume noiseless measurements, the influence of noise will be studied in section 10.2.2.

First we divide the available output samples into three blocks, which we call respectively 'past' outputs Y_{pa}, 'present' outputs Y_{pr}, and 'future' outputs Y_{fu}:

$$\left[\begin{array}{c} Y_{pa} \\ \hline Y_{pr} \\ \hline Y_{fu} \end{array}\right] = \left[\begin{array}{c} Y_{1|i} \\ \hline Y_{i+1|i+L} \\ \hline Y_{i+L+1|2\cdot i+L} \end{array}\right]$$

$$= \left[\begin{array}{ccc} \mathbf{y_1} & \cdots & \mathbf{y_j} \\ \vdots & & \vdots \\ \mathbf{y_i} & \cdots & \mathbf{y_{i+j-1}} \\ \hline \mathbf{y_{i+1}} & \cdots & \mathbf{y_{i+j}} \\ \vdots & & \vdots \\ \mathbf{y_{i+L}} & \cdots & \mathbf{y_{i+L+j-1}} \\ \hline \mathbf{y_{i+L+1}} & \cdots & \mathbf{y_{i+L+j}} \\ \vdots & & \vdots \\ \mathbf{y_{2\cdot i+L}} & \cdots & \mathbf{y_{2\cdot i+L+j-1}} \end{array}\right]. \tag{10.4}$$

Note that the division between past present and future outputs is slightly different than that in the previous chapter. The first step in both algorithms computes the following two oblique projections:

$$Z_1 = Y_{pr}/_{Y_{fu}} Y_{pa} \tag{10.5}$$

$$Z_2 = Y_{pr}/_{Y_{pa}} Y_{fu}. \tag{10.6}$$

To compute Z_1, we project the present outputs along the future outputs onto the past outputs, and for Z_2 we project the present outputs along the past outputs onto the future outputs. As will be shown next, Z_1 and Z_2 equal:

$$Z_1 = \mathcal{H}_L(:, 1:L) \cdot X_{i+1-L|i} \tag{10.7}$$

$$Z_2 = \mathcal{H}_L(:, L+1:2\cdot L) \cdot X_{i+1|i+L} \tag{10.8}$$

such that:

$$Y_{pr} = Z_1 + Z_2.$$

The matrices $\mathcal{H}_L(:, 1:L)$ and $\mathcal{H}_L(:, L+1:2\cdot L)$ equal:

$$\mathcal{H}_L(:,1:L) = \left[\begin{array}{ccc} \mathbf{h_L} & \cdots & \mathbf{h_1} \\ & \ddots & \vdots \\ & & \mathbf{h_L} \end{array}\right], \quad \mathcal{H}_L(:,L+1:2\cdot L) = \left[\begin{array}{ccc} \mathbf{h_0} & & \\ \vdots & \ddots & \\ \mathbf{h_{L-1}} & \cdots & \mathbf{h_0} \end{array}\right].$$

Both oblique projections Z_1 and Z_2 are rank-L matrices (see equations 10.7 and 10.8) and can be computed from one LQ decomposition, see property 10.1.3. We now show that equation 10.7 holds, the proof of equation 10.8 is similar.

Proof: We prove that $Y_{pr} = Z_1 + Z_2$, by writing Y_{pr} as a linear combination of Y_{pa} and Y_{fu}:

$$Y_{pr} = \mathcal{H}_L \cdot \left[\frac{X_{i+1-L|i}}{X_{i+1|i+L}}\right]$$

$$
\begin{aligned}
&= \mathcal{H}_L \cdot \left[\begin{array}{c} \left[\begin{array}{cc} 0 & I_L \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{pa} \\ \hline \left[\begin{array}{cc} I_L & 0 \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{fu} \end{array} \right] \\
&= \left[\begin{array}{c|c} \mathcal{H}_L(:,1:L) & \mathcal{H}_L(:,L+1:2\cdot L) \end{array}\right] \cdot \left[\begin{array}{c} \left[\begin{array}{cc} 0 & I_L \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{pa} \\ \hline \left[\begin{array}{cc} I_L & 0 \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{fu} \end{array} \right] \\
&= \left[\begin{array}{cc} 0 & \mathcal{H}_L(:,1:L) \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{pa} + \left[\begin{array}{cc} \mathcal{H}_L(:,L+1:2\cdot L) & 0 \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{fu}.
\end{aligned}
$$

Note that from assumption 9.2.3, $\mathcal{H}_i$ is indeed left invertible. Further from assumption 10.2.1 we know that the intersection of the row spaces of $Y_{pa} = Y_{1|i}$ and $Y_{fu} = Y_{i+L+1|2i+L}$ is empty. Hence, using property 10.1.2, the first oblique projection equals:

$$
\begin{aligned}
Z_1 &= \left[\begin{array}{cc} 0 & \mathcal{H}_L(:,1:L) \end{array}\right] \cdot \mathcal{H}_i^\dagger \cdot Y_{1|i} \\
&= \left[\begin{array}{ccc} 0 & \mathbf{h_L} & \cdots & \mathbf{h_1} \\ \vdots & & \ddots & \vdots \\ 0 & & & \mathbf{h_L} \end{array}\right] \cdot X_{1-L|i} \\
&= \left[\begin{array}{ccc} \mathbf{h_L} & \cdots & \mathbf{h_1} \\ & \ddots & \vdots \\ & & \mathbf{h_L} \end{array}\right] \cdot \left[\begin{array}{ccc} x(i+1-L) & \cdots & x(i+j-L) \\ \vdots & & \vdots \\ x(i) & \cdots & x(i+j-1) \end{array}\right].
\end{aligned}
$$

The expression for Z_2 can be proven in a similar way. □

Since neither the expression for Z_1 nor the expression for Z_2 contains all channel parameters, both have to be combined. Depending on the way they are combined different algorithms result.

10.2.1 Deterministic subspace algorithm 1

Algorithm 1 retrieves the channel parameters from the oblique projections of equation 10.5 and 10.6 as follows. Define $\mathcal{P}$ as:

$$
\mathcal{P} \triangleq \left[\begin{array}{c} Z_1(1:(L-1)\cdot M, 2:j) - Z_1(M+1:L\cdot M, 1:j-1) \\ Z_1((L-1)\cdot M+1:L\cdot M, 2:j) \end{array}\right].
$$

This can be rewritten as:

$$
\mathcal{P} = \left[\begin{array}{ccc|c} \mathbf{h_L} & \cdots & \mathbf{h_2} & \mathbf{h_1} \\ & \ddots & & \vdots \\ & & \mathbf{h_L} & \mathbf{h_{L-1}} \\ & & & \mathbf{h_L} \end{array}\right] \cdot \left[\begin{array}{ccc} x(i+2-L) & \cdots & x(i+j-L) \\ \vdots & & \vdots \\ x(i) & & x(i+j-2) \\ \hline x(i+1) & \cdots & x(i+j-1) \end{array}\right]
$$

$$
\begin{aligned}
- & \left[\begin{array}{c|ccc} 0 & \mathbf{h_L} & \cdots & \mathbf{h_2} \\ \vdots & & \ddots & \\ & & & \mathbf{h_L} \\ 0 & 0 & \cdots & 0 \end{array}\right] \cdot \left[\begin{array}{ccc} x(i+1-L) & \cdots & x(i+j-L-1) \\ \hline x(i+2-L) & \cdots & x(i+j-L) \\ \vdots & & \vdots \\ x(i) & \cdots & x(i+j-2) \end{array}\right] \\
= & \left[\begin{array}{c} \mathbf{h_1} \\ \vdots \\ \mathbf{h_L} \end{array}\right] \cdot \left[\begin{array}{ccc} x(i+1) & \cdots & x(i+j-1) \end{array}\right]. \qquad (10.9)
\end{aligned}
$$

Without noise, we obtain a rank-one model. In a similar way it can be shown that:

$$
\begin{aligned}
\mathcal{Q} & \triangleq \left[\begin{array}{c} Z_2(1:M, 1:j-1) \\ Z_2(M+1:(L+1)\cdot M, 1:j-1) - Z_2(1:L\cdot M, 2:j) \end{array}\right] \\
& = \left[\begin{array}{c} \mathbf{h_0} \\ \vdots \\ \mathbf{h_{L-1}} \end{array}\right] \cdot \left[\begin{array}{ccc} x(i+1) & \cdots & x(i+j-1) \end{array}\right].
\end{aligned}
$$

When $\mathcal{P}$ and $\mathcal{Q}$ are added, we obtain a rank-one model that contains all channel parameters:

$$
\left[\begin{array}{c} 0 \\ \mathcal{P} \end{array}\right] + \left[\begin{array}{c} \mathcal{Q} \\ 0 \end{array}\right] = \left[\begin{array}{c} \mathbf{h_0} \\ 2\cdot\mathbf{h_1} \\ \vdots \\ 2\cdot\mathbf{h_{L-1}} \\ \mathbf{h_L} \end{array}\right] \cdot \left[\begin{array}{ccc} x(i+1) & \cdots & x(i+j-1) \end{array}\right]. \qquad (10.10)
$$

When noise corrupts the measurements the matrix $\left[\begin{smallmatrix} 0 \\ \mathcal{P} \end{smallmatrix}\right] + \left[\begin{smallmatrix} \mathcal{Q} \\ 0 \end{smallmatrix}\right]$ will no longer be exactly rank 1. Let the SVD of $\left[\begin{smallmatrix} 0 \\ \mathcal{P} \end{smallmatrix}\right] + \left[\begin{smallmatrix} \mathcal{Q} \\ 0 \end{smallmatrix}\right]$ be:

$$
\left[\begin{array}{c} 0 \\ \mathcal{P} \end{array}\right] + \left[\begin{array}{c} \mathcal{Q} \\ 0 \end{array}\right] = U \cdot \Sigma \cdot V^H
$$

then an estimate of the channel parameters can be obtained as the left singular vector corresponding to the largest singular value of Σ, i.e.:

$$
\left[\begin{array}{c} \mathbf{h_0} \\ 2\cdot\mathbf{h_1} \\ \vdots \\ 2\cdot\mathbf{h_{L-1}} \\ \mathbf{h_L} \end{array}\right] = U(:,1).
$$

As a byproduct of the algorithm, the right singular vector corresponding the largest singular value provides an estimate of the transmitted symbol sequence. The final algorithm is summarized as algorithm 10.2.1.

