

Polynomial Matrix Eigenvalue Decomposition Techniques for Multichannel Signal Processing



Zeliang Wang

School of Engineering
Cardiff University

This dissertation is submitted for the degree of
Doctor of Philosophy

September 2017

To my parents,
for their constant love and support.

DECLARATION

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award.

Signed (candidate) Date

STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD.

Signed (candidate) Date

STATEMENT 2

This thesis is the result of my own independent work/investigation, except where otherwise stated, and the thesis has not been edited by a third party beyond what is permitted by Cardiff University's Policy on the Use of Third Party Editors by Research Degree Students. Other sources are acknowledged by explicit references. The views expressed are my own.

Signed (candidate) Date

STATEMENT 3

I hereby give consent for my thesis, if accepted, to be available online in the University's Open Access repository and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed (candidate) Date

Abstract

Polynomial eigenvalue decomposition (PEVD) is an extension of the eigenvalue decomposition (EVD) for para-Hermitian polynomial matrices, and it has been shown to be a powerful tool for broadband extensions of narrowband signal processing problems. In the context of broadband sensor arrays, the PEVD allows the para-Hermitian matrix that results from the calculation of a space-time covariance matrix of the convolutively mixed signals to be diagonalised. Once the matrix is diagonalised, not only can the correlation between different sensor signals be removed but the signal and noise subspaces can also be identified. This process is referred to as broadband subspace decomposition, and it plays a very important role in many areas that require signal separation techniques for multichannel convolutive mixtures, such as speech recognition, radar clutter suppression, underwater acoustics, etc.

The multiple shift second order sequential best rotation (MS-SBR2) algorithm, built on the most established SBR2 algorithm, is proposed to compute the PEVD of para-Hermitian matrices. By annihilating multiple off-diagonal elements per iteration, the MS-SBR2 algorithm shows a potential advantage over its predecessor (SBR2) in terms of the computational speed. Furthermore, the MS-SBR2 algorithm permits us to minimise the order growth of polynomial matrices by shifting rows (or columns) in the same direction across iterations, which can potentially reduce the computational load of the algorithm.

The effectiveness of the proposed MS-SBR2 algorithm is demonstrated by various para-Hermitian matrix examples, including randomly generated matrices with different sizes and matrices generated from source models with different dynamic ranges and relations between the sources' power spectral densities. A worked example is presented to demonstrate how the MS-SBR2 algorithm can be used to strongly decorrelate a set of convolutively mixed signals. Furthermore, the performance metrics and computational complexity of MS-SBR2 are analysed and compared to other existing PEVD algorithms by means of numerical examples.

Finally, two potential applications of the MS-SBR2 algorithm, including multichannel spectral factorisation and decoupling of broadband multiple-input multiple-output (MIMO) systems, are demonstrated in this dissertation.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Prof. John McWhirter for his continuous support, advice and encouragement throughout my PhD studies. Without his gracious guidance, I cannot imagine how difficult it would be for me to reach the destination. I would also like to thank the Engineering and Physical Sciences Research Council (EPSRC) and the University Defence Research Collaboration (UDRC) in Signal Processing, Grant number EP/K014307/1, for partially funding my studies. My special thanks go to Dr. Stephan Weiss for always being there whenever help is needed, also for providing constructive ideas and proofreading my publications.

Publications

- Z. Wang and J. G. McWhirter, “A New Multichannel Spectral Factorization Algorithm for Parahermitian Polynomial Matrices,” in *10th IMA International Conference on Mathematics in Signal Processing*, Birmingham, England, Dec. 2014.
- Z. Wang, J. G. McWhirter, J. Corr and S. Weiss, “Multiple Shift Second Order Sequential Best Rotation Algorithm for Polynomial Matrix EVD,” in *23rd European Signal Processing Conference (EUSIPCO)*, Nice, France, Sep. 2015.
- Z. Wang, J. G. McWhirter and S. Weiss, “Multichannel Spectral Factorization Algorithm Using Polynomial Matrix Eigenvalue Decomposition,” in *49th Asilomar Conference on Signals, Systems and Computers*, CA, USA, Nov. 2015.
- Z. Wang, A. Sandmann, J. G. McWhirter and A. Ahrens, “Multiple Shift SBR2 Algorithm for Calculating the SVD of Broadband Optical MIMO Systems,” in *39th IEEE International Conference on Telecommunications and Signal Processing*, Vienna, Austria, Jun. 2016.
- Z. Wang, J. G. McWhirter, J. Corr and S. Weiss, “Order-Controlled Multiple Shift SBR2 Algorithm for Para-Hermitian Polynomial Matrices,” in *9th IEEE Sensor Array and Multichannel Signal Processing Workshop*, Rio de Janeiro, Brazil, Jul. 2016.
- A. Ahrens, A. Sandmann, S. Lochmann and Z. Wang, “Decomposition of Optical MIMO Systems using Polynomial Matrix Factorization,” in *2nd IET International Conference on Intelligent Signal Processing*, London, UK, Dec. 2015.
- A. Ahrens, A. Sandmann, Z. Wang and J. G. McWhirter, “Polynomial Matrix SVD Algorithms for Broadband Optical MIMO Systems,” in *7th International Conference on Optical Communication Systems*, Lisbon, Portugal, Jul. 2016.
- J. G. McWhirter and Z. Wang, “A Novel Insight to the SBR2 Algorithm for Diagonalising Para-Hermitian Matrices,” in *11th IMA International Conference on Mathematics in Signal Processing*, Birmingham, England, Dec. 2016.
- Z. Wang, A. Sandmann, J. G. McWhirter and A. Ahrens, “Decoupling of Broadband Optical MIMO Systems Using the Multiple Shift SBR2 Algorithm,” *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems (Invited Paper)*, vol. 6(1), pp. 30-37, 2017.

Contents

List of Figures	ix
List of Tables	xiii
Nomenclature	xiv
1 Introduction	1
1.1 Motivation	1
1.2 Thesis Contributions	4
1.3 Organisation of the Thesis	6
1.4 Choice of Notation	8
2 Background to Polynomial Matrix Decomposition Techniques	10
2.1 Introduction	10
2.2 Matrix Decomposition Techniques for Narrowband Channels	11
2.2.1 Instantaneous Mixing Model	11
2.2.2 Blind Source Separation	12
2.2.3 Eigenvalue Decomposition	15
2.2.4 Singular Value Decomposition	18
2.3 Matrix Decomposition Techniques for Broadband Channels	20
2.3.1 Background to Polynomial Matrices	20
2.3.2 Convolutional Mixing Model	22
2.3.3 Polynomial Eigenvalue Decomposition	23
2.3.4 Polynomial Singular Value Decomposition	24
2.4 Other Polynomial Matrix Decomposition Techniques	25
2.4.1 Smith Form	25
2.4.2 Smith-McMillan Form	25
2.4.3 FIR Lossless System Decomposition	26
2.4.4 Lambert's Approach for Multichannel Blind Deconvolution	28
2.4.5 Approximate Polynomial Eigenvalue Decomposition	28
2.5 Chapter Summary	29
3 Iterative Algorithms for Polynomial Eigenvalue Decomposition	30
3.1 Introduction	30
3.2 Generality of Iterative PEVD Algorithms	31
3.2.1 PEVD Uniqueness and Ambiguity	31
3.2.2 Anatomy of the PEVD Algorithms	32

3.3	Second Order Sequential Best Rotation Algorithm	35
3.3.1	Outline of the SBR2 Algorithm	35
3.3.2	Applications of the SBR2 Algorithm	37
3.4	A Novel Insight into the SBR2 Algorithm	41
3.4.1	Coding Gain	41
3.4.2	Proof of Spectral Majorisation	42
3.4.3	Modified SBR2 Algorithm	44
3.5	Sequential Matrix Diagonalisation Algorithms	46
3.5.1	The SMD Algorithm	48
3.5.2	Maximum Element SMD Algorithm	50
3.5.3	Multiple Shift Maximum Element SMD Algorithm	50
3.6	Polynomial Order Shortening Methods	51
3.6.1	Limitations of the PEVD Algorithms	51
3.6.2	Truncation Methods for Para-Hermitian Matrices	53
3.6.3	Truncation Methods for Paraunitary Matrices	54
3.7	Chapter Summary	57
4	Multiple Shift SBR2 Algorithm for Polynomial Eigenvalue Decomposition	58
4.1	Introduction	58
4.2	Outline of the MS-SBR2 Algorithm	59
4.3	Proof of Convergence	62
4.4	Time-Shift Strategies of the MS-SBR2 Algorithm	64
4.4.1	The Conventional Time-Shift Method	64
4.4.2	The Direction-Fixed Time-Shift Method	65
4.5	Numerical Examples	69
4.5.1	Performance Metrics	70
4.5.2	Simulation Scenario	71
4.5.3	Algorithm Convergence	72
4.5.4	Polynomial Matrix Order	72
4.6	Impact of Source Model Matrix Conditioning	77
4.6.1	Source Model	78
4.6.2	Simulation Scenario	79
4.6.3	Algorithm Convergence and Paraunitary Order	80
4.6.4	Spectral Ordering	81
4.7	Strong Decorrelation of Convolutively Mixed Signals	83
4.7.1	Simulation Scenario	83
4.7.2	Results and Analysis	84
4.8	Chapter Summary	94
5	Comparative Analysis of the PEVD Algorithms	96
5.1	Introduction	96
5.2	Analysis of Computational Complexity	97
5.2.1	Computational Complexity of the SBR2 Family	98
5.2.2	Computational Complexity of the SMD Family	99
5.3	Experimental Results and Analysis	102
5.3.1	Simulation Scenario and Performance Metrics	102
5.3.2	Computational Run-Time Evaluation	103

5.3.3	Convergence Analysis	104
5.3.4	Polynomial Order Truncation	106
5.4	Chapter Summary	112
6	Multichannel Spectral Factorisation using the Polynomial Eigenvalue Decomposition Method	115
6.1	Introduction	115
6.2	One-Dimensional Spectral Factorization	117
6.3	Multichannel Spectral Factorisation	118
6.3.1	Ambiguity of Multichannel Spectral Factorisation	119
6.3.2	Outline of the Proposed Algorithm	120
6.3.3	Order Shortening of the Spectral Factor	121
6.4	Numerical Examples	122
6.4.1	Example 1	122
6.4.2	Example 2	125
6.5	Chapter Summary	128
7	Decoupling of Broadband MIMO Systems using the Polynomial Singular Value Decomposition by Polynomial Eigenvalue Decomposition Method	129
7.1	Introduction	129
7.2	The State of the Art	130
7.3	Decoupling of MIMO Channel using PSVD	132
7.3.1	Accuracy of the Decomposition	136
7.3.2	Transmission Quality	137
7.4	Numerical Examples	138
7.4.1	Example 1	138
7.4.2	Example 2	143
7.5	Chapter Summary	148
8	Conclusions and Future Work	150
8.1	Conclusions	150
8.2	Suggestions for Future Work	152
	References	155

List of Figures

1.1	Typical setup with multiple sources and multiple sensors.	3
2.1	Block diagram of a narrowband MIMO system with additive noise . .	12
2.2	A block diagram of the MIMO communication system using the SVD-based equalisation scheme.	19
2.3	Block diagram of a broadband MIMO system consisting of a number of FIR channels	23
3.1	A 3D illustration showing a single iteration of the SBR2 algorithm when applied to diagonalising a 5×5 para-Hermitian matrix example.	37
3.2	Strong decorrelation for a set of convolutively mixed sensor signals. .	38
3.3	The stem plot of (a) the CSD matrix and (b) the diagonalised CSD matrix obtained from the modified SBR2.	46
3.4	Convergence of the SBR2 algorithm when applied to the example CSD matrix in Figure 3.3, showing (a) the behaviour of the coding gain $G^{(i)}$ and (b) the behaviour of the magnitude of the maximum off-diagonal element found at the i -th iteration.	47
4.1	Illustration of the search space for a 6×6 para-Hermitian matrix example in the MS-SBR2 algorithm.	62
4.2	A 3D illustration of the MS-SBR2 algorithm with the conventional time-shift method, showing a single iteration of diagonalising a 5×5 example para-Hermitian matrix.	66
4.3	A 3D illustration of the MS-SBR2 algorithm with the direction-fixed time-shift method, showing a single iteration of diagonalising a 5×5 example para-Hermitian matrix.	68
4.4	Convergence comparison of different versions of SBR2 for diagonalising 2000 randomly generated para-Hermitian matrices.	73
4.5	Comparison of polynomial orders among different versions of SBR2 without using any truncation methods.	74
4.6	Results obtained from different SBR2 algorithms for the lag based truncation method with $\mu_{PU} = 10^{-4}$	75
4.7	Results obtained from different SBR2 algorithms for the energy based truncation method with $\mu_{PH} = 10^{-4}$	76
4.8	Block diagram of a source model consisting of N independent zero mean unit variance complex Gaussian sources, innovation filters and a paraunitary convolutive mixing system.	78

4.9	Remaining off-diagonal energy versus iterations for the SBR2 and MS-SBR2 algorithms over an ensemble of 200 random realisations, showing both majorisation types with (a) 10 dB dynamic range, and (b) 30 dB dynamic range.	80
4.10	Paraunitary matrix order for the SBR2 and MS-SBR2 algorithms over an ensemble of 200 random realisations, showing both majorisation types with (a) 10 dB dynamic range, and (b) 30 dB dynamic range. . .	81
4.11	PSDs of the on-diagonal polynomials for (a) a strictly majorised source model, and (b) an unmajorised source model, both with 10 dB dynamic range obtained from MS-SBR2 after 150 iterations, superimposed on light grey ideal PSDs.	82
4.12	PSDs of the on-diagonal polynomials for (a) a strictly majorised source model, and (b) an unmajorised source model, both with 30 dB dynamic range obtained from MS-SBR2 after 150 iterations, superimposed on light grey ideal PSDs.	83
4.13	The stem plot of the estimated space-time covariance matrix of the convolutively mixed signals with the chosen SNR of 5.08 dB	85
4.14	The magnitude of the maximum off-diagonal element found at the i -th iteration for the SBR2 and MS-SBR2 algorithms.	86
4.15	The stem plot of the strongly decorrelated CSD matrix after applying the MS-SBR2 algorithm.	87
4.16	The stem plot of the generated paraunitary polynomial matrix after applying the MS-SBR2 algorithm.	87
4.17	Power spectral densities of (a) the convolutively mixed signals, and (b) the strongly decorrelated signals generated using the paraunitary matrix obtained from the MS-SBR2 algorithm.	89
4.18	Total power spectral density of the expected signals, before and after applying the paraunitary transformation matrix obtained from the MS-SBR2 algorithm.	90
4.19	The polynomial orders of: (a) the paraunitary matrix and (b) the transformed para-Hermitian matrix at the end of each iteration i of the MS-SBR2 algorithm for the cases when no truncation method is used and when the lag based truncation method is applied with a chosen set of values of μ_{PU} and μ_{PH}	93
4.20	The truncated diagonal matrix produced by applying the MS-SBR2 algorithm to the CSD matrix example shown in Figure 4.13, implementing the energy based para-Hermitian truncation method with $\mu_{PH} = 10^{-4}$	93
4.21	The truncated paraunitary matrix produced by applying the MS-SBR2 algorithm to the CSD matrix example shown in Figure 4.13, implementing the lag based paraunitary truncation method with $\mu_{PU} = 10^{-4}$	94
5.1	Time complexity of different PEVD algorithms for diagonalising randomly generated para-Hermitian matrices, showing mean execution time of a single iteration for varying matrix and lag dimensions.	104

5.2	Convergence comparison of the PEVD algorithms showing the reduction of off-diagonal energy vs. mean execution time over 100 iterations for an ensemble of randomised para-Hermitian matrices with sizes $M = 6, 12,$ and 24	105
5.3	Comparison of paraunitary order truncation methods when applied to the different PEVD algorithms.	110
5.4	Average paraunitary matrix reconstruction error vs. iterations for the different truncation methods and PEVD algorithms.	111
6.1	The resulting matrices obtained from applying the MS-SBR2 algorithm to Example (6.14), showing (a) the diagonalised matrix with lag bound fixed truncation and (b) the truncated paraunitary matrix resulting from lag based truncation.	123
6.2	Power spectral density for the on-diagonal polynomials of the diagonalised matrix obtained from MS-SBR2.	123
6.3	The resulting spectral factors of Example (6.14), showing (a) the <i>outer</i> spectral factor and (b) the <i>inner</i> spectral factor.	124
6.4	The diagonalised matrix obtained from applying the MS-SBR2 algorithm to the CSD matrix example in Figure 4.13 and truncated to the same order as the CSD matrix.	126
6.5	The <i>outer</i> spectral factor of the CSD matrix example in Figure 4.13, implementing the lag based truncation with $\mu_{\text{PU}} = 10^{-2}$	127
6.6	The <i>outer</i> spectral factor of the CSD matrix example in Figure 4.13, implementing the row-shift corrected truncation with $\mu'_{\text{PU}} = 10^{-2}$	127
7.1	A block diagram of the proposed MIMO communication system using the PSVD based equalisation scheme.	134
7.2	A block diagram of the layer-specific ZF equalisation for each SISO channel obtained using the PSVD.	135
7.3	The equivalent SISO channel model with ISI removed using the layer-specific ZF equalisation.	135
7.4	The stem plot of the 4×3 broadband MIMO channel matrix, showing the magnitudes of the channel impulse responses at different time lags.	139
7.5	The stem plot of the 4×4 paraunitary matrix obtained from the PSVD by MS-SBR2 method, showing the magnitudes of the coefficients.	140
7.6	The stem plot of the 3×3 paraunitary matrix obtained from the PSVD by MS-SBR2 method, showing the magnitudes of the coefficients.	140
7.7	The stem plot of the diagonalised 4×3 MIMO channel matrix obtained from the PSVD by MS-SBR2 method, showing the magnitudes of the coefficients.	141
7.8	Power spectral densities of the on-diagonal polynomials in the 4×3 MIMO channel matrix, showing (a) before diagonalisation and (b) after diagonalisation using the PSVD by MS-SBR2.	143
7.9	Intensity distribution patterns of an MMF when launching the light with the radial offsets (a) $\delta = 0 \mu\text{m}$ (centric) and (b) $\delta = 15 \mu\text{m}$ (eccentric), where the dashed circle has a diameter of $50 \mu\text{m}$	144

7.10	An overview of the testbed for measuring the impulse responses of a 2×2 optical MIMO channel.	145
7.11	The stem plot of the measured 2×2 optical MIMO channel matrix, showing the magnitudes of the channel impulse responses.	146
7.12	The resulting paraunitary matrices obtained from the MS-SBR2 algorithm.	146
7.13	The stem plot of the diagonalised channel matrix obtained from applying the PSVD by MS-SBR2 algorithm to the example in Figure 7.11.	147
7.14	Results of BER obtained from applying the proposed PSVD based equalisation scheme to a measured 2×2 optical MIMO channel, showing the comparison between different QAM transmission modes with and without the power allocation scheme.	149