Algorithm 10.2.1
Deterministic subspace algorithm 1

1. *compute the LQ decomposition of:*

$$\begin{bmatrix} Y_{pa} \\ Y_{fu} \\ Y_{pr} \end{bmatrix} = \begin{bmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix}$$

2. *determine the channel order, see section 10.3*
3. *compute Z_1 and Z_2:*

$$Z_1 = \begin{bmatrix} L_{31} & L_{32} \end{bmatrix} \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}^{\dagger} \begin{bmatrix} I_{M \cdot i} \\ 0 \end{bmatrix} Y_{pa}$$

$$Z_2 = \begin{bmatrix} L_{31} & L_{32} \end{bmatrix} \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}^{\dagger} \begin{bmatrix} 0 \\ I_{M \cdot i} \end{bmatrix} Y_{fu}$$

4. *form the matrices $\mathcal{P}$ and $\mathcal{Q}$:*

$$\mathcal{P} = \begin{bmatrix} Z_1(1:(L-1)\cdot M, 2:j) - Z_1(M+1:L\cdot M, 1:j-1) \\ Z_1((L-1)\cdot M+1:L\cdot M, 2:j) \end{bmatrix}$$

$$\mathcal{Q} = \begin{bmatrix} Z_2(1:M, 1:j-1) \\ Z_2(M+1:(L+1)\cdot M, 1:j-1) - Z_2(1:L\cdot M, 2:j) \end{bmatrix}$$

5. *compute the left singular vector corresponding to the largest singular value of* $\begin{bmatrix} 0 \\ \mathcal{P} \end{bmatrix} + \begin{bmatrix} \mathcal{Q} \\ 0 \end{bmatrix}$*:*

$$\begin{bmatrix} \mathbf{h_0} \\ 2 \cdot \mathbf{h_1} \\ \vdots \\ 2 \cdot \mathbf{h_{L-1}} \\ \mathbf{h_L} \end{bmatrix} = U(:,1)$$

10.2.2 Further discussion

Algorithm 10.2.1 is deterministic in the sense that it provides perfect channel estimates with a finite number of data, as long as the model assumptions hold and no noise corrupts the measurements. For an analysis of the algorithm in the presence of noise, we study the behavior of the oblique projections using noisy data, when the number of columns of Y_{pa}, Y_{pr} and Y_{fu} tends to infinity, i.e. when $j \to \infty$.

For the asymptotic analysis of the oblique projections, we make the same as-

sumptions as for the stochastic algorithm of chapter 9. We assume that assumptions 9.1.1 and 9.1.2 on source symbols and noise properties hold. The noise is thus assumed to be temporally white but can be spatially colored. We further assume an infinite number of data (assumption 9.2.2).

In the derivation of algorithm 10.2.1, we have assumed that the matrix $X_{1-L|2\cdot i+L}$ has full row rank, see assumption 10.2.1. This assumption is very likely to hold, even for moderate values of j (the number of columns of $X_{1-L|2\cdot i+L}$) as long as the input is sufficiently excited. When the assumptions 9.1.2 and 9.2.2 hold, $X_{1-L|2\cdot i+L}$ has not only full row rank but the rows of $X_{1-L|2\cdot i+L}$ are also orthogonal.

Hence, under the assumptions 9.1.1, 9.1.2 and 9.2.2, the past and future outputs Y_{pa} and Y_{fu} will be asymptotically orthogonal:

$$\begin{aligned}
&\lim_{j\to\infty} \frac{1}{j}(Y_{pa} \cdot Y_{fu}^H) \\
&= \lim_{j\to\infty} \frac{1}{j}(Y_{1|i} \cdot Y_{i+L+1|2\cdot i+L}^H) \\
&= \lim_{j\to\infty} (\frac{1}{j}(\mathcal{H}_i \cdot X_{1-L|i} + N_{1|i}) \cdot (X_{i+1|2\cdot i+L}^H \cdot \mathcal{H}_i^H + N_{i+L+1|2\cdot i+L}^H)) \\
&= 0.
\end{aligned}$$

If two matrices B and C are orthogonal, then (see also property 10.1.1 and equation 10.1):

$$A/_B\ C = A/C.$$

This means that, as j goes to infinity, the two oblique projections Z_1 and Z_2 of equations 10.5 and 10.6 can be rewritten as orthogonal projections, i.e.:

$$\lim_{j\to\infty}(Z_1) = Y_{pr}/Y_{pa} \tag{10.11}$$

$$\lim_{j\to\infty}(Z_2) = Y_{pr}/Y_{fu}. \tag{10.12}$$

These expressions are (almost) identical to those of the stochastic subspace algorithm presented in the previous chapter. Based on the results of the previous chapter, Z_1 and Z_2, as defined in equations 10.11 and 10.12, equal:

$$\lim_{j\to\infty}(Z_1) = H_L(:, 1:L) \cdot \hat{X}_{i+1-L|i} \tag{10.13}$$

$$\lim_{j\to\infty}(Z_2) = H_L(:, L+1:2\cdot L) \cdot \hat{\hat{X}}_{i+1|i+L} \tag{10.14}$$

where $\hat{X}_{i+1-L|i}$ and $\hat{\hat{X}}_{i+1|i+L}$ are Kalman filter state estimates (corresponding to respectively a forward and backward state space model (see chapter 9)).

Even for $j \to \infty$, the matrices $\hat{X}_{i+1-L|i}$ and $\hat{\hat{X}}_{i+1|i+L}$ will only be approximately Hankel, see [135, 146]. So even when the number of observations goes to infinity, equation 10.10 will only be approximately rank 1. As will be shown in the simulation results of chapter 11, the mean squared error (MSE) curve of the channel estimate of the algorithm 10.2.1 flattens out for increasing burst length (i.e. increasing j). Therefore the algorithm should preferably be used for short bursts. On the other hand the Kalman filter state estimates take into account the (unknown) spatial noise covariance and hence make algorithm 10.2.1 robust with respect to the spatial noise covariance (as will be illustrated by simulation examples).

Another important observation is that for $j \to \infty$, Z_1 and Z_2 both become rank deficient rank-L matrices, and their column spaces are parameterized by the channel parameters. This will be exploited in the second algorithm. The second algorithm, which is presented in the next section, recovers the channel parameters from the oblique projections Z_1 and Z_2, only relying on column space information.

10.2.3 Deterministic subspace algorithm 2

Using the results from the previous section, we now develop the second algorithm along the same lines of the stochastic algorithm presented in chapter 9. However, the algorithm developed here is deterministic (in the same sense as algorithm 10.2.1) and provides perfect channel estimates when the number of observations goes to infinity.

The second algorithm also consists of two steps. The first step is based on oblique projections as in algorithm 10.2.1. This step can be considered as a noise filtering preprocessing step, which makes the algorithm robust against the spatial color of the additive noise. In the second step the channel parameters are then recovered using a subspace approach similar to that of the stochastic algorithm of chapter 9.

For the derivation of the second algorithm, we again rely on Theorem 9.3.2. The algorithm proceeds as follows. In a first step we compute the two rank-L oblique projections Z_1 and Z_2 as given in equations 10.5 and 10.6. Then we add Z_1 and Z_2 and obtain in the noiseless case:

$$Z_1 + Z_2 = \underbrace{\begin{bmatrix} \mathcal{H}_L(:, 1:L) & \mathcal{H}_L(:, L+1:2\cdot L) \end{bmatrix}}_{\mathcal{H}_L} \cdot X_{i+1-L|i+L}$$

or in the noisy case with $j \to \infty$:

$$\lim_{j\to\infty}(Z_1 + Z_2) = \underbrace{\begin{bmatrix} \mathcal{H}_L(:, 1:L) & \mathcal{H}_L(:, L+1:2\cdot L) \end{bmatrix}}_{\mathcal{H}_L} \cdot \begin{bmatrix} \hat{X}_{i+1-L|i} \\ \hat{\hat{X}}_{i+1|i+L} \end{bmatrix}.$$

Note that in the noiseless case $Z_1 + Z_2$ will exactly equal $Y_{pr} = Y_{i+1|i+L}$ and the preprocessing step will have no effect. With noise and $j \to \infty$, $Z_1 + Z_2$ will form a rank-$2 \cdot L$ estimate of Y_{pr}. In both equations, the matrices $Z_1 + Z_2$ and $\mathcal{H}_L$ will share the same column space. If the rank conditions of Theorem 9.3.2 hold, the channel parameters can be determined from this column space. The channel parameters are now recovered in the same way as for the stochastic algorithm of chapter 9. Compute the SVD of $Z_1 + Z_2$:

$$\begin{aligned} Z_1 + Z_2 &= U \cdot \Sigma \cdot V^H \\ &= \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}. \end{aligned} \tag{10.15}$$

Extract the left null space $U^\perp$ of $Z_1 + Z_2$, such that:

$$U^{\perp H} \cdot \mathcal{H}_L = 0 \quad \text{and} \quad U^{\perp H} = \begin{bmatrix} U_1^\perp & \cdots & U_L^\perp \end{bmatrix}. \tag{10.16}$$

Finally we rewrite equation 10.16 as a function of the unknown channel parameters:

$$\begin{bmatrix} U_L^\perp & & & \\ \vdots & \ddots & & \\ U_1^\perp & \cdots & U_L^\perp & \\ & U_1^\perp & \cdots & U_L^\perp \\ & & \ddots & \vdots \\ & & & U_1^\perp \end{bmatrix} \cdot \begin{bmatrix} \mathbf{h_0} \\ \vdots \\ \mathbf{h_L} \end{bmatrix} = 0.$$

The channel coefficient vector can then be estimated as the right singular vector corresponding to the smallest singular value of the above matrix.

Remark 10.2.2 Note that $Z_1 + Z_2 = Y_{pr} / \begin{bmatrix} Y_{pa} \\ Y_{fu} \end{bmatrix}$. In practice, we will compute $Z_1 + Z_2$ from one orthogonal projection, rather than adding two oblique projections.

Remark 10.2.3 Note that

$$\lim_{j \to \infty} \left(\frac{1}{j} \begin{bmatrix} Y_{pa} \\ Y_{fu} \end{bmatrix} \cdot N_{pr}^H \right) = 0.$$

Hence the projection of Y_{pr} into the joint row spaces of Y_{pa} and Y_{fu}, projects away the noise component. The projection can thus also be interpreted in an instrumental variable framework [85].

The complete algorithm is summarized as algorithm 10.2.3.

Algorithm 10.2.3
Deterministic subspace algorithm 2

1. *compute the LQ decomposition of:*

$$\begin{bmatrix} Y_{pa} \\ Y_{fu} \\ Y_{pr} \end{bmatrix} = \begin{bmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix}$$

2. *determine the channel order, see section 10.3*
3. *compute $Z_1 + Z_2$:*

$$Z_1 + Z_2 = \begin{bmatrix} L_{31} & L_{32} \end{bmatrix} \cdot \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}^{\dagger} \begin{bmatrix} Y_{pa} \\ Y_{fu} \end{bmatrix}$$

4. *compute the SVD of $Z_1 + Z_2$:*

$$\begin{aligned} Z_1 + Z_2 &= \begin{bmatrix} U^s & U^{\perp} \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix} \\ U^{\perp H} &= \begin{bmatrix} U_1^{\perp} & \cdots & U_L^{\perp} \end{bmatrix} \end{aligned}$$

5. *compute the right singular vector corresponding to the smallest singular value of:*

$$\begin{bmatrix} U_L^{\perp} & & & \\ \vdots & \ddots & & \\ U_1^{\perp} & \cdots & U_L^{\perp} & \\ & U_1^{\perp} & \cdots & U_L^{\perp} \\ & & \ddots & \vdots \\ & & & U_1^{\perp} \end{bmatrix}$$

Remark 10.2.4 Since we are only interested in the column space of $Z_1 + Z_2$, $Z_1 + Z_2$ can be replaced by:

$$\begin{bmatrix} L_{31} & L_{32} \end{bmatrix} \cdot \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}^{\dagger} \cdot \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix},$$

which has smaller dimensions.