List of Tables

4.1	Performance comparisons between the SBR2 and MS-SBR2 algorithms when applied to the same CSD matrix example in Figure 4.13, without implementing any order truncation process.	88
4.2	Performance measures of the MS-SBR2 algorithm when applied to the same CSD matrix example in Figure 4.13, implementing the energy based para-Hermitian truncation and lag based paraunitary truncation methods for different values of μ_{PH} and μ_{PU}	92
5.1	Computational complexity of the different PEVD algorithms.	101
5.2	Average length of the resulting polynomial matrices obtained from applying various PEVD algorithms to an ensemble of randomised para-Hermitian matrices with sizes $M = 6, 12,$ and 24 for 100 iterations, without using any truncation methods.	106
5.3	Average order of the truncated para-Hermitian matrix and the corresponding reconstruction error obtained from the PEVD algorithms after 100 iterations, implementing the energy based truncation method with different values of μ_{PH}	108
5.4	Average order of the truncated paraunitary matrix obtained from the PEVD algorithms after 100 iterations, implementing the lag based and row-shift corrected truncation methods, respectively.	111
5.5	Average paraunitary matrix reconstruction error obtained from the PEVD algorithms after 100 iterations, showing (a) implementing the lag based truncation with μ_{PU} and (b) the row-shift corrected truncation with μ'_{PU}	112
7.1	Results obtained from applying the SBR2 and MS-SBR2 algorithms for calculating the PSVD to the polynomial matrix in Figure 7.4, with the truncation parameters set as $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-4}$ and the stopping condition as $\varepsilon = 10^{-3}$ in both methods.	142
7.2	Investigated transmission modes of the 2×2 optical MIMO system.	148

Nomenclature

Notation

a, A	regular lower or upper case characters denote scalar quantities
\mathbf{v}	bold lower case characters denote normal vector quantities
\mathbf{A}	bold upper case characters denote normal matrix quantities
v_j	j -th element of vector \mathbf{v}
a_{jk}	element located in the j -th row and k -th column of matrix \mathbf{A}
$\underline{\mathbf{v}}(z)$	underscored bold lowercase characters denote polynomial vectors
$\underline{\mathbf{A}}(z)$	underscored bold uppercase characters denote polynomial matrices
\mathbf{I}_M	$M \times M$ identity matrix
$\ \cdot\ _F$	Frobenius norm (the square-root sum of squares of all elements)
$\ \cdot\ _2$	L_2 norm
$ \cdot $	modulus
$\ \cdot\ _\infty$	L_∞ norm
$E\{\cdot\}$	expectation operator
$\{\cdot\}^*$	complex conjugate operator
$\{\cdot\}^T$	transpose operator
$\{\cdot\}^H$	Hermitian transpose operator
$\tilde{\{\cdot\}}$	paraconjugate operator
$\det\{\mathbf{A}\}$	determinant of matrix \mathbf{A}
$\text{diag}\{\underline{\mathbf{A}}(z)\}$	diagonal matrix with diagonal elements given by $\underline{\mathbf{A}}(z)$
$\text{off}\{\underline{\mathbf{A}}(z)\}$	a matrix with diagonal entries equal to 0 and off-diagonal entries given by $\underline{\mathbf{A}}(z)$
$\text{rank}\{\mathbf{A}\}$	rank of matrix \mathbf{A}
$\text{trace}\{\mathbf{A}\}$	trace of matrix \mathbf{A}
$\mathbb{C}, \mathbb{C}^N, \mathbb{C}^{M \times N}$	set of complex numbers, vectors with N rows and matrices with M rows and N columns
$\underline{\mathbb{C}}^{M \times N}$	set of $M \times N$ polynomial matrices with complex coefficients (without specifying the number of lags)
$\mathbb{C}^{M \times N \times L}$	set of $M \times N$ complex polynomial matrices with lag length of L

$\mathbb{R}, \mathbb{R}^N, \mathbb{R}^{M \times N}$	set of real numbers, vectors with N rows and matrices with M rows and N columns
\mathbb{Z}	set of integer numbers
z^{-1}	unit delay operator (indeterminate variable of polynomial quantities)

Abbreviations

BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
BSS	Blind Source Separation
CCI	Co-Channel Interference
CSD	Cross Spectral Density
DOA	Direction of Arrival
DSP	Digital Signal Processing
EVD	Eigenvalue Decomposition
FIR	Finite Impulse Response
HOS	Higher Order Statistics
IBI	Inter-Block Interference
ICA	Independent Component Analysis
ISI	Inter-Symbol Interference
LTI	Linear, Time-Invariant
ME-SMD	Maximum Element SMD Algorithm
MIMO	Multiple Input Multiple Output
MLSE	Maximum Likelihood Sequence Estimation
MSF	Multichannel Spectral Factorization
MSME-SMD	Multiple Shift Maximum Element SMD Algorithm
MS-SBR2	Multiple Shift SBR2 Algorithm
OCMS-SBR2	Order Controlled Multiple Shift SBR2 Algorithm
OFDM	Orthogonal Frequency Division Multiplexing
PCA	Principal Component Analysis
PCFB	Principal Component Filter Bank
PEVD	Polynomial Eigenvalue Decomposition
PSD	Power Spectral Density
PSVD	Polynomial Singular Value Decomposition
QAM	Quadrature Amplitude Modulation
SBR2	Second Order Sequential Best Rotation Algorithm
SISO	Single Input Single Output

Nomenclature

SMD	Sequential Matrix Diagonalisation
SNR	Signal-to-Noise Ratio
SOS	Second Order Statistics
STVC	Spatio Temporal Vector Coding
SVD	Singular Value Decomposition
ZF	Zero Forcing

Chapter 1

Introduction

1.1 Motivation

Digital signal processing (DSP) has become a major area of interest since the 1960s when digital computers first became available [1, 2]. In 1965, Cooley and Tukey [3] invented the fast Fourier transform (FFT), which dramatically boosted the development of signal processing. The FFT underpins a wide range of applications such as speech processing, digital medical imaging, and wireless communications. DSP is a discipline encompassing mathematics, algorithms, and techniques that can be used to manipulate signals like voice, audio, video, pressure, temperature, etc. These signals usually originate as digitised sensory data from the real world, such as sound waves, visual images, seismic vibrations and electromagnetic waves. DSP has a variety of goals ranging from data compression for transmission and storage to speech recognition, enhancement of visual images, and data encryption, among others.

The early applications of DSP were predominantly focused on four key areas: radar and sonar, medical imaging, space exploration, and oil exploration. Since the invention of personal computers and cellular phones in the 1980s, DSP has rapidly expanded into many new areas driven by the commercial marketplace, such as digital communications and sensor array processing [1]. A commonly desired objective in these areas

is to estimate the source signals (or a particular source) from a set of observed signals, which are usually masked by noise. This process is often referred to as blind source separation (BSS), where the term *blind* signifies the lack of a priori knowledge of the source signals and the mixing model from the sources to the sensors. BSS can generally be described by a typical multiple-source and multiple-sensor scenario, as shown in Figure 1.1. Depending on whether the observed signals are instantaneously mixed or convolutively mixed, BSS problems can generally be divided into two categories. In the instantaneous case, the mixing system is modelled by a scalar matrix whose elements are generally complex numbers, and each sensor signal is simply the sum of differently weighted source signals plus the noise. Many algorithms have been developed to address the instantaneous BSS problem, and the majority of these algorithms requires the same pre-processing step based on second-order statistics (SOS). This pre-processing step uses a matrix decomposition method such as eigenvalue decomposition (EVD) or singular value decomposition (SVD) to decorrelate the sensor signals, and then higher-order statistics (HOS) is used to complete the source separation process. For further details of this method, see [4].

In the more complicated convolutive mixing model, the source signals are received at an array of sensors over multiple paths and with different time delays. A typical example could be speech recordings made in a room in the presence of background noise. Each element of the mixing matrix is now seen as a finite impulse response (FIR) filter, and therefore the convolutive mixing matrix takes the form of a polynomial matrix in terms of the indeterminate variable z^{-1} . In the past, most techniques have addressed the convolutive BSS problem by transforming the signals into a number of narrower frequency bands using the discrete Fourier transform (DFT), and each frequency band is then considered as a narrowband BSS problem. This is commonly referred to as independent frequency band (IFB) processing [5, 6]. However, the IFB approach may ignore the important correlations between different bands, and it can also lead to a lack of coherence of signals. An algorithm, called second order sequential

best rotation (SBR2), has been developed for calculating the EVD of a para-Hermitian polynomial matrix [7]. This algorithm is capable of performing strong decorrelation on a set of convolutively mixed signals. In other words, this algorithm can be used as the pre-processing method for broadband BSS, thus avoiding the drawbacks imposed by the IFB approach.

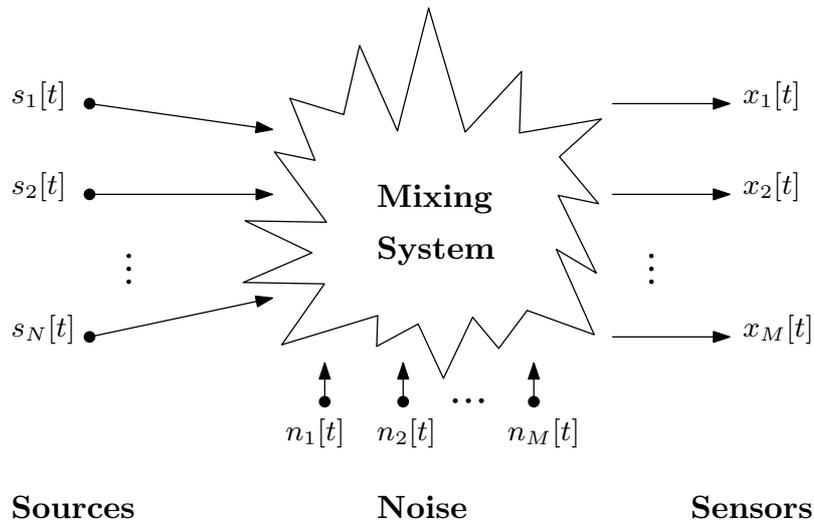


Figure 1.1 Typical setup with multiple sources and multiple sensors.