10.3 Channel order estimation

So far, we have assumed prior knowledge of the channel order. In this section, we succinctly discuss several channel order estimation methods.

From property 10.1.3 we know that the oblique projection $Z_1 = Y_{pr}/_{Y_{fu}} Y_{pa}$ is computed from an LQ decomposition of:

$$\begin{bmatrix} Y_{pa} \\ Y_{fu} \\ Y_{pr} \end{bmatrix} = \begin{bmatrix} Y_{1|i} \\ Y_{i+L+1|2\cdot i+L} \\ Y_{i+1|i+L} \end{bmatrix} = \begin{bmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix}$$

as,

$$Z_1 = \begin{bmatrix} L_{31} & L_{32} \end{bmatrix} \cdot \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}^{\dagger} \begin{bmatrix} I_{M\cdot i} \\ 0 \end{bmatrix} Y_{pa}.$$

Without noise $\left[\begin{smallmatrix} L_{11} & \\ L_{21} & L_{22} \end{smallmatrix}\right]$ has exactly $2 \cdot (i + L)$ singular values different from zero, the order of the system can thus be estimated from the SVD spectrum of the small $2Mi \times 2Mi$ matrix $\left[\begin{smallmatrix} L_{11} & \\ L_{21} & L_{22} \end{smallmatrix}\right]$. Alternatively, when the channel order is estimated correctly, $Y_{pr} = Y_{i+1|i+L}$ lies in the joint row space of $Y_{pa} = Y_{1|i}$ and $Y_{fu} = Y_{i+L+1|2\cdot i+L}$, and hence $L_{33} = 0$ (no noise). If the channel order is overestimated as $\hat{L} > L$, $Y_{i+1|i+\hat{L}}$ will no longer lie in the joint row space of $Y_{1|i}$ and $Y_{i+\hat{L}+1|2\cdot i+\hat{L}}$ and hence $L_{33} \neq 0$. Therefore one could alternatively decrease the estimated channel order till the norm of L_{33} drops below a predefined threshold, which determines the channel order.

When the number of observations is large, it is possible to exploit the fact that for $\hat{L} \geq L$: $\lim_{j\to\infty}(\frac{1}{j} \cdot Y_{pa} \cdot Y_{fu}^H) = 0$. As was the case for the stochastic algorithm derived in the previous chapter, when $\hat{L} > L$ and $j \to \infty$ the first $(\hat{L} - L) \cdot M$ rows of Z_2 and the last $(\hat{L} - L) \cdot M$ rows of Z_1 will be zero. Finally, for an infinite number of data, the rank of $Z_1 + Z_2$ equals exactly $2 \cdot L$. Inspecting the SVD spectrum computed in equation 10.15 thus provides a way to check whether the order was determined correctly.

Remark 10.3.1 If the LQ decomposition has to be recomputed due to an initial wrong order estimate, the new decomposition does not have to be computed from scratch, but instead the results of the first LQ decomposition can be updated using LQ updating techniques, see e.g. [53].

10.4 Conclusions

In this chapter we have applied the concept of oblique projection to the deterministic blind channel identification problem. Oblique projections allow to decompose a matrix into two non-orthogonal matrices. This property has been used to derive two new deterministic blind channel estimation algorithms. Both algorithms are robust against the spatial color of the additive noise. Furthermore, they are deterministic in the sense that they allow perfect channel recovery with a finite number of samples if the model assumptions hold and if there is no additive noise. The first algorithm is based on two oblique projections and a rank-one matrix approximation. The algorithm is computationally simple and can be used for short data bursts. In the second algorithm, the oblique projections can be considered as a noise filtering preprocessing step, subsequently a subspace type algorithm can be applied to recover the channel parameters. In the next chapter we compare the performance of these two algorithms with the stochastic algorithm developed in chapter 9 and several other algorithms that appeared in literature.

Chapter 11

Performance analysis

This chapter compares the performance of the stochastic algorithm derived in chapter 9, the two deterministic algorithms of chapter 10 and the algorithms [97], [121] and [2] that have recently appeared in literature. Simulation results show that the first deterministic algorithm has a performance similar to that of existing blind channel estimation algorithms [97, 121]. The stochastic and the second deterministic algorithm are designed specifically for situations where the noise is spatially colored. We demonstrate that these algorithms have a better performance than the algorithm of [2] (also robust against the spatial noise color).

In section 11.1, 11.2 and 11.3 we shortly review the algorithms [97], [121] and [2]. Section 11.4 provides a motivation for the simulation scenarios presented in this chapter. It details the similarities and differences between the algorithms of chapters 9 and 10 and the algorithms [97], [121] and [2]. Section 11.5 presents simulation results and finally in section 11.6 some conclusions are drawn.

11.1 Moulines et al. [97]

The algorithm [97] is one of the first subspace based blind channel estimation algorithms that appeared in literature. The following matrix equation is formed for $i \geq L+1$ (assume zero noise),

$$Y_{k|k+i-1} = \mathcal{H}_i \cdot X_{k-L|k+i-1}.$$

It is assumed that $\mathcal{H}_i$ has full column rank for $i \geq L$ and that $X_{k-L|k+i-1}$ has full row rank. Next we compute the SVD of $Y_{k|k+i-1}$:

$$Y_{k|k+i-1} = \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}.$$

It was shown in [97] that the channel parameters can be recovered (up to a scaling factor) from

$$U^{\perp H} \cdot \mathcal{H}_i = 0$$

by exploiting the block Toeplitz structure of $\mathcal{H}_i$.

11.2 Tong and Zhao [121]

Although the original algorithm was presented in a least squares smoothing framework, we will rewrite the algorithm using subspace notation. We only consider the case where the channel order is known. The algorithm begins with the construction of three matrices of output symbols.

$$\begin{aligned} Y_{pa}^{'} &= Y_{1|i} \\ &= \mathcal{H}_i \cdot X_{1-L|i} \\ Y_{pr}^{'} &= Y_{i+1|L+i+1} \\ &= \mathcal{H}_{L+1} \cdot X_{i+1-L|L+i+1} \\ Y_{fu}^{'} &= Y_{i+L+2|L+2\cdot i+1} \\ &= \mathcal{H}_i \cdot X_{i+2|2\cdot i+L+1}. \end{aligned}$$

Note that 'past', 'present' and 'future' outputs are slightly defined differently compared to equation 10.4. The smoothing factor $i \geq L$, such that $\mathcal{H}_i$ has full column rank. The key observation of the algorithm is now that the joint row space of $Y_{pa}^{'}$ and $Y_{fu}^{'}$ comprises the row space of $\begin{bmatrix} X_{i+1-L|i} \\ X_{i+2|i+L+1} \end{bmatrix}$.

The second step of the algorithm consists of the computation of the SVD of the following matrix:

$$\begin{aligned} \begin{bmatrix} Y_{pa}^{'} \\ Y_{fu}^{'} \end{bmatrix} &= U \cdot \Sigma \cdot V^H \\ &= \begin{bmatrix} U^s & U^\perp \end{bmatrix} \cdot \begin{bmatrix} \Sigma^s & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V^{sH} \\ V^{\perp H} \end{bmatrix}. \end{aligned}$$

Projecting the present outputs $Y_{pr}^{'}$ onto $V^{\perp}$ gives:

$$\begin{aligned} Y_{pr}^{'} \cdot V^{\perp} \cdot V^{\perp H} &= \mathcal{H}_{L+1} \cdot \begin{bmatrix} x(i+1-L) & \cdots & x(i-L+j) \\ \vdots & & \vdots \\ x(L+i+1) & \cdots & x(L+i+j) \end{bmatrix} \cdot V^{\perp} \cdot V^{\perp H} \\ &= \mathcal{H}_{L+1} \cdot \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & & \vdots \\ x(i+1) & \cdots & x(i+j) \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \cdot V^{\perp} \cdot V^{\perp H} \\ &= \begin{bmatrix} \mathbf{h_0} \\ \vdots \\ \mathbf{h_L} \end{bmatrix} \cdot \begin{bmatrix} x(i+1) & \cdots & x(i+j) \end{bmatrix} \cdot V^{\perp} \cdot V^{\perp H}. \end{aligned}$$

The above matrix is a rank-one matrix, and an estimate of the channel parameters can be obtained from the left singular vector corresponding to the largest singular value of $Y_{pr}^{'} \cdot V^{\perp} \cdot V^{\perp H}$.

11.3 Abed-Meraim et al. [2]

Here we reformulate the algorithm of [2] using our own notation. The algorithm begins with the construction of the block Hankel matrices $Y_{pa}^{''}$ and $Y_{fu}^{''}$:

$$\begin{aligned} Y_{pa}^{''} &= Y_{1|i} = \begin{bmatrix} \mathbf{y_1} & \cdots & \mathbf{y_j} \\ \vdots & & \vdots \\ \mathbf{y_i} & \cdots & \mathbf{y_{j+i-1}} \end{bmatrix} \\ Y_{fu}^{''} &= Y_{i+1|2\cdot i} = \begin{bmatrix} \mathbf{y_{i+1}} & \cdots & \mathbf{y_{j+i}} \\ \vdots & & \vdots \\ \mathbf{y_{2\cdot i}} & \cdots & \mathbf{y_{j+2\cdot i-1}} \end{bmatrix}. \end{aligned}$$

Note that $Y_{pa}^{''}$ and $Y_{fu}^{''}$ are differently defined compared to Y_{pa} and Y_{fu} of equation 10.4. It is assumed that the channel order L is known and the smoothing factor i is chosen equal to L. Then the matrix R_y is formed as:

$$\begin{aligned} R_y &= Y_{fu}^{''} \cdot Y_{pa}^{''H} + Y_{pa}^{''} \cdot Y_{fu}^{''H} \\ &= \mathcal{H}_i \cdot \underbrace{\begin{bmatrix} 0_{L\times L} & I_{L\times L} \\ I_{L\times L} & 0_{L\times L} \end{bmatrix}}_{R_x} \cdot \mathcal{H}_i^H. \end{aligned} \tag{11.1}$$

The second equation follows from the first equation under the assumptions 9.1.1, 9.1.2 and 9.2.2 stated earlier. Note that the matrix R_x has full rank. The paper [2] shows that if Theorem 9.3.2 holds, the matrix $\mathcal{H}_i$ can be uniquely determined from the null space of R_y.

Remark 11.3.1 Note that when R_y is defined as:

$$R_y = Y_{pr} \cdot Y_{pa}^H + Y_{pr} \cdot Y_{fu}^H$$

a similar algorithm can be derived. This shows the close relationship between the algorithm of [2] and the algorithms presented in chapters 9 and 10.

11.4 Algorithm comparison

In this section, we explain the link between the algorithms developed in chapters 9 and 10 and the algorithms presented in the previous sections. We indicate when each of the algorithms can be applied, and hence motivate the choice of the simulation examples that are presented in section 11.5.

The first deterministic algorithm (algorithm 10.2.1) and the algorithm of [121] both exploit row space information to recover the channel parameters. The algorithm of [121] uses orthogonal projections, whereas algorithm 10.2.1 is based on oblique projections. Algorithm 10.2.1 maintains row space information, unlike the algorithm of [121] that destroys this information. This allows to exploit the Hankel structure of the input matrix (containing the digital symbols) in algorithm 10.2.1, which is not possible in the algorithm of [121]. Furthermore algorithm 10.2.1 allows for a simultaneous channel and symbol recovery, whereas the algorithm of [121] only provides direct channel estimation.

Table 11.1 shows that the algorithm of [97] can be used when the additive noise is spatially white. The algorithm of [121] works well when the additive noise is white and when bursts are short, as will be shown in the simulation results. Deterministic algorithm 1 should preferably be used for short bursts, but is robust against the spatial color of the additive noise.

As the algorithm of [2], the stochastic algorithm of chapter 9 (algorithm 9.3.2) and the second deterministic algorithm (algorithm 10.2.3) are designed for situations where the spatial covariance of the noise is unknown. All three algorithms work for any color of the additive noise, but if the noise is known to be spatially white it is more advantageous to use the algorithm [97]. Therefore we will only test these algorithms in the case of spatially colored noise. Although the stochastic algorithm of chapter 9 was developed for a large data set, simulation results show that even for a relatively small amount of data (100 data samples) the algorithm already has a good performance.

	SD. & WN.	LD. & WN.	SD. & CN.	LD. & CN.
algorithm [97]	xx	xx		
algorithm [121]	xx			
det. alg. 1	xx		xx	
algorithm [2]	x	x	xx	xx
stoch. alg.	x	x	x	xx
det. alg. 2	x	x	xx	xx

Table 11.1: Situations where the different algorithms can be used, SD. = small amount of data, LD. = large amount of data, WN. = white noise, CN. = spatially colored noise, x = the algorithm can be applied, xx = the algorithm is designed for this situation.

Remark 11.4.1 We do not explicitly compare the adaptive direct symbol estimation algorithms of part I with the channel estimation algorithms developed in the previous chapters. The deterministic adaptive algorithms of part I were developed with the concern of computational complexity and are able to track time varying channels. The algorithms provide perfect symbol estimates in the absence of noise as long as model assumptions hold, but the noise statistics were not taken into account in the problem formulation. In part III we focussed on block processing algorithms for channel recovery in the presence of spatially colored noise in a time-invariant environment. Compared to the algorithms of part I, the algorithms require somewhat larger burst lengths and an extra symbol recovery step.

These different boundary conditions make it hard to provide a fair comparison between the two sets of algorithms. Generally speaking, channel estimation can be preferred over direct symbol estimation when channels are well conditioned and bursts are long (the number of channel parameters to be estimated does not increase with the burst length (if the channel is stationary), and hence they can be estimated consistently). However, for shorter bursts and/or ill defined channel lengths direct symbol estimation algorithms may provide better results, see also [133].

11.5 Simulation results

As in the simulations of chapter 5, we again use the channel of length $L = 4$ with $M = 4$ outputs from [97], see equation 5.6. The modulation format is BPSK and the SNR is defined as in equation 5.5. As a performance measure

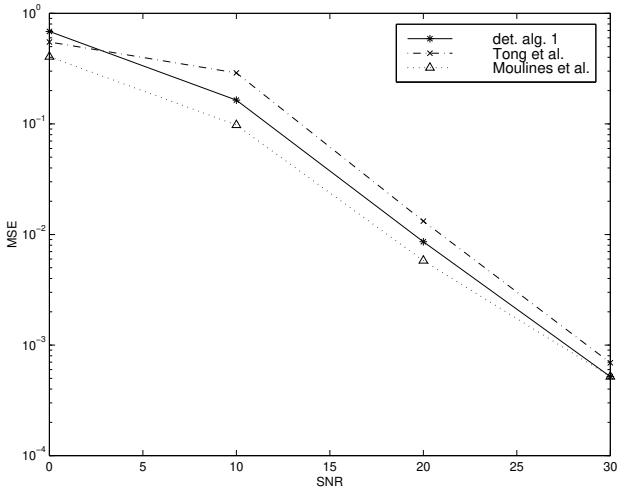


Figure 11.1: MSE for varying SNR (burst length $N = 100$).

we use the (normalized) mean squared error of the channel estimate:

$$MSE = E\{\frac{\|H - \hat{H}\|_F^2}{\|H\|_F^2}\}.$$

Since blind algorithms recover the channel parameters only up to a constant, the parameters $\hat{H}$ estimated by the different algorithms are modified in the following way before the MSE is determined. The constant α is determined as the least squares solution to:

$$\text{vec}(H) = \text{vec}(\hat{H}) \cdot \alpha$$

and $\text{vec}(\hat{H})$ is then modified as:

$$\text{vec}(\hat{H}) \leftarrow \text{vec}(\hat{H}) \cdot \alpha.$$

In all simulations we assume that the channel order is known beforehand.

In a first simulation we compare the performance of deterministic algorithm 1 (algorithm 10.2.1) with that of algorithms [121] and [97]. We choose $i = L$ for algorithm 10.2.1 and $i = L + 1$ for algorithms [121] and [97]. First we study performance as a function of SNR. We take bursts of 100 BPSK symbols and average over 500 runs. Figure 11.1 shows the results. It is seen that algorithm 10.2.1 has a performance between that of algorithms [121] and [97]. As SNR increases the performance of the algorithms converges, since all three algorithms are deterministic. Next, we vary the burst length and keep the SNR fixed at 20 dB. Figure 11.2 depicts the results. For both algorithm 10.2.1 and algorithm

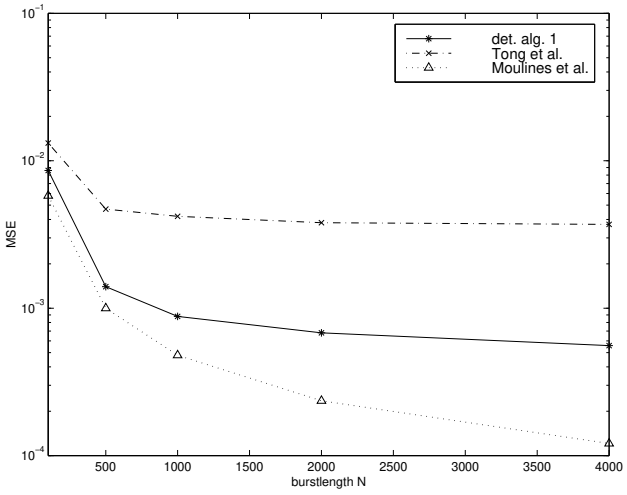


Figure 11.2: MSE for varying burst length (SNR = 20 dB).

[121], the MSE curve flattens out for higher burst lengths. For algorithm 10.2.1 this can intuitively be understood from the discussion in section 10.2.2.

Finally we compare the performance of these three algorithms when the noise is spatially colored. The noise covariance matrix R_n is defined as:

$$\begin{aligned} R_n &= E\{\mathbf{n_k} \cdot \mathbf{n_k}^H\} \\ &= \sigma^2 \cdot \begin{bmatrix} 1 & 0.7 & 0.7^2 & \dots & 0.7^{M-1} \\ 0.7 & 1 & 0.7 & & 0.7^{M-2} \\ \vdots & & \ddots & & \vdots \\ 0.7^{M-1} & & & & 1 \end{bmatrix}. \end{aligned}$$

We take bursts of 100 BPSK symbols, SNR = 20 dB and we average over 500 runs. Figure 11.3 shows the results. Now algorithm 10.2.1 provides the most accurate results, which indicates that the oblique projections make the deterministic algorithm 1 fairly robust against the spatial noise color.

In a second simulation we compare the performance of the stochastic algorithm (algorithm 9.3.2), the second deterministic algorithm (algorithm 10.2.3) and algorithm [2] which are all robust against spatially colored noise. For all algorithms, we took a smoothing factor $i = L$.

Figure 11.4 shows MSE curves as a function of SNR for a burst length of $N = 500$. Deterministic algorithm 2 clearly outperforms algorithm [2] and also slightly improves the results of the stochastic algorithm. Next we vary the burst length and keep the SNR fixed at 20 dB. Figure 11.5 shows the results.

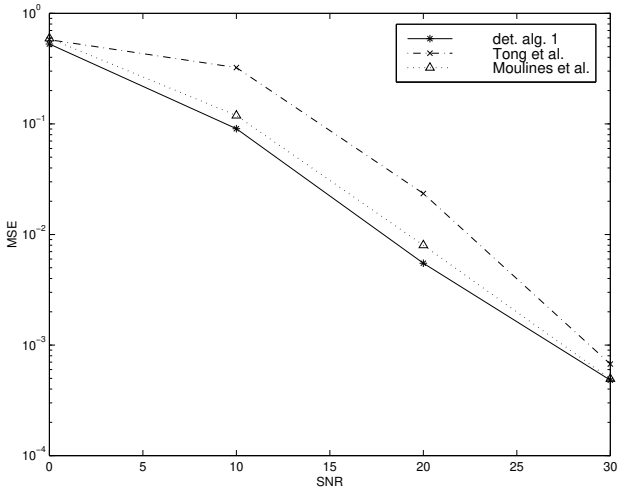


Figure 11.3: MSE for varying SNR (burst length $N = 100$), colored noise.

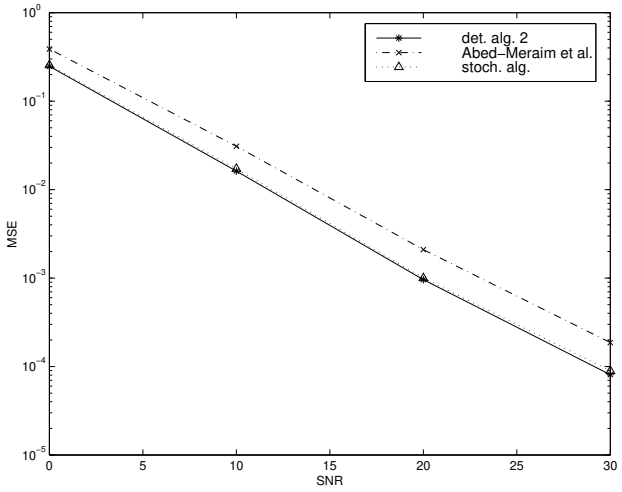


Figure 11.4: MSE for varying SNR (burst length $N = 500$), colored noise.