Over the last decade, polynomial EVD (PEVD) has been exploited widely in the areas of broadband signal processing. Just as unitary or orthogonal matrix decomposition techniques such as EVD and SVD are fundamental to most narrowband signal processing formulations [8], PEVD is providing optimal solutions to many broadband extensions of narrowband problems, including broadband sensor array processing [9–13], MIMO communications [14–16], channel coding [17], spectral factorisation [18–20], filter bank design for subband coding [21], and broadband BSS [22–25]. The motivation behind this thesis is to further develop PEVD techniques which can be used to address broadband signal processing problems. In particular, improved versions of the SBR2 algorithm are developed for computing the PEVD. These algorithms have been proven to converge and are also numerically robust. Furthermore, potential applications of the proposed algorithms are demonstrated.

1.2 Thesis Contributions

The main contributions of this thesis are summarised as follows.

- **A novel insight into the SBR2 algorithm for diagonalising para-Hermitian matrices [26]**

In broadband sensor array processing, the SBR2 algorithm [7] has been used to decorrelate the convolutively mixed signals measured from sensors. It has been shown that the strongly decorrelated signals obtained using the SBR2 algorithm also possess the spectral majorisation property [27]; however, this property has not been proven in the existing literature. We take a fresh look at the SBR2 algorithm in terms of its potential for optimising the subband coding gain, which leads to the proof of the spectral majorisation property. Further details of this work are presented in Chapter 3.

- **Multiple shift SBR2 algorithm for polynomial eigenvalue decomposition [28, 29]**

Aiming to diagonalise a para-Hermitian polynomial matrix in fewer iterations, an improved version of the SBR2 algorithm, namely multiple shift SBR2 (MS-SBR2), is proposed for implementing the PEVD. The MS-SBR2 algorithm is developed based on the original SBR2 algorithm; however, it can achieve faster convergence than SBR2 in terms of reducing the off-diagonal energy. Specifically, by using a multiple-shift strategy akin to that of the multiple shift maximum element sequential matrix diagonalisation (MSME-SMD) algorithm [30], MS-SBR2 can annihilate multiple off-diagonal elements at each iteration compared to SBR2. Furthermore, two different time-shift methods are introduced in MS-SBR2, including a conventional shift method similar to that of SBR2 and a direction-fixed shift method. In particular, using the direction-fixed shift method keeps all the row (column) shifts in the same direction throughout each iteration,

which therefore gives us the flexibility to control unnecessary growth in the order of the resulting polynomial matrices. This leads to the order-controlled MS-SBR2 algorithm, which is advantageous over conventional MS-SBR2 in terms of reducing polynomial order. The reduced polynomial order reduces the computational load of the algorithm, and therefore produce a computationally fast algorithm. The MS-SBR2 algorithm comprises the core part of this thesis, and a detailed account is presented in Chapter 4.

- **Multichannel spectral factorisation based on polynomial eigenvalue decomposition [19, 20]**

Spectral factorisation plays a fundamental role in designing a minimum-phase system that can be used in many areas of signal processing. We propose a novel method for solving the multichannel spectral factorisation problem. The proposed method uses a PEVD algorithm to diagonalise the power spectral density matrix, which effectively converts the multichannel spectral factorisation problem into a number of independent scalar spectral factorisation problems for which suitable algorithms already exist. This work contributes one of the potential applications of the proposed MS-SBR2 algorithm, which is demonstrated in Chapter 6.

- **Decoupling of broadband multiple-input multiple-output systems using the PSVD by PEVD method [31, 32]**

A popular strategy for channel equalisation of narrowband multiple-input multiple-output (MIMO) communication systems is to use SVD to decompose the MIMO channel into a set of independent single input single output (SISO) channels, provided that the channel state information is available at both the transmitter and the receiver. The resulting SISO channels can then be used to enhance the channel capacity or to increase diversity [33]. When the problem is extended to the broadband case, the MIMO channel becomes time-dispersive and therefore is

represented by a polynomial (or convolutive) mixing matrix, where each of the entries can be seen as an FIR filter to account for the multipath effect. Thus, apart from the co-channel interference (CCI) caused by the MIMO components, there is also inter-symbol interference (ISI) between the transmit symbols. To address the broadband MIMO channel equalisation problem, PSVD can be used to remove the CCI by decomposing the frequency-selective MIMO channel into a number of independent frequency-selective SISO channels [34], and the remaining ISI for each SISO channel can be eliminated by further equalisation techniques, such as zero-forcing (ZF) equalisation or maximum likelihood sequence estimations (MLSE). The main contributions of this work include the demonstration of the proposed PSVD by MS-SBR2 method in terms of solving the broadband MIMO decoupling problem and performance comparisons with the existing PSVD by SBR2 method. In addition, we present two examples demonstrating our proposed method, including a simulated MIMO channel and a measured 2×2 optical MIMO channel. Details of this work can be found in Chapter 7.

1.3 Organisation of the Thesis

The rest of the chapters of this thesis are structured as follows.

Chapter 2 presents a literature review chapter which gives a comprehensive account of MIMO channel mixing models and their corresponding matrix decomposition techniques. Depending on characteristics of the channel, the channel can either be modelled by an instantaneous mixing matrix in the narrowband (frequency-flat) signal processing or a convolutive mixing matrix in the broadband (frequency-selective) case. In the narrowband sensor array processing, two common scalar matrix decomposition techniques, including EVD and SVD, will be introduced, along with a discussion of relevant applications. Following on that, some ex-

isting approaches for broadband sensor array processing are briefly reviewed. In particular, two polynomial matrix decomposition techniques are briefly introduced, including the PEVD and the PSVD methods.

Chapter 3 concentrates on the PEVD techniques. It starts by presenting a general anatomy of PEVD, including the uniqueness and ambiguity problems of the PEVD. Existing PEVD algorithms, including SBR2 [7], sequential matrix diagonalisation (SMD) [35], maximum element SMD (ME-SMD) [35], and multiple shift ME-SMD (MSME-SMD) [30], are then discussed. This gives a clear overview of how far the PEVD techniques have been developed so far. In particular, a novel insight into the SBR2 algorithm is presented in order to prove that it possesses the spectral majorisation property. In addition, we introduce some of the existing polynomial order truncation methods that are used to reduce the unnecessary order growth of the resulting polynomial matrices obtained from the PEVD algorithms.

Chapter 4 details the proposed MS-SBR2 algorithm for calculating PEVD of para-Hermitian matrices. First, the idea of the MS-SBR2 algorithm is discussed. Then, two different time-shift methods for MS-SBR2 are presented. Computer simulations are designed to test the effectiveness of the MS-SBR2 algorithm by means of different para-Hermitian matrix examples, and results are also compared with the original SBR2 algorithm. Last but not least, a simple broadband MIMO communication channel is modelled to demonstrate the ability and effectiveness of the MS-SBR2 algorithm in terms of decorrelating a set of convolatively mixed signals.

Chapter 5 presents a comparative analysis of all PEVD algorithms discussed in this thesis, including SBR2, MS-SBR2, SMD, ME-SMD, and MSME-SMD. We first investigate the computational complexity of the PEVD algorithms, followed by computer simulations to examine the real computational time of each PEVD al-

gorithm using a set of para-Hermitian matrices of different sizes. Furthermore, all algorithms are assessed and compared from the perspective of various performance metrics, including diagonalisation measure, convergence speed, and polynomial order.

Chapter 6 demonstrates a potential application of the proposed MS-SBR2 algorithm to multichannel (or matrix) spectral factorisation. Numerical examples are included to examine the validity of the proposed spectral factorisation method.

Chapter 7 discusses how the proposed MS-SBR2 algorithm can be used to formulate the PSVD and therefore applied to decouple broadband MIMO systems. We include two worked examples to demonstrate our proposed method. In the first example, a simulated 3×4 broadband MIMO channel is chosen to compare the performance between the PSVD by MS-SBR2 method and the PSVD by SBR2 method. In the second example, the proposed PSVD by MS-SBR2 algorithm is applied to decouple a measured 2×2 optical MIMO channel, and the bit error rate (BER) performance is assessed and compared for different transmission schemes with a fixed spectral efficiency.

Chapter 8 concludes the research presented in this thesis and provides suggestions for future work.

1.4 Choice of Notation

Throughout this thesis, scalar quantities are represented by regular lower or upper case characters. Vectors and matrices are denoted with bold lower and upper case characters, respectively. The entries of a vector \mathbf{v} are denoted by v_j , where the subscript j denotes the j -th element in \mathbf{v} . Similarly, the entries of a matrix \mathbf{A} are indicated by a_{jk} , where the subscripts j, k represent the j -th row and k -th column of \mathbf{A} . \mathbf{I}_M represents the $M \times M$ identity matrix. Dependency on a discrete variable is denoted by square brackets, while

dependency on a continuous variable is indicated by round brackets. The use of $\{\cdot\}$ under any matrices or vectors denotes polynomial quantities, e.g., $\mathbf{A}(z) = \sum_{\tau} \mathbf{A}[\tau]z^{-\tau}$, $\tau \in \mathbb{Z}$ represents a polynomial matrix in terms of the indeterminate variable z^{-1} . The superscript $\{\cdot\}^*$ is chosen as the complex conjugate operator of the coefficients of a polynomial matrix or vector, and the superscripts $\{\cdot\}^T$ and $\{\cdot\}^H$ stand for matrix transpose and Hermitian conjugate operation, respectively. The notation of $\{\tilde{\cdot}\}$ upon a polynomial matrix is used to denote the paraconjugate operation.

Chapter 2

Background to Polynomial Matrix

Decomposition Techniques

2.1 Introduction

In the context of sensor array processing, the problem of source separation can be classified into two categories depending on how the propagation of signals from sources to sensors have been modelled. For a narrowband sensor array, the propagation is usually modelled by an instantaneous mixing matrix whose elements are complex scalars. However, in the case of a broadband sensor array, the propagation of signals cannot be modelled by a scalar mixing matrix. Instead, a matrix of finite impulse response (FIR) filters is required to describe the convolutive mixtures.

This chapter provides a literature review of different matrix decomposition techniques for sensor array processing ranging from the narrowband case to the broadband case. It starts by introducing the simple instantaneous mixing model followed by the available scalar matrix decomposition techniques which can be used to address narrowband problems such as source separation and design of equalisation for multiple input multiple output (MIMO) communications. After that, the more complicated convolutive combination system is discussed together with a comprehensive account of

polynomial matrices including the definitions and properties. Furthermore, some existing polynomial matrix decomposition techniques are briefly discussed, including polynomial eigenvalue decomposition (PEVD), polynomial singular value decomposition (PSVD), Smith-McMillan decomposition, FIR lossless system decomposition, Lambert's FIR matrix eigenroutine, and approximate polynomial eigenvalue decomposition.

2.2 Matrix Decomposition Techniques for Narrowband Channels

2.2.1 Instantaneous Mixing Model

In an instantaneous mixing model, the transformation from the source signals to the sensors is assumed to be linear, time-invariant (LTI) and instantaneous, and noise is considered to be additive and independent at each sensor.

Given a MIMO system with N sources and M sensors in an instantaneous mixing environment, the transfer function of the channel can be modelled by an $M \times N$ scalar matrix \mathbf{C} , also referred to as the *instantaneous mixing matrix*, in which the elements c_{mn} for $m = 1, \dots, M$ and $n = 1, \dots, N$ represent the channel between the n -th source and m -th sensor. Here c_{mn} is considered to be a complex number which represents a scaling in amplitude caused by the attenuation of propagation and a phase shift that accounts for the propagation delay [7]. This type of MIMO channel is characterised as frequency-flat fading, as the coherence bandwidth¹ of the channel is greater than the transmitted signal bandwidth. This is also the reason it is often called a narrowband MIMO system.

¹Coherence bandwidth is a statistical measurement of the approximate maximum bandwidth or frequency interval over which two frequencies of a signal are likely to experience correlated amplitude fading, and it can be approximately calculated as the inverse of the multipath time delay spread [33].

Considering the propagation of N independent source signals $\mathbf{s}[t] \in \mathbb{C}^{N \times 1}$, where $t \in \{0, 1, \dots, T-1\}$, through an instantaneous mixing system \mathbf{C} , the observed sensor signals $\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$ can be expressed as

$$\mathbf{x}[t] = \mathbf{C}\mathbf{s}[t] + \mathbf{n}[t], \quad t \in \{0, 1, \dots, T-1\}, \quad (2.1)$$

where $\mathbf{n}[t] \in \mathbb{C}^{M \times 1}$ denotes the additive Gaussian noise observed at the receiver with variance of $\sigma^2 \mathbf{I}_M$. In other words, each of the observed sensor signals $x_m[t]$ for $m = 1, \dots, M$ is formulated as a scalar sum of differently weighted source signals plus sensor noise, i.e.,

$$x_m[t] = \sum_{n=1}^N c_{mn} s_n[t] + n_m[t]. \quad (2.2)$$

Figure 2.1 shows a block diagram of a narrowband MIMO system with additive noise. In the context of this thesis, the mixing system \mathbf{C} is assumed to be overdetermined with $M \geq N$.

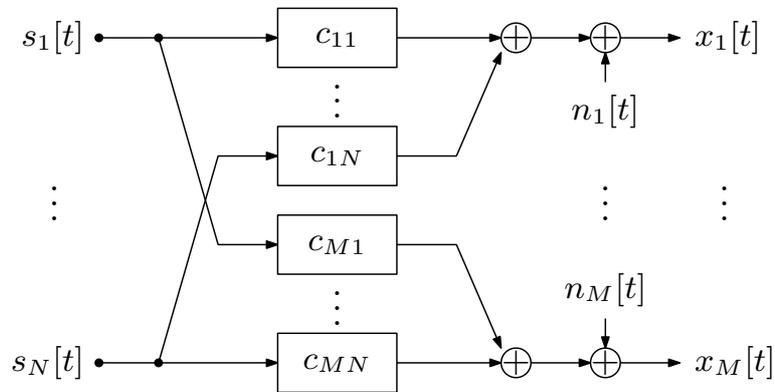


Figure 2.1 Block diagram of a narrowband MIMO system with additive noise.

2.2.2 Blind Source Separation

Blind source separation (BSS) is a process of retrieving the independent source signals from a set of noisy mixtures, such as those described in (2.1), without resorting to any

a priori information about the sources $\mathbf{s}[t]$ and the mixing matrix \mathbf{C} . It utilises only the information carried by the observed sensor signals themselves. Due to the lack of information about the mixing matrix, some additional assumptions on source signals are necessary, and these assumptions may vary in different approaches. Nonetheless, one of the most important assumptions in many approaches is that the source signals are assumed to be statistically independent.² Yellin and Weinstein [36] proved that if the sources are statistically independent, then a necessary and sufficient condition for signal separation is that the outputs are also statistically independent. In other words, separation is deemed successful if the output signals satisfy the independence criterion. This process is also known as independent component analysis (ICA), where higher order statistics (HOS) is often needed to minimize the dependence between the sensor signals. Note that ICA generalises principal component analysis (PCA) to produce independent signals rather than simply uncorrelated signals [4]. For further details of ICA, see [37–40].

BSS has been exploited in a wide variety of applications, such as underwater acoustic signals recorded in passive sonar [41], MIMO communications [42], analysis of astronomical images [43], and interpreting biomedical data from electrocardiogram (ECG) and electroencephalography (EEG) readings [44–50].

Approaches for Instantaneous Blind Source Separation

For the instantaneous mixing model as depicted in Figure 2.1, the problem of BSS is concerned with the existence of matrix \mathbf{W} such that

$$\mathbf{y}[t] = \mathbf{W}\mathbf{x}[t], \quad (2.3)$$

where the data vector $\mathbf{y}[t]$ denotes the outputs of the separation system and \mathbf{W} is an unknown separation matrix. For the sake of simplicity, we assume that the number

²Statistical independence means that given one of the source signals, nothing can be predicted or estimated about any other source signals.

of sensors is the same as the number of sources. Successful separation of the sources requires that the components in $\mathbf{y}[t]$ are as mutually independent as possible, thereby providing an estimate of the source signals $\mathbf{s}[t]$. If the effect of the noise is ignored, the source signals can be identified by requiring $\mathbf{WC} = \mathbf{I}$, where the mixing matrix \mathbf{C} can be estimated by methods in [51, 52]. In other words, the separation system is the inverse of the mixing system.

There exist numerous algorithms for solving the instantaneous BSS problem; the majority of these algorithms implement a two-stage approach. In the first stage, the sensor outputs are decorrelated and normalised using a PCA method, such as EVD or SVD, that exploits second order statistics (SOS) of the sensor signals. This stage is also known as whitening. Then, the second stage uses HOS to minimise the dependence of the whitened signals to obtain estimates of the source signals [53]. It is beyond the scope of this thesis to discuss further details of the different algorithms for instantaneous BSS. However, some of the best known algorithms, which are based on this two-stage approach, include Joint Approximation Diagonalisation of Eigenmatrices (JADE) [54], BLInd Signal Separation (BLISS) [55, 56], FastICA [40], and KernelICA [57]. All these algorithms use the same second order stage but different methods for higher order stages. In [4], Pope and Bogner provide a detailed review on instantaneous BSS.

The main objective of this section is to highlight the importance and value of the two scalar matrix decomposition techniques (EVD and SVD) in relation to the instantaneous mixing model. This also laid the groundwork for the more complicated convolutive mixing model and the corresponding polynomial matrix decomposition techniques.

2.2.3 Eigenvalue Decomposition

The eigenvalue decomposition (EVD) of a Hermitian matrix $\mathbf{R} \in \mathbb{C}^{M \times M}$ is defined as

$$\mathbf{R}\mathbf{q}_i = d_i\mathbf{q}_i, \quad i = 1, \dots, M, \quad (2.4)$$

where $d_i \in \mathbb{R}$, $d_i \geq 0$ represents the eigenvalues of \mathbf{R} , and $\mathbf{q}_i \in \mathbb{C}^{M \times 1}$ is the corresponding eigenvectors satisfying $\mathbf{q}_i^H \mathbf{q}_j = \delta(i - j)$ for $i, j = 1, \dots, M$. In general, the eigenvectors \mathbf{q}_i can also have arbitrary phase α , such that

$$\mathbf{R}\mathbf{q}_i e^{j\alpha} = d_i \mathbf{q}_i e^{j\alpha}. \quad (2.5)$$

The above equation can also be expressed as the following matrix representation

$$\mathbf{R} = \mathbf{Q}\mathbf{D}\mathbf{Q}^H, \quad (2.6)$$

where $\mathbf{D} \in \mathbb{R}^{M \times M}$ is a diagonal matrix, and it is unique when the eigenvalues d_i for $i = 1, \dots, M$ are ordered. $\mathbf{Q} \in \mathbb{C}^{M \times M}$ is a unitary matrix whose columns are the orthonormal eigenvectors of \mathbf{R} , and it satisfies $\mathbf{Q}\mathbf{Q}^H = \mathbf{I}$. Note that \mathbf{Q} is ambiguous as a result of the arbitrary phase α in (2.5).