Deterministic algorithm 2 outperforms both algorithm [2] and the stochastic algorithm. The difference with the stochastic algorithm is especially noticeable for short bursts. For large burst lengths the two algorithms coincide, since for $j \to \infty$, the oblique projections reduce to orthogonal projections.

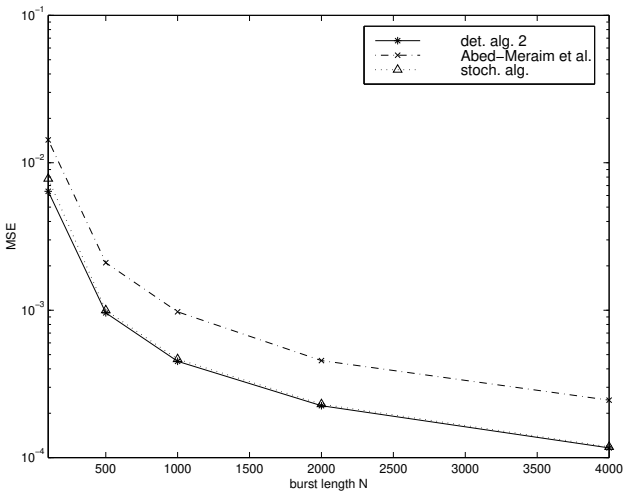


Figure 11.5: MSE for varying burst length (SNR = 20 dB), colored noise.

11.6 Conclusions

In this chapter we have compared the performance of the stochastic and deterministic algorithms derived in chapters 9 and 10. The first deterministic algorithm is well suited for short data bursts and simulation results have shown that its performance is comparable to that of other blind deterministic channel estimation algorithms that have recently appeared in literature [97, 121]. The stochastic subspace algorithm and the second deterministic subspace algorithm are specifically designed for situations where the spatial color of the additive noise is unknown. Simulation results have shown that these new algorithms improve the performance of the subspace algorithm of [2] which is also robust against the spatial color of the additive noise.

Chapter 12

Conclusions and open problems

This last chapter contains two sections. Section 12.1 succinctly reviews the main contributions of this thesis. Section 12.2 gives some suggestions for further research.

12.1 Contributions

In recent years, blind equalization has been an active area of research thanks to its potential applicability in wireless communication systems. Blind algorithms based on second order statistics form a class of spatio-temporal algorithms that may provide both increased capacity and quality of wireless systems.

In this thesis we have developed a number of deterministic blind subspace techniques for symbol recovery, equalizer design and channel estimation. The algorithms share the common feature of being very data efficient, i.e. they provide excellent results with a modest number of samples as long as model assumptions hold. We considered both single and multi-user blind equalization. Two major themes recurred throughout this thesis. First we searched for *low complexity algorithms*. In the first part we have developed adaptive single user algorithms. In the second part we have shown how coding at the transmitter allows to develop low complexity decoding algorithms at the receiver. The second theme is the *robustness* of subspace algorithms. In the second part of this thesis, we have shown that the use of coding avoids the need to identify the exact channel length of each user. We have also demonstrated how coding improves robustness against rank detection problems. Further in the third part

we have developed subspace techniques robust against the spatial color of the noise. In the next paragraphs we summarize the main contributions of each part.

In part one we have presented adaptive subspace algorithms for direct blind symbol estimation that are able to reduce the computational requirements of comparable subspace algorithms that are based on block processing. We have shown how subspace techniques are used to construct a set of homogeneous equations in the unknown transmitted symbols such that adaptive processing is facilitated. Applying different non-triviality constraints has led to three different algorithms having different complexity and performance. We have considered a finite alphabet constraint, a monic constraint and a quadratic non-triviality constraint resulting in respectively a Viterbi algorithm a Kalman filter algorithm and a Recursive Total Least Squares (RTLS) solution. Both the Viterbi and Kalman filter algorithm are standard building blocks that can almost be straightforwardly applied to the problem at hand. For the recursive total least squares problem, we developed two new algorithms that explicitly exploit the structure of the set of homogeneous equations and combine it with a specific decision feedback mechanism to obtain algorithms with constant computational and memory requirements in each iteration step. For one of the RTLS algorithms we have further shown how it can be modified to solve deconvolution problems involving a non-homogeneous set of equations. In part one we have also analyzed complexity and performance of the algorithms and we have compared them to a similar non-adaptive blind symbol estimation algorithm [80]. Simulation results have shown that the Viterbi algorithm has both the best performance and the highest complexity. The complexity of the RTLS algorithms and the Kalman filter solutions are approximately equal and are lower than that of the block processing algorithm [80] (a factor 5 to 20 for burst lengths between 100 en 200 symbol periods). However, the RTLS algorithms have a better performance than the Kalman filter solution. We also demonstrated that the adaptive algorithms are able to track time varying channels. We have shown that the RTLS algorithm for non-homogeneous equations closely approximates the performance of the standard TLS algorithm at sufficiently high SNR at this at a strongly reduced computational complexity. Finally we succinctly discussed a DSP implementation of one of the RTLS algorithms.

In part two we have developed multi-user blind equalization algorithms. We have shown that the use of coding at the transmitter side allows simple decoding algorithms. Instead of solving a (large) d-dimensional problem (for d users), as is done usually, the coding allows to solve d one-dimensional problems in parallel. We have considered finite alphabet and quadratic non-triviality constraints. We have shown how direct symbol estimation and equalizer design can proceed. We have demonstrated how the algorithms developed in part one are almost extended in a trivial way for the multi-user case. Simulation

results have demonstrated that, in a blind signal separation context, performance of the new algorithms is comparable to that of existing approaches but the new algorithms do not suffer from initialization or convergence problems. Perhaps even more important, we investigated the robustness provided by coding methods. Subspace algorithms are known to suffer from order detection problems. These can be alleviated by the use of coding. Simulation results indicated that accurate symbol estimation is possible, even if the system order is largely overestimated. Further we have indicated how the symbol sequences of different users can be identified without requiring the knowledge of their individual channel lengths. The estimation of these channel lengths usually involves a cumbersome iterative procedure, see e.g. [81]. In the new procedure, only a lower bound on the channel lengths needs to be known. Finally, using a simulation example we have demonstrated that the potential of multi-user algorithms with respect to ideal single user algorithms.

In part three, we returned to the single user scenario. However, now we aimed at channel estimation rather than direct symbol estimation. Further we gave special attention to the robustness of the algorithms against spatially colored noise. We have developed one stochastic and two deterministic algorithms based on respectively orthogonal and oblique projections. The projections can be considered as a noise filtering preprocessing step. The first deterministic algorithm is based on two oblique projections (one LQ-decomposition) and a rank-one matrix approximation and its performance was shown to be comparable to that of existing channel estimation algorithms [97, 121]. The stochastic algorithm and the second deterministic algorithm are similar and are based on orthogonal and oblique projections respectively, combined with a subspace based channel recovery step. Their performance was shown to be superior to that of the algorithm of [2] which is also robust against the spatial noise color.

12.2 Suggestions for further research

Blind equalization based on second order statistics is a new domain that has been extensively explored in recent years. This new class of space-time algorithms has the potential of improving performance of both space only and time only processing. However, current research efforts have mainly been focussed on theoretical studies coupled to computer simulation results. Evaluations on real life data are rare [54]. The DSP setup described in section 5.5 is still in a development phase, but forms a first step towards the integration of blind equalization algorithms into a complete modem structure. The addition of an antenna front-end will allow more realistic simulation scenarios in the future.

In this thesis, we have considered only systems based on TDMA and FDMA (like the GSM system). However, third generation wireless systems will most

probably be based on wideband CDMA [24, 100]. In a CDMA system, all users occupy the same time and frequency band but employ different orthogonal codes. Similar blind equalization techniques as for TDMA/FDMA based systems can be developed [79, 84, 75, 76].

Current standardization efforts [24] for third generation wireless systems foresee user-dependent pilot signals for channel estimation. These pave the way for adaptive antenna employment. In a blind equalization context, relatively little attention has been paid to semi-blind algorithms. These algorithms combine elements of training based and blind methods and can efficiently exploit pilot signals. Hence, in the perspective of the standardization process it is meaningful to further investigate semi-blind techniques.

Another interesting research topic is the application of spatial multiplexing techniques in a wireless local area network (WLAN) environment, where channel characteristics are only slowly time varying compared to the outdoor mobile communications environment. In this context SDMA techniques have been successfully combined with orthogonal frequency division multiplexing (OFDM) techniques [153, 154].

Bibliography

[1] K. Abed-Meraim, J.-F. Cardoso, A.Y. Gorokhov, P. Loubaton, and E. Moulines, "On Subspace Methods for Blind Identification of Single-Input Multiple-Output FIR Systems," *IEEE Transactions on Signal Processing*, vol. 45, no. 1, pp. 42–55, Jan. 1997.

[2] K. Abed-Meraim, Y. Hua, P. Loubaton, and E. Moulines, "Subspace Method for Blind Identification of Multichannel FIR Systems in Noise Fields with Unknown Spatial Covariance," *IEEE Signal Processing Letters*, vol. 4, no. 5, pp. 135–137, May 1997.

[3] K. Abed-Meraim, P. Loubaton, and E. Moulines, "A Subspace Algorithm for Certain Blind Identification Problems," *IEEE Transactions on Information Theory*, vol. 43, no. 2, pp. 499–511, Mar. 1997.

[4] K. Abed-Meraim, E. Moulines, and P. Loubaton, "Prediction Error Method for Second-Order Blind Identification," *IEEE Transactions on Signal Processing*, vol. 45, no. 3, pp. 694–705, Mar. 1997.

[5] H. Akaike, "A New Look at the Statistical Model Identification," *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, Dec. 1974.

[6] K. Anand, G. Mathew, and V.U. Reddy, "Blind Separation of Multiple Co-Channel BPSK Signals Arriving at an Antenna Array," *IEEE Signal Processing Letters*, vol. 2, no. 9, pp. 176–178, Sept. 1995.

[7] S. Anderson, U. Forssen, and J. Karlsson, "Ericsson/Mannesmann GSM Field-Trials with Adaptive Antennas," in *Proceedings of the 1996 IEE Colloquim on Advanced TDMA Techniques and Applications, London, U.K.*, 1996.

[8] S. Andersson, M. Milnert, M. Viberg, and B. Wahlberg, "An Adaptive Array for Mobile Communication Systems," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, pp. 230–236, 1991.

[9] P. Balaban and J. Salz, "Optimum Diversity Combining and Equalization in Digital Data Transmission with Applications to Cellular Mobile Radio - Part I : Theoretical Considerations," *IEEE Transactions on Communications*, vol. 40, no. 5, pp. 885–894, May 1992.

[10] J. Barlow, P. Yoon, and H. Zha, "An Algorithm and a Stability Theory for Downdating the ULV Decomposition," *BIT*, vol. 36, pp. 14–40, Jan. 1996.