The EVD diagonalises the Hermitian matrix \mathbf{R} by using a unitary or orthogonal transformation \mathbf{Q} , so the total energy has been preserved such that $\|\mathbf{R}\|_{\mathbb{F}}^2 = \|\mathbf{D}\|_{\mathbb{F}}^2$, where the notation $\|\cdot\|_{\mathbb{F}}$ represents the Frobenius norm. The EVD exists for every Hermitian matrix \mathbf{R} , and it does not matter if its entries are real or complex. One of the earliest algorithms for computing the EVD of a Hermitian matrix was the Jacobi algorithm [8] which iteratively reduces the matrix into a diagonal form by using a unitary (orthogonal) transformation. The implementation of the Jacobi algorithm is simple but ineffective, as it performs operations across all the off-diagonal elements in \mathbf{R} . Most of the current efficient algorithms preliminarily reduce the real symmetric (or complex Hermitian) matrix \mathbf{R} to a tridiagonal matrix \mathbf{T} by using a non-iterative algorithm in

a finite number of steps and then work with the tridiagonal matrix based on QR (or QL) iterations, such as using the xSTEQR computational subroutine from the LAPACK library [58]. Discussions of the specific algorithms are beyond the scope of this thesis; a detailed review of eigenvalue computation can be found in [59, 60].

The EVD has been extensively used in many areas of DSP. For example, it acts as an important tool in the Karhunen-Loeve transform for optimal data compaction [61]. It also plays a fundamental role in PCA for most narrowband BSS problems [4]. In the context of this thesis, we will now demonstrate how the EVD can be used to decorrelate a set of instantaneously mixed signals, which is considered the first stage of the two-stage BSS approach [49].

Decorrelation of Instantaneously Mixed Signals

The following assumptions are made when using the EVD to decorrelate instantaneously mixed signals:

1. The observed sensor signals $\mathbf{x}[t]$ from (2.1) are assumed to be zero mean and stationary;
2. The mixing matrix \mathbf{C} is assumed to be time invariant; and
3. The sensor noise $\mathbf{n}[t]$ is assumed to be white Gaussian noise and uncorrelated between the sensors.

Thus, the covariance matrix of the sensor signals $\mathbf{x}[t]$ can be calculated as

$$\begin{aligned}
 \mathbf{R}_{xx} &= E\{\mathbf{x}[t]\mathbf{x}^H[t]\} = E\{\mathbf{C}\mathbf{s}[t](\mathbf{C}\mathbf{s}[t])^H\} + \sigma^2\mathbf{I}_M \\
 &= \mathbf{C}E\{\mathbf{s}[t]\mathbf{s}^H[t]\}\mathbf{C}^H + \sigma^2\mathbf{I}_M \\
 &= \mathbf{C}\mathbf{R}_{ss}\mathbf{C}^H + \sigma^2\mathbf{I}_M,
 \end{aligned} \tag{2.7}$$

where $E\{\cdot\}$ denotes the expectation operator and $\mathbf{R}_{ss} = E\{\mathbf{s}[t]\mathbf{s}^H[t]\}$ is the spatial covariance matrix of the source signals [62]. As the source signals $\mathbf{s}[t]$ in (2.1) are assumed to be statistically independent, there is no cross-correlation between pairs of

source signals, i.e., $E\{s_k[t]s_l^*[t]\} = 0, k \neq l, \forall t$. Therefore, \mathbf{R}_{ss} is a diagonal matrix. However, the covariance matrix \mathbf{R}_{xx} will generally not be diagonal, because the observed signals $\mathbf{x}[t]$ comprise a linear combination of the source signals and are therefore correlated with one another.

To demonstrate the process of decorrelating the sensor signals using the EVD, we define a matrix

$$\mathbf{X} = [\mathbf{x}[0], \mathbf{x}[1], \dots, \mathbf{x}[T-1]], \quad (2.8)$$

which contains the sensor output samples $\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$ for $t = 0, \dots, T-1$. The covariance matrix of the sensor signals can then be estimated as

$$\hat{\mathbf{R}}_{xx} = \frac{\mathbf{X}\mathbf{X}^H}{T}. \quad (2.9)$$

As $\hat{\mathbf{R}}_{xx}$ is Hermitian, it can be diagonalised using the EVD in (2.6) such that

$$\mathbf{Q}^H \hat{\mathbf{R}}_{xx} \mathbf{Q} = \mathbf{D}, \quad (2.10)$$

where $\mathbf{Q} \in \mathbb{C}^{M \times M}$ denotes a unitary matrix and $\mathbf{D} \in \mathbb{C}^{M \times M}$ is the diagonalised covariance matrix whose elements d_{mm} for $m = 1, \dots, M$ are generally ordered to produce a unique solution.

Following the diagonalisation of the covariance matrix, the unitary matrix \mathbf{Q}^H is applied to the data matrix \mathbf{X} , resulting in the transformed data matrix

$$\mathbf{Y} = \mathbf{Q}^H \mathbf{X}. \quad (2.11)$$

This process is known as decorrelation (or whitening), which removes the spatial correlation between the sensors. The transformed signals $\mathbf{y}[t]$ for $t = 0, \dots, T-1$, with an estimated covariance matrix $\hat{\mathbf{R}}_{yy} = \mathbf{D}$, represent the decorrelated signals. Note that the EVD-based PCA, which only considers the SOS, can only decorrelate the signals and therefore is not generally sufficient for separation (separation requires a stronger

condition of independence of the signals). Therefore, HOS are needed to minimise the dependence between the signals, which can be implemented by the ICA algorithms, such as JADE [54] and FastICA [63]. Source separation using HOS requires that at most one of the source signals has a Gaussian distribution [49, 64]. However, a work by Gerven and Compernelle [65] shows that two source signals can be separated by decorrelation if the mixing system is minimum phase. Furthermore, if the majority of the sensor signals have very different power levels, then source separation can be generally achieved by using the EVD or SVD [7].

2.2.4 Singular Value Decomposition

Compared to the EVD, which only works on Hermitian (or symmetric) matrices, the SVD applies to all matrices. It plays a fundamental role in matrix computation and analysis [8]. The SVD of an arbitrary matrix $\mathbf{A} \in \mathbb{C}^{M \times N}$, where M does not have to be the same as N , is defined as

$$\mathbf{A} = \mathbf{U}^H \mathbf{\Sigma} \mathbf{V}, \quad (2.12)$$

where $\mathbf{U} \in \mathbb{C}^{M \times M}$ and $\mathbf{V} \in \mathbb{C}^{N \times N}$ are both unitary matrices, such that $\mathbf{U}\mathbf{U}^H = \mathbf{U}^H\mathbf{U} = \mathbf{I}_M$ and $\mathbf{V}\mathbf{V}^H = \mathbf{V}^H\mathbf{V} = \mathbf{I}_N$. $\mathbf{\Sigma} \in \mathbb{R}^{M \times N}$ is a diagonal matrix, i.e., $\mathbf{\Sigma} = \text{diag}\{\sigma_{11}, \dots, \sigma_{nn}\}$, where $n = \min\{M, N\}$. The diagonal elements of $\mathbf{\Sigma}$ represent the singular values of the matrix \mathbf{A} and satisfy $\sigma_{11} \geq \sigma_{22} \geq \dots \geq \sigma_{nn}$. Due to the unitary transformation, the SVD is also norm preserving as with EVD, i.e., $\|\mathbf{A}\|_{\text{F}}^2 = \|\mathbf{\Sigma}\|_{\text{F}}^2$.

Narrowband MIMO Channel Equalisation

In a narrowband MIMO system, as depicted in Figure 2.1, the SVD is a popular strategy for addressing the channel equalisation problem [33, 66]. If the mixing matrix \mathbf{C} is known at both the transmitter and the receiver, the SVD can be used to diagonalise the channel matrix, i.e., $\mathbf{C} = \mathbf{U}^H \mathbf{\Sigma} \mathbf{V}$. The produced unitary matrices \mathbf{V}^H and \mathbf{U} are then

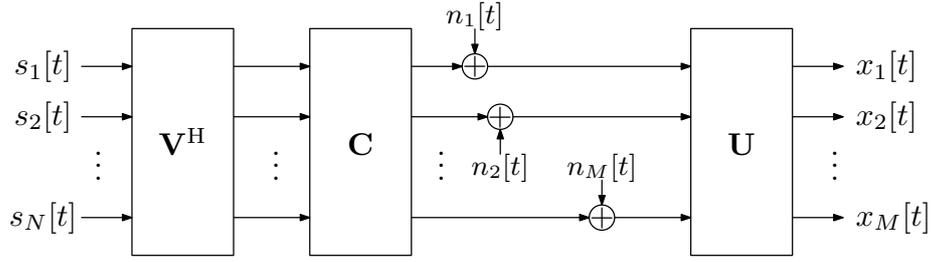


Figure 2.2 A block diagram of the MIMO communication system using the SVD-based equalisation scheme.

respectively applied at the transmitter and receiver, such that

$$\mathbf{x}[t] = \mathbf{U}\mathbf{C}\mathbf{V}^H\mathbf{s}[t] + \mathbf{U}\mathbf{n}[t]. \quad (2.13)$$

By substituting $\mathbf{C} = \mathbf{U}^H\mathbf{\Sigma}\mathbf{V}$ into (2.13), we have

$$\mathbf{x}[t] = \mathbf{\Sigma}\mathbf{s}[t] + \mathbf{n}'[t], \quad (2.14)$$

where $\mathbf{n}'[t] = \mathbf{U}\mathbf{n}[t]$. Figure 2.2 shows a block diagram of the MIMO communication system based on the SVD. As both matrices \mathbf{U} and \mathbf{V} are unitary, the transmit power is not increased, nor is the channel noise enhanced. In effect, the SVD transfers the instantaneous MIMO channel into a number of independent parallel subchannels for which co-channel interference (CCI) no longer exists.

The unitary matrices \mathbf{U}^H and \mathbf{V} can be obtained from the decomposition of the two matrices $\mathbf{C}^H\mathbf{C}$ and $\mathbf{C}\mathbf{C}^H$. By separately pre- and post-multiplying the channel matrix \mathbf{C} with its Hermitian transpose, we obtain the following pair of equations.

$$\begin{aligned} \mathbf{C}^H\mathbf{C} &= \mathbf{V}^H\mathbf{\Sigma}^H\mathbf{\Sigma}\mathbf{V} \\ \mathbf{C}\mathbf{C}^H &= \mathbf{U}^H\mathbf{\Sigma}\mathbf{\Sigma}^H\mathbf{U} \end{aligned} \quad (2.15)$$

In fact, $\mathbf{C}^H\mathbf{C}$ and $\mathbf{C}\mathbf{C}^H$ are Hermitian matrices by construction, so the above equations form two EVDs, where $\mathbf{\Sigma}^H\mathbf{\Sigma} \in \mathbb{C}^{N \times N}$ and $\mathbf{\Sigma}\mathbf{\Sigma}^H \in \mathbb{C}^{M \times M}$ are the diagonal matrices.

The SVD can also be applied to the PCA for whitening a set of instantaneously mixed signals; however, this is different from the EVD-based PCA in which the data covariance matrix is diagonalised. Instead, the SVD is directly applied to the sensor data. For this reason, the SVD-based PCA is less computationally expensive than the EVD-based PCA. For further details, see [38, 67].

2.3 Matrix Decomposition Techniques for Broadband Channels

Channels

2.3.1 Background to Polynomial Matrices

A polynomial matrix is simply a matrix with polynomial entries; alternatively, it can be seen as a polynomial with matrix-valued coefficients [68, 69]. Polynomial matrices are represented in terms of the indeterminate variable z^{-1} , which can also be seen as a unit delay in the context of this thesis. For an $M \times N$ polynomial matrix $\underline{\mathbf{A}}(z)$, it can be expressed as

$$\underline{\mathbf{A}}(z) = \sum_{\tau=T_1}^{T_2} \mathbf{A}[\tau]z^{-\tau} = \begin{bmatrix} \underline{a}_{11}(z) & \underline{a}_{12}(z) & \cdots & \underline{a}_{1N}(z) \\ \underline{a}_{21}(z) & \underline{a}_{22}(z) & \cdots & \underline{a}_{2N}(z) \\ \vdots & \vdots & \ddots & \vdots \\ \underline{a}_{M1}(z) & \underline{a}_{M2}(z) & \cdots & \underline{a}_{MN}(z) \end{bmatrix}, \quad (2.16)$$

where $\tau \in \mathbb{Z}$ and $T_2 \geq T_1$. The *order* of this polynomial matrix is given by $T_2 - T_1$, such that $\mathbf{A}[T_1] \neq \mathbf{0}$ and $\mathbf{A}[T_2] \neq \mathbf{0}$. The coefficient matrices $\mathbf{A}[\tau]$, $\tau \in [T_1, T_2]$ generally contains complex scalar entries, and $\mathbf{A}[\tau]$ is referred to as the coefficient matrix at the discrete time index τ . In particular, $\mathbf{A}[0]$ represents the coefficient matrix at the zero lag, which is very important for the discussion of the PEVD algorithms throughout this thesis.

Special Polynomial Matrices and their Properties

Given a polynomial matrix $\underline{\mathbf{A}}(z)$, its *paraconjugate* $\tilde{\underline{\mathbf{A}}}(z)$ can be calculated as

$$\tilde{\underline{\mathbf{A}}}(z) = \underline{\mathbf{A}}^H(z^{-1}), \quad (2.17)$$

where the paraconjugate operator $\{\tilde{\cdot}\}$ implies performing the Hermitian conjugate for the coefficient matrices $\mathbf{A}[\tau]$, $\forall \tau$, and time-reversing all entries inside.

Analogue to the notion of Hermitian for a scalar matrix, a polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{M \times M}$ is *para-Hermitian* if it is equal to its paraconjugate, i.e.,

$$\underline{\mathbf{A}}(z) = \tilde{\underline{\mathbf{A}}}(z). \quad (2.18)$$

Hence, the individual coefficients $a_{jk}[\tau]$, for $j, k = 1, 2, \dots, M$ associated with the polynomial matrix $\underline{\mathbf{A}}(z)$ satisfy $a_{jk}[\tau] = a_{kj}^*[-\tau]$, $\forall \tau \in \mathbb{Z}$.

A polynomial matrix $\underline{\mathbf{A}}(z)$ is said to be *paraunitary* if it follows

$$\underline{\mathbf{A}}(z)\tilde{\underline{\mathbf{A}}}(z) = \tilde{\underline{\mathbf{A}}}(z)\underline{\mathbf{A}}(z) = \mathbf{I}. \quad (2.19)$$

A more general definition in [69] states that $\underline{\mathbf{A}}(z)$ is a paraunitary matrix if it satisfies $\underline{\mathbf{A}}(z)\tilde{\underline{\mathbf{A}}}(z) = c^2\mathbf{I}$, where c^2 is a constant coefficient. However, we will only consider the case when $c^2 = 1$, so the paraunitary definition in (2.19) will be adopted in this thesis.

Finally, the *Frobenius norm* of the polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{M \times N}$ is defined as

$$\begin{aligned} \|\underline{\mathbf{A}}(z)\|_F &= \sqrt{\sum_{\tau=T_1}^{T_2} \sum_{m=1}^M \sum_{n=1}^N |a_{mn}[\tau]|^2} \\ &= \sqrt{\text{trace}\{[\underline{\mathbf{A}}(z)\tilde{\underline{\mathbf{A}}}(z)]|_0\}}, \end{aligned} \quad (2.20)$$

where $[\cdot]|_0$ denotes the zero-lag (z^0) coefficient matrix of the polynomial matrix.

2.3.2 Convolutional Mixing Model

The assumption of an instantaneous mixing model as described in 2.2.1 is unrealistic in many real-world applications, as it fails to accommodate cases where the transmission channels have multipath effects and delays. The mixing process for real-world applications, such as in an audio separation system, is more complex. In such systems, the observed sensor signals are differently weighted and delayed, and each source signal contributes to the sum with different time delays corresponding to multiple paths by which an acoustic signal propagates to a microphone. This means that the source signals are filtered rather than simply scaled as they propagate to the sensors. Such a filtering process is referred to as *convolutional mixing*, in which the bandwidth of the transmitted signals or source signals are greater than the coherence bandwidth of the channel, and for this reason, systems of this type are often called broadband sensor array systems.

It is assumed that N independent source signals organised in a vector $\mathbf{s}[t] \in \mathbb{C}^{N \times 1}$ for $t \in \{0, \dots, T-1\}$ propagate through a convolutional channel. The received signals, denoted $\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$, can then be expressed as

$$\mathbf{x}[t] = \sum_{l=0}^L \mathbf{C}[l] \mathbf{s}[t-l] + \mathbf{n}[t], \quad t \in \{0, \dots, T-1\}, \quad (2.21)$$

where $\mathbf{n}[t] \in \mathbb{C}^{M \times 1}$ is an additive Gaussian noise process with variance $\sigma^2 \mathbf{I}_M$ and $\mathbf{C}[l] \in \mathbb{C}^{M \times N}$ for $l \in \{0, \dots, L\}$ represent the coefficient matrices of the convolutional (polynomial) mixing matrix $\underline{\mathbf{C}}(z)$, i.e.,

$$\underline{\mathbf{C}}(z) = \sum_{l=0}^L \mathbf{C}[l] z^{-l}. \quad (2.22)$$

Each of the polynomial entries $\underline{c}_{mn}(z)$ of $\underline{\mathbf{C}}(z)$, where $m = 1, \dots, M$ and $n = 1, \dots, N$, can be seen as an FIR filter between the n -th source and m -th sensor. A block diagram of this convolutional MIMO system is depicted in Figure 2.3.

A more compact expression can be obtained by taking the z transform of (2.21), such that

$$\underline{\mathbf{x}}(z) = \underline{\mathbf{C}}(z)\underline{\mathbf{s}}(z) + \underline{\mathbf{n}}(z) , \quad (2.23)$$

where $\underline{\mathbf{x}}(z) = \sum_t \mathbf{x}[t]z^{-t}$, $\underline{\mathbf{s}}(z) = \sum_t \mathbf{s}[t]z^{-t}$, and $\underline{\mathbf{n}}(z) = \sum_t \mathbf{n}[t]z^{-t}$ denote the algebraic power series of the sensor signals, the source signals, and the noise, respectively.