[11] L. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occuring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *Ann. Math. Stat.*, vol. 41, pp. 164–171, 1970.

[12] R.T. Behrens and L. L. Scharf, "Signal Processing Applications of Oblique Projection Operators," *IEEE Transactions on Signal Processing*, vol. 42, no. 6, pp. 1413–1424, 1994.

[13] D. Borah, R. Kennedy, and I. Fijalkow, "Effects of Colored Noise on the Performance of Linear Equalizers," in *Proceedings ICASSP*, pp. 1497–1500, Mar. 1999.

[14] D. Boss, K.-D. Kammeyer, and T. Petermann, "Is Blind Estimation Feasible in Mobile Communication Systems? A Study Based on GSM," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1479–1492, Oct. 1998.

[15] J. Bunch and P. Nielsen, "Updating the Singular Value Decomposition," *Numer. Math.*, vol. 31, pp. 111–129, 1978.

[16] P. Bunch, "Updating a Singular Value Decomposition," *BIT*, vol. 10, pp. 376–396, 1970.

[17] M. Cedervall, B. Ng, and A. Paulraj, "Structured Methods for Blind Multi-Channel Identification," in *Proceedings of DSP 97*, pp. 387–390, 1997.

[18] T. Chan, "An Improved Algorithm for Computing the Singular Value Decomposition," *ACM Trans. Math. Software*, vol. 8, pp. 72–83, 1982.

[19] A. Chevreuil and P. Loubaton, "Blind Second-Order Identification of FIR Channels: Forced Cyclostationarity and Structured Subspace Method," *IEEE Signal Processing Letters*, vol. 4, no. 7, pp. 204–206, July 1997.

[20] Y.M. Cho, G. Xu, and T. Kailath, "Fast Recursive Identification of State Space Models via Exploitation of Displacement Structure," *Automatica, Special Issue on Statistical Signal Processing and Control*, vol. 30, pp. 45–59, 1994.

[21] H. Cirpan and M. Tsatsanis, "Stochastic Maximum Likelihood Methods for Semi-Blind Channel Estimation," *IEEE Signal Processing Letters*, vol. 5, no. 1, pp. 21–24, Jan. 1998.

[22] D. Cox, "910 MHz Urban Mobile Radio Propagation: Multipath Characteristics in New York City," *IEEE Transactions on Communications*, vol. 21, no. 11, pp. 1188–1194, Nov. 1973.

[23] R. Cupo, G. Golden, C. Martin, K. Sherman, N. Sollenberger, J. Winters, and P. Wolniansky, "A Four Element Adaptive Array for IS-136 PCS base stations," in *Proceedings of the 47th IEEE Vehicular Technology conference, Phoenix, AZ, USA*, 1997.

[24] E. Dahlman, B. Gudmundson, M. Nilsson, and J. Sköld, "UMTS/IMT-2000 Based on Wideband CDMA," *IEEE Communications Magazine*, pp. 70–80, Sept. 1998.

[25] B. De Moor, "Structured Total Least Squares and L_2 Approximation Problems," *Linear Algebra and its Applications*, vol. 188-189, no. 11, pp. 163–207, July 1993.

[26] B. De Moor, "Total Least Squares for Affinely Structured Matrices and the Noisy Realization Problem," *IEEE Transactions on Signal Processing*, vol. 42, no. 11, pp. 3104–3113, 1994.

[27] J.R. Deller, Jr., J.G. Proakis, and J.H.L. Hansen, *Discrete-Time Processing of Speech Signals*. Englewood Cliffs: Macmillan Publishing Company, 1993.

[28] S. Dieusaert and J. De Lafonteyne, "Implementatie op een Hardware Demonstrator van Blinde Egalisatie-Algoritmen voor Mobiele Communicatie," Master's thesis, Katholieke Universiteit Leuven, Faculteit Toegepaste Wetenschappen Departement Elektrotechniek, 1998.

[29] Z. Ding, "Characteristics of Band-Limited Channels Unidentifiable from Second-Order Cyclostationary Statistics," *IEEE Signal Processing Letters*, vol. 3, no. 5, pp. 150–152, May 1996.

[30] Z. Ding, "Multipath Channel Identification Based on Partial System Information," *IEEE Transactions on Signal Processing*, vol. 45, no. 1, pp. 235–240, Jan. 1997.

[31] Z. Ding, R.A. Kennedy, B.D.O. Anderson, and C. R. Johnson, Jr., "Ill-Convergence of Godard Blind Equalizers in Data Communication Systems," *IEEE Transactions on Communications*, vol. 39, no. 9, pp. 1313–1327, Sept. 1991.

[32] Z. Ding and G. Li, "Single-Channel Blind Equalization for GSM Cellular Systems," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1493–1505, Oct. 1998.

[33] J.J. Dongarra, J.R. Bunch, C.B. Moler, and G.W. Stewart, *LINPACK Users' Guide*. SIAM, 1979.

[34] D.B. Ducan and S.D. Horn, "Linear Dynamic Recursive Estimation from the Viewpoint of Regression Analysis," *J. Amer. Statist. Assoc.*, vol. 67, pp. 815–821, 1972.

[35] T.J. Endres, D.O. Anderson, C. R. Johnson, Jr., and L. Tong, "On the Robustness of FIR Channel Identification from Fractionally Spaced Received Signal Second-Order Statistics," *IEEE Signal Processing Letters*, vol. 3, no. 5, pp. 153–155, May 1996.

[36] V.M. Eyuboğlu and S.U. Qureshi, "Reduced-State Sequence Estimation with Set Partitioning and Decision Feedback," *IEEE Transactions on Communications*, vol. 36, pp. 13–20, Jan. 1988.

[37] D.D. Falconer and F.R. Magee, Jr., "Adaptive Channel Memory Truncation for Maximum Likelihood Sequence Estimation," *The Bell system technical journal*, vol. 52, pp. 1541–1562, Nov. 1973.

[38] G.D. Forney, Jr., "Maximum-Likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference," *IEEE Transactions on Information Theory*, vol. 18, no. 3, pp. 363–378, May 1972.

[39] G.D. Forney, Jr., "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[40] D. Gerlach and A. Paulraj, "Adaptive Transmitting Antenna Arrays With Feedback," *IEEE Signal Processing Letters*, vol. 1, no. 10, pp. 150–152, Oct. 1994.

[41] D. Gesbert and P. Duhamel, "Robust Blind Channel Identification and Equalization Based on Multi-Step Predictors," in *Proceedings ICASSP*, pp. 3621–3624, Apr. 1997.

[42] D. Gesbert, P. Duhamel, and S. Mayrargue, "On-Line Blind Multichannel Equalization Based on Mutually Referenced Filters," *IEEE Transactions on Signal Processing*, vol. 45, no. 9, pp. 2307–2317, Sept. 1997.

[43] M. Ghosh, "Blind Decision Feedback Equalization for Terrestrial Television Receivers," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2070–2081, Oct. 1998.

[44] M. Ghosh and C. Weber, "Maximum-Likelihood Blind Equalization," *Optical Engineering*, vol. 31, no. 6, pp. 1224–1228, June 1992.

[45] G. Giannakis, "Filterbanks for Blind Channel Identification and Equalization," *IEEE Signal Processing Letters*, vol. 4, no. 6, pp. 184–187, 1997.

[46] G. Giannakis and S. Halford, "Asymptotically Optimal Blind Fractionally-Spaced Channel Estimation and Performance Analysis," *IEEE Transactions on Signal Processing*, vol. 45, no. 7, pp. 1815–1830, 1997.

[47] G. Giannakis and S. Halford, "Blind Fractionally Spaced Equalization of Noisy FIR Channels: Direct and Adaptive Solutions," *IEEE Transactions on Signal Processing*, vol. 45, no. 9, pp. 2277–2292, 1997.

[48] G. Giannakis and J. Mendel, "Identification of Non-Minimum Phase Systems Using Higher-Order Statistics," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. 37, pp. 360–377, 1989.

[49] G. Giannakis and E. Serpedin, "Blind Channel Identification with Modulation Induced Cyclostationarity," in *DSP*, 1997.

[50] L. Godara, "Applications of Antenna Arrays to Mobile Communications, Part I: Performance Improvement, Feasibility and System Considerations," *Proceedings of the IEEE*, vol. 85, no. 7, pp. 1031–1060, July 1997.

[51] L. Godara, "Applications of Antenna Arrays to Mobile Communications, Part II: Beam-Forming and Direction-of-Arrival Considerations," *Proceedings of the IEEE*, vol. 85, no. 7, pp. 1195–1245, July 1997.

[52] D.N. Godard, "Self-Recovering Equalization and Carrier Tracking in Two-Dimensional Data Communication Systems," *IEEE Transactions on Communications*, vol. 28, pp. 1867–1875, Nov. 1980.

[53] G. H. Golub and C. F. Van Loan, *Matrix Computations*. The Johns Hopkins University Press, 1989.

[54] A. Gorokhov, M. Kristensson, B. Ottersten, and M. Youssefmir, "Some Results on Blind Deconvolution Applied to Digital Communication Signals," in *Proceedings of DSP 97*, 1997.

[55] J. Gunther and A. Swindlehurst, "Algorithms for Blind Equalization with Multiple Antennas Based on Frequency Domain Subspaces," in *Proceedings ICASSP*, pp. 2421–2424, 1996.

[56] J. Gunther and A. Swindlehurst, "Blind Sequential Symbol Estimation of Co-Channel Finite Alphabet Signals," in *Proceedings Asilomar*, 1996.

[57] F. Gustafsson and B. Wahlberg, "Blind Equalization by Direct Examination of the Input Sequences," *IEEE Transactions on Communications*, vol. 43, no. 7, pp. 2213–2222, July 1995.

[58] B. Halder, B. Ng, A. Paulraj, and T. Kailath, "Unconditional Maximum Likelihood Approach for Blind Estimation of Digital Signals," in *Proceedings of ICASSP*, pp. 1081–1084, 1996.

[59] S. Haykin, *Adaptive filter theory, 2nd ed.* Prentice-Hall Inc., Englewood Cliffs, N.J., 1991.

[60] S. Haykin, ed., *Blind Deconvolution.* Prentice Hall, 1994.

[61] R.W. Heath, Jr., S.D. Halford, and G.B. Giannakis, "Adaptive Blind Channel Identification of FIR Channels for Viterbi Decoding," in *Proceedings Asilomar*, 1996.

[62] Y. Hua, "Fast Maximum Likelihood for Blind Identification of Multiple FIR Channels," *IEEE Transactions on Signal Processing*, vol. 44, pp. 661–672, Mar. 1996.

[63] C. R. Johnson, Jr., "Admissibility in Blind Adaptive Channel Equalization," *IEEE Control Systems*, no. 1, pp. 3–15, 1991.

[64] T. Kailath, *Linear Systems.* Prentice Hall, 1980.

[65] R.E. Kalman, "A New Approach to Linear Filtering and Prediction," *Trans. ASME J. Basic Eng.*, vol. 82D, pp. 34–45, 1960.

[66] R.E. Kamel and Y. Bar-Ness, "Blind Maximum Likelihood Sequence Estimation of Digital Sequences in Presence of Intersymbol Interference," *Electronic Letters*, vol. 30, no. 7, pp. 537–539, Mar. 1994.

[67] H. Krim and M. Viberg, "Two Decades of Array Signal Processing Research," *IEEE Signal Processing Magazine*, pp. 67–94, July 1996.

[68] M. Kristensson and B. Ottersten, "A Statistical Approach to Subspace Based Blind Identification," *IEEE Transactions on Signal Processing*, vol. 46, no. 6, pp. 1612–1623, June 1998.

[69] M. Kristensson, B. Ottersten, and D. Slock, "Blind Subspace Identification of a BPSK Communication Channel," in *Proceedings Asilomar*, 1996.

[70] P.A. Laurent, "Exact and Approximate Construction of Digital Phase Modulations by Superposition of Amplitude Modulated Pulses (AMP)," *IEEE Transactions on Communications*, vol. 34, pp. 150–162, Feb. 1986.

[71] R.E. Lawrence and H. Kaufman, "The Kalman Filter for the Equalization of a Digital Communications Channel," *IEEE Transactions on Communications*, vol. 19, no. 6, pp. 1137–1141, Dec. 1971.

[72] R.E. Lawrence and H. Kaufman, "Kalman Filter Equalization for QPSK Communication," *IEEE Transactions on Communications*, pp. 361–364, Mar. 1976.

[73] P. Lemmerling, *Structured Total Least Squares: Analysis, Algorithms and Applications.* PhD thesis, Dept. of Elect. Eng., ESAT-SISTA, Katholieke Universiteit Leuven, May 1999.

[74] P. Lemmerling, S. Van Huffel, and B. De Moor, "Structured Total Least Squares Problems: Formulations, Algorithms and Applications," in *Proceedings of 2nd International Workshop on TLS and Errors in Variables Modelling*, pp. 215–223, SIAM, Philadelphia, 1997.

[75] G. Leus and M. Moonen, "An Adaptive Blind Receiver for Asynchronous DS-CDMA Based on Recursive SVD and Viterbi Decoding," in *Proceedings of the IEEE Symposium on Communications and Vehicular technology, Enschede, The Netherlands*, pp. 40–47, Oct. 1997.

[76] G. Leus and M. Moonen, "An Adaptive Blind Receiver for Asynchronous DS-CDMA Based on Recursive SVD and RLS," in *Proceedings of the ProRISC/IEEE Workshop on Circuits, Systems and Signal Processing, Mierlo, The Netherlands*, pp. 615–620, Nov. 1997.

[77] F. Li and R.J. Vaccaro, "Performance Analysis for the State Space Realization (TAM) and ESPRIT Algorithms for DOA Estimation," *IEEE Trans. Antennas and Propagation*, vol. 39, pp. 418–423, Mar. 1991.

[78] Y. Li and Z. Ding, "Blind Channel Identification Based on Second Order Cyclostationary Statistics," in *Proceedings ICASSP*, vol. 4, pp. 81–84, 1993.

[79] G. Liu, H. Xu, "A Subspace Method for Signature Waveform Estimation in Synchronous CDMA Systems," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1346–1354, Oct. 1996.

[80] H. Liu and G. Xu, "A Deterministic Approach to Blind Symbol Estimation," *IEEE Signal Processing Letters*, vol. 1, no. 12, pp. 205–207, Dec. 1994.

[81] H. Liu and G. Xu, "Smart Antennas in Wireless Systems: Uplink Multiuser Blind Channel and Sequence Detection," *IEEE Transactions on Communications*, vol. 45, no. 2, pp. 187–199, Feb. 1997.

[82] H. Liu, G. Xu, L. Tong, and T. Kailath, "Recent Developments in Blind Channel Equalization: From Cyclostationarity to Subspaces," *Signal Processing*, vol. 50, pp. 83–99, 1996.

[83] H. Liu and H. Xu, "Multiuser Blind Channel Estimation and Spatial Channel Pre-Equalization," in *Proceedings ICASSP*, pp. 1756–1759, 1995.

[84] H. Liu and M. Zoltowski, "Blind Equalization in Antenna Array CDMA Systems," *IEEE Transactions on Signal Processing*, vol. 45, pp. 161–172, Jan. 1997.

[85] L. Ljung, *System Identification: Theory for the User*. Prentice Hall, 1987.

[86] 3L Ltd., *C4x Parallel C Version 2.04*. 3L Ltd., 1995.

[87] H. Luo and Y. Li, "Application of Blind Channel Identification Techniques to Prestack Seismic Deconvolution," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2082–2089, Oct. 1998.

[88] A. Mansour, C. Jutten, and P. Loubaton, "Subspace Method for Blind Separation of Sources in Convolutive Mixture," in *Proceedings of Eusipco*, Mar. 1996.

[89] S. Mayrargue, "Spatial Equalization of a Radio-Mobile Channel without Beamforming using the Constant Modulus Algorithm (CMA)," in *Proceedings ICASSP*, vol. 3, pp. 344–347, Apr. 1993.

[90] J. Mendel, "Tutorial on Higher-Order Statistics (Spectra) in Signal Processing and System Theory: Theoretical Results and Some Applications," *Proc. IEEE*, vol. 79, pp. 278–305, 1991.

[91] J.M. Mendel, *Lessons in Estimation Theory for Signal Processing, Communications, and Control*. Prentice Hall Signal Processing Series, 1995.

[92] N. Metzner, F. David, and H. Eul, "A Real-Time Testbed for a Third Generation Mobile Communication System," in *Proceedings of the 44th IEEE Vehicular Technology conference, Stockholm, Sweden*, 1994.

[93] P Mogensen, F. Frederiksen, H. Dam, K. Olesen, and L. Larsen, "A Hardware Testbed for Evaluation of Adaptive Antennas in GSM/UMTS," in *Proceedings of the 7th IEEE PIMRC conference, Taipei, Taiwan*, 1996.

[94] M. Moonen, "Systolic Algorithms for Recursive Total Least Squares Parameter Estimation and Mixed RLS/RTLS Problems," *International Journal of High Speed Electronics*, vol. 4, no. 1, pp. 55–68, 1993.

[95] M. Moonen, E. Deprettere, I.K. Proudler, and J.G. McWhirter, "On the Derivation of Parallel Filter Structures for Adaptive Eigenvalue and Singular Value Decompositions," in *Proceedings ICASSP-95*, pp. 3247–3250, May 1995.

[96] M. Moonen and J.G. McWhirter, " A Systolic Array for Recursive Least Squares by Inverse Updating," *Electronics Letters*, vol. 29, no. 13, pp. 1217–1218, 1993.

[97] E. Moulines, P. Duhamel, J. Cardoso, and S. Mayrargue, "Subspace Methods for the Blind Identification of Multichannel FIR Filters," *IEEE Transactions on Signal Processing*, vol. 43, no. 2, pp. 516–525, Feb. 1995.

[98] M. Mouly and M.-B. Pautet, *The GSM System for Mobile Communications*. Palaiseau, France, 1992.

[99] B. Ng, D. Gesbert, and A. Paulraj, "A Semi-Blind Approach to Structured Channel Equalization," in *Proceedings of ICASSP*, pp. 3385–3388, May 1998.

[100] T. Ojanperä and R. Prasad, "An Overview of Air Interface Multiple Access for IMT-2000/UMTS," *IEEE Communications Magazine*, pp. 82–95, Sept. 1998.

[101] C.C. Paige and M. Saunders, "Least Squares Estimation of Discrete Linear Dynamic Systems Using Orthogonal Transformations," *SIAM J. Numer. Anal.*, vol. 14, no. 2, pp. 180–193, 1977.

[102] C. Pan and R. Plemmons, "Least Squares Modifications with Inverse Factorization: Parallel Implications.," *J. Computational and Applied Mathematics*, vol. 27, no. 1-2, pp. 109–127, 1989.

[103] A. H. Paulraj and C.B. Papadias, "Space-Time Processing for Wireless Communications," *IEEE Signal Processing Magazine*, pp. 49–83, Nov. 1997.

[104] J.G. Proakis, *Digital Communications*. Mc. Graw-Hill, 2 ed., 1989.

[105] V. Radionov and S. Mayrargue, "Semi-Blind Approach to Second Order Identification of SIMO-FIR Channel Driven by Finite-Alphabet Sequence," in *Proceedings of DSP 97*, pp. 115–118, 1997.

[106] V.U. Reddy, C.B. Papadias, and A.J. Paulraj, "Blind Identifiability of Certain Classes of Multipath Channels from Second-Order Statistics Using Antenna Arrays," *IEEE Signal Processing Letters*, vol. 4, no. 5, pp. 138–141, May 1997.

[107] J. Rissanen, "Modeling by Shortest Data Description," *Automatica*, vol. 14, pp. 465–471, 1978.

[108] H. Sampath and A. Paulraj, "Space-Time Processing TDMA Wireless Testbed," in *Proceedings of ICASSP 99, Phoenix AZ, USA*, 1999.

[109] Y. Sato, "A Method of Self-Recovering Equalization for Multilevel Amplitude-Modulation Systems," *IEEE Transactions on Communications*, vol. 23, pp. 679–682, June 1975.

[110] L. Sharf, "The SVD and Reduced Rank Signal Processing," *Signal Processing*, vol. 25, pp. 113–133, 1991.

[111] D. Slock, "Blind Fractionally-Spaced Equalization, Perfect Reconstruction Filterbanks, and Multilinear Prediction," in *Proceedings ICASSP, Adelaide, Australia*, 1994.

[112] D.T.M. Slock and C.B. Papadias, "Blind Fractionally-Spaced Equalization Based on Cyclostationarity," in *Proc. Vehicular Technology Conf., Stockholm, Sweden*, June 1994.

[113] G.W. Stewart, "Error and Perturbation Bounds for Subspaces Associated with Certain Eigenvalue Problems," *SIAM Review*, vol. 15, pp. 727–764, Oct. 1973.

[114] G.W. Stewart, "Updating a Rank-Revealing ULV Decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 14, no. 2, pp. 494–499, 1993.

[115] S. Talwar, M. Viberg, and A. Paulraj, "Blind Separation of Synchronous Co-Channel Digital Signals Using an Antenna Array - Part I: Algorithms," *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1184–1197, May 1996.

[116] L. Tong, "Blind Sequence Estimation," *IEEE Transactions on Communications*, vol. 43, pp. 2986–2994, Dec. 1995.

[117] L. Tong and S. Perreau, "Multichannel Blind Identification: From Subspace to Maximum Likelihood Methods," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 1951–1968, Oct. 1998.

[118] L. Tong, G. Xu, B. Hassibi, and T. Kailath, "Blind Channel Identification Based on Second-Order Statistics: A Frequency-Domain Approach," *IEEE Transactions on Information Theory*, vol. 41, no. 1, pp. 329–334, Jan. 1995.

[119] L. Tong, G. Xu, and T. Kailath, "A New Approach to Blind Identification and Equalization of Multipath Channels," in *Proc. of the 25th Asilomar Conference on Signals, Systems & Computers, Pacific Grove, CA*, pp. 856–860, Nov. 1991.

[120] L. Tong, G. Xu, and T. Kailath, "Fast Blind Equalization via Antenna Arrays," in *Proceedings ICASSP*, vol. 4, pp. 272–275, 1993.

[121] L. Tong and Q. Zhao, "Blind Channel Estimation by Least Squares Smoothing," in *Proceedings ICASSP*, vol. 4, pp. 2121–2124, May 1998.

[122] J.R. Treichler and B.G. Agee, "A New Approach to Multipath Correction of Constant Modulus Signals," *IEEE Transactions on ASSP*, vol. 31, no. 2, Apr. 1983.

[123] J.R. Treichler, M.G. Larimore, and J.C. Harp, "Practical Blind Demodulators for High-Order QAM Signals," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 1907–1926, Oct. 1998.

[124] M. Tsatsanis and G. Giannakis, "Transmitter Induced Cyclostationarity for Blind Channel Equalization," *IEEE Transactions on Signal Processing*, vol. 45, no. 7, pp. 1785–1794, July 1997.

[125] J.K. Tugnait, "On Blind Identifiability of Multipath Channels Using Fractional Sampling and Second Order Cyclostationary Statistics," *IEEE Trans. Inform. Theory*, vol. 41, pp. 308–311, Jan. 1995.

[126] G. Ungerboeck, "Fractional Tap-Spacing Equalizer and Consequences for Clock Recovery in Data-Modems," *IEEE Transactions on Communications*, vol. COM-24, no. 8, pp. 856–864, Aug. 1976.

[127] A.-J. van der Veen, "Resolution Limits of Blind Multi-User Multi-Channel Identification Schemes - the Bandlimited Case," in *Proceedings ICASSP*, pp. 2722–2725, May 1996.

[128] A.-J. van der Veen, "Analytical Method for Blind Binary Signal Separation," *IEEE Transactions on Signal Processing*, vol. 45, no. 4, pp. 1078–1082, Apr. 1997.

[129] A.-J. van der Veen, E. Deprettere, and A. Swindlehurst, "Subspace-Based Signal Analysis Using Singular Value Decomposition," *IEEE Proceedings*, vol. 81, no. 9, pp. 1277–1308, Sept. 1993.

[130] A.-J. van der Veen and A. Paulraj, "An Analytical Constant Modulus Algorithm," *IEEE Transactions on Signal Processing*, vol. 44, pp. 1136–1155, May 1996.

[131] A.-J. van der Veen and A. Paulraj, "Singular Value Analysis of Space-Time Equalization in the GSM Mobile System," in *Proceedings ICASSP*, pp. 1073–1076, May 1996.

[132] A.-J. van der Veen, S. Talwar, and A. Paulraj, "Blind Estimation of Multiple Digital Signals Transmitted over FIR Channels," *IEEE Signal Processing Letters*, vol. 2, no. 5, pp. 99–102, May 1995.

[133] A.-J. van der Veen, S. Talwar, and A. Paulraj, "A Subspace Approach to Blind Space-Time Signal Processing for Wireless Communication Systems," *IEEE Transactions on Signal Processing*, vol. 45, no. 1, pp. 173–190, Jan. 1997.

[134] S. Van Huffel and J. Vandewalle, *The Total Least Squares Problem, Computational Aspects and Analysis.* SIAM, 1991.

[135] P. Van Overschee and B. De Moor, *Subspace Identification for Linear Systems: Theory, Implementation, Applications.* Dordrecht: Kluwer Academic Publishers, 1996.

[136] P. Vandaele, G. Leus, and M. Moonen, "A Non-Iterative Blind Binary Signal Separation Algorithm Based on Linear Coding," in *Proceedings of the 2nd IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Annapolis, MD, USA*, pp. 98–101, 1999.

[137] P. Vandaele, G. Leus, and M. Moonen, "A Non-Iterative Blind Signal Separation Algorithm Based on Transmit Diversity and Coding," Tech. Rep. 99-50, K.U.Leuven ESAT-SISTA, 1999. Accepted for publication, ASILOMAR '99.

[138] P. Vandaele and M. Moonen, "An Adaptive Blind Multi-user Detection Algorithm for Mobile Radio Communication," in *Proceedings of the Fourth IEEE Symposium on Communications and Vehicular Technology in the Benelux, Gent, Belgium*, pp. 73–80, Oct. 1996.

[139] P. Vandaele and M. Moonen, "An 'SVD + Viterbi' Algorithm for Adaptive Blind Equalization of Mobile Radio Channels," in *Proceedings of the Fourth Bayona (Pre Cost #254) Workshop on Intelligent Methods for Signal Processing and Communications, Bayona, Vigo*, pp. 159–163, June 1996.

[140] P. Vandaele and M. Moonen, "A Recursive Total Least Squares Algorithm for Single User Blind Channel Equalization," Tech. Rep. 97-04, K.U.Leuven ESAT-SISTA, 1997. Accepted for publication in *Proceedings of the COST#254 workshop on intelligent processing and its practical applications, Budapest, Hungary*, 1997.

[141] P. Vandaele and M. Moonen, "Antenna Array Based Adaptive Algorithms for Blind Symbol Estimation," in *Proceedings of the COST 260 Workshop on Smart Antenna Design and Technology, Dubrovnik, Croatia*, pp. 2.51–2.58, Nov. 1997.

[142] P. Vandaele and M. Moonen, "A Channel Coding Based Algorithm for Multi-User Blind Symbol Estimation," in *Proceedings of the 5th Symposium on Communications and Vehicular Technology in the Benelux, Twente, The Netherlands*, pp. 64–69, Oct. 1997.

[143] P. Vandaele and M. Moonen, "Performance Analysis of Adaptive Subspace Based Algorithms for Blind Equalization of Single-User Systems," in *Proceedings of the COST 254 Workshop on 'Emerging Techniques for Communication Terminals', Toulouse, France*, p. 5.5, July 1997.

[144] P. Vandaele and M. Moonen, "A Recursive Total Least Squares Algorithm for Single User Blind Channel Equalization," Tech. Rep. 97-29, K.U.Leuven ESAT-SISTA, 1997. Submitted for publication.

[145] P. Vandaele and M. Moonen, "A Recursive Total Least Squares Algorithm for Deconvolution Problems," in *Proceedings of ICASSP, Seattle, USA*, pp. 1401–1404, May 1998.

[146] P. Vandaele and M. Moonen, "A Stochastic Subspace Algorithm for Blind Channel Identification in Noise Fields With Unknown Spatial Covariance," Tech. Rep. 98-117, K.U.Leuven ESAT-SISTA, 1998. Accepted for publication in Signal Processing.

[147] P. Vandaele and M. Moonen, "Adaptive Algorithms for Single User Blind Channel Equalization," *Mathematics in Signal Processing IV,*

(McWhirter J.G., Proudler I.K., eds.), Proc. of the Fourth IMA conference on Mathematics in Signal Processing, Warwick, UK, Dec. 1996, Oxford University Press (Oxford, UK), pp. 179–194, 1998.

[148] P. Vandaele and M. Moonen, "An 'SVD + Viterbi' Algorithm for Multi-User Adaptive Blind Equalization of Mobile Radio Channels," *IEEE Signal Processing Letters*, vol. 5, pp. 260–264, Oct. 1998.

[149] P. Vandaele and M. Moonen, "A New Non-Iterative Multi-User Blind Equalization Algorithm Based on Linear Block Coding," Tech. Rep. 98-102, K.U.Leuven ESAT-SISTA, 1998.

[150] P. Vandaele and M. Moonen, "Two Deterministic Blind Channel Estimation Algorithms Based on Oblique Projections," Tech. Rep. 98-114, K.U.Leuven ESAT-SISTA, 1998. Accepted for publication in Signal Processing.

[151] P. Vandaele and M. Moonen, "A Stochastic Subspace Algorithm for Blind Channel Identification in Noise Fields With Unknown Spatial Color," in *Proceedings of ICASSP, Phoenix AZ, USA*, pp. 1373–1376, Mar. 1999.

[152] P. Vandaele, G. Rombouts, and M. Moonen, "Implementation of an RTLS Blind Equalization Algorithm on DSP," in *Proceedings of the 9th IEEE International Workshop on Rapid System Prototyping, Leuven, Belgium*, pp. 150–155, June 1998.

[153] P. Vandenameele, S. Thoen, M. Engels, and H. De Man, "A Combined OFDM/SDMA Approach for WLAN," in *Proceedings of VTC, Houston, Texas*, pp. 1712–1716, May 1999.

[154] P. Vandenameele, L. Van der Perre, M. Engels, and H. De Man, "A Novel Class of Uplink OFDM/SDMA Algorithms for WLAN," in *Globecom '99*, Dec. 1999.

[155] J. Vandewalle and B. De Moor, "A Variety of Applications of the Singular Value Decomposition in Identification and Signal Processing," *SVD and Signal Processing: Algorithms, Applications and Architectures (E. Deprettere, ed.)*, pp. 43–91, 1988.

[156] A.J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. 13, pp. 260–269, Apr. 1967.

[157] M. Wax and T. Kailath, "Detection of Signals by Information Theoretic Criteria," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 2, pp. 387–392, Apr. 1985.

[158] Wirawan, P. Duhamel, and H. Maître, "Multi-Channel High Resolution Blind Image Restoration," in *Proceedings ICASSP*, pp. 1391–1394, Mar. 1999.

[159] G. Xu, H. Liu, L. Tong, and T. Kailath, "A Least-Squares Approach to Blind Channel Identification," *IEEE Transactions on Signal Processing*, vol. 43, no. 12, pp. 2982–2993, Dec. 1995.

[160] G. Xu, L. Tong, and H. Liu, "A New Algorithm for Fast Blind Equalization of Wireless Communication Channels," in *Proceedings ICASSP*, vol. 4, pp. 589–592, Apr. 1994.

[161] H. Zeng and L. Tong, "Connections Between the Least-Squares and Subspace Approaches to Blind Channel Estimation," *IEEE Transactions on Signal Processing*, vol. 44, pp. 1593–1596, June 1996.

[162] H. Zeng and L. Tong, "Blind Channel Estimation Using the Second-Order Statistics: Algorithms," *IEEE Transactions on Signal Processing*, vol. 45, no. 8, pp. 1919–1930, Aug. 1997.

[163] H. Zeng and L. Tong, "Blind Channel Estimation Using the Second-Order Statistics: Asymptotic Performance and Limitations," *IEEE Transactions on Signal Processing*, vol. 45, no. 8, pp. 2060–2071, Aug. 1997.