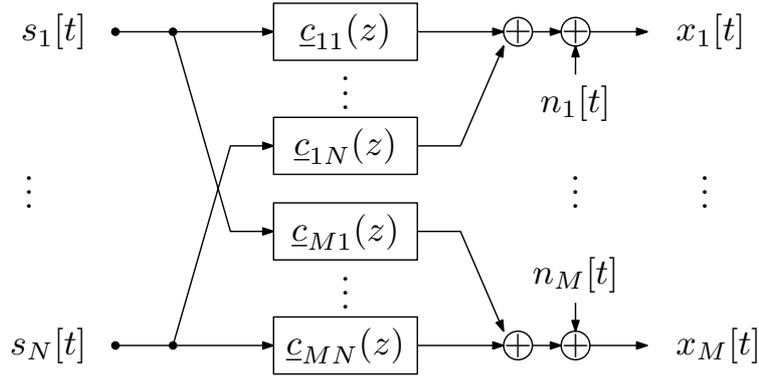


Figure 2.3 Block diagram of a broadband MIMO system consisting of a number of FIR channels.

2.3.3 Polynomial Eigenvalue Decomposition

The conventional EVD algorithm is only suitable for diagonalising the covariance matrix of narrowband signals. When the problem is extended to broadband scenarios, as shown in Figure 2.3, PEVD techniques need to be taken into account. Assuming that the convolutively mixed signals $\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$ have zero mean, the space-time covariance matrix

$$\mathbf{R}[\tau] = E \{ \mathbf{x}[t] \mathbf{x}^H[t - \tau] \} , \quad t \text{ and } \tau \in \mathbb{Z} \quad (2.24)$$

represents the correlation between pairs of signals sampled at a time instant τ . The corresponding cross spectral density (CSD) matrix $\underline{\mathbf{R}}(z)$ is a polynomial matrix and can be obtained by taking the z -transform of (2.24), i.e.,

$$\underline{\mathbf{R}}(z) = \sum_{\tau=-T}^T \mathbf{R}[\tau] z^{-\tau} , \quad (2.25)$$

where $\underline{\mathbf{R}}(z)$ is para-Hermitian matrix, i.e., $\underline{\mathbf{R}}(z) = \tilde{\underline{\mathbf{R}}}(z)$. To decorrelate the convolatively mixed signals, which corresponds to eliminating the cross-correlations between different sensors over all time delays, PEVD has been proposed [7]. This takes the form

$$\underline{\mathbf{H}}(z)\underline{\mathbf{R}}(z)\tilde{\underline{\mathbf{H}}}(z) \approx \underline{\mathbf{D}}(z) \quad (2.26)$$

where $\underline{\mathbf{H}}(z) \in \mathbb{C}^{M \times M}$ is a paraunitary matrix, i.e., $\underline{\mathbf{H}}(z)\tilde{\underline{\mathbf{H}}}(z) = \tilde{\underline{\mathbf{H}}}(z)\underline{\mathbf{H}}(z) = \mathbf{I}_M$, and $\underline{\mathbf{D}}(z) \in \mathbb{C}^{M \times M}$ is (ideally) a diagonal matrix. Here, the paraunitary matrix $\underline{\mathbf{H}}(z)$ can be seen as a multichannel all-pass filter, which preserves the energy of the combined signals [69]. A detailed discussion of the existing PEVD algorithms will be presented in Chapter 3.

The PEVD techniques have attracted significant interest in digital signal processing and communications over the past few years. Applications of PEVD have been found in areas such as decorrelation of the signals received by broadband sensor arrays [7], estimation of broadband angle of arrival [12, 70], subband coding [21], precoding and equalisation for MIMO communications [34], multichannel spectral factorisation [18, 19], and convolutive BSS [23, 24, 71] among others. In this thesis, two potential applications of the proposed PEVD algorithm will be presented in Chapters 6 and 7.

2.3.4 Polynomial Singular Value Decomposition

The PSVD of the polynomial channel $\underline{\mathbf{C}}(z) \in \mathbb{C}^{M \times N}$ is given by

$$\underline{\mathbf{C}}(z) = \tilde{\underline{\mathbf{U}}}(z)\underline{\Sigma}(z)\underline{\mathbf{V}}(z), \quad (2.27)$$

where $\underline{\mathbf{U}}(z) \in \mathbb{C}^{M \times M}$ and $\underline{\mathbf{V}}(z) \in \mathbb{C}^{N \times N}$ are paraunitary matrices, such that $\underline{\mathbf{U}}(z)\tilde{\underline{\mathbf{U}}}(z) = \tilde{\underline{\mathbf{U}}}(z)\underline{\mathbf{U}}(z) = \mathbf{I}_M$ and $\underline{\mathbf{V}}(z)\tilde{\underline{\mathbf{V}}}(z) = \tilde{\underline{\mathbf{V}}}(z)\underline{\mathbf{V}}(z) = \mathbf{I}_N$. $\underline{\Sigma} \in \mathbb{R}^{M \times N}$ is a diagonal polynomial matrix.

In analogy to how the SVD can be used to address the channel equalisation problem for narrowband MIMO systems, the effectiveness of the PSVD will be demonstrated

by a potential application of broadband MIMO channel equalisation in the penultimate chapter.

2.4 Other Polynomial Matrix Decomposition Techniques

2.4.1 Smith Form

Given a generic $M \times N$ polynomial matrix $\underline{\mathbf{G}}(z)$, its Smith form, i.e., simpler forms such as diagonal, upper, or lower triangular polynomial matrices, can be obtained by means of elementary row and column operations [69, 72]. Similar to the elementary operations for scalar matrices [8], there exist three elementary operations for a polynomial matrix, i.e.,

- multiplying a row or column by a non-zero constant;
- interchanging two rows or two columns; and
- adding a polynomial multiple of a row or column to another row or column.

These operations are performed by either pre-multiplying or post-multiplying $\underline{\mathbf{G}}(z)$ with the corresponding elementary matrices, which are known as unimodular polynomial matrices. A polynomial matrix is unimodular if its determinant is a non-zero constant; the inverse of a unimodular matrix remains unimodular. Pre-multiplication of $\underline{\mathbf{G}}(z)$ by an elementary matrix corresponds to a row operation, while post-multiplication indicates a column operation.

2.4.2 Smith-McMillan Form

The Smith-McMillan form is probably the most well known polynomial matrix diagonalisation method. For a broadband MIMO channel matrix $\underline{\mathbf{C}}(z) \in \mathbb{C}^{M \times N}$, if there exists a set of elementary polynomial matrices $\underline{\mathbf{P}}_i(z) \in \mathbb{C}^{M \times M}$ for $i = 1, \dots, K_1$ and

$\underline{\mathbf{Q}}_j(z) \in \underline{\mathbb{C}}^{N \times N}$ for $j = 1, \dots, K_2$, such that

$$\underline{\mathbf{P}}_{K_1}(z) \cdots \underline{\mathbf{P}}_2(z) \underline{\mathbf{P}}_1(z) \underline{\mathbf{C}}(z) \underline{\mathbf{Q}}_1(z) \underline{\mathbf{Q}}_2(z) \cdots \underline{\mathbf{Q}}_{K_2}(z) = \underline{\mathbf{\Lambda}}(z), \quad (2.28)$$

then these two polynomial matrices $\underline{\mathbf{C}}(z)$ and $\underline{\mathbf{\Lambda}}(z)$ are said to be equivalent. By choosing suitable $\underline{\mathbf{P}}_i(z)$ and $\underline{\mathbf{Q}}_j(z)$, the polynomial matrix $\underline{\mathbf{C}}(z)$ can be transformed into an $M \times N$ diagonal matrix $\underline{\mathbf{\Lambda}}(z)$, which is called the Smith-McMillan form of $\underline{\mathbf{C}}(z)$. Thus, (2.28) can be rewritten as

$$\underline{\mathbf{U}}(z) \underline{\mathbf{C}}(z) \underline{\mathbf{V}}(z) = \underline{\mathbf{\Lambda}}(z), \quad (2.29)$$

where $\underline{\mathbf{U}}(z) = \prod_{i=1}^{K_1} \underline{\mathbf{P}}_i(z)$, and $\underline{\mathbf{V}}(z) = \prod_{j=1}^{K_2} \underline{\mathbf{Q}}_j(z)$. As the product of any two unimodular matrices is also unimodular, $\underline{\mathbf{U}}(z)$ and $\underline{\mathbf{V}}(z)$ are both unimodular matrices. Note that the matrices $\underline{\mathbf{U}}(z)$ and $\underline{\mathbf{V}}(z)$ do not necessarily possess the paraunitary property, i.e., $\underline{\mathbf{U}}(z)\tilde{\underline{\mathbf{U}}}(z) \neq \mathbf{I}_M$ and $\underline{\mathbf{V}}(z)\tilde{\underline{\mathbf{V}}}(z) \neq \mathbf{I}_N$, so this is quite distinct from the polynomial EVD or SVD techniques proposed in this thesis.

In broadband MIMO systems, the Smith-McMillan decomposition can be used to design the transmit and receive filter banks to produce parallel sub-channels, where the interference between two different channels is removed. However, as the resulting polynomial filter banks are not paraunitary (or lossless), the noise power is no longer preserved when using $\tilde{\underline{\mathbf{U}}}(z)$ as the receive filter bank. Also, the noise is no longer additive white Gaussian, which may bring further challenges for equalisation and decoding schemes [73]. The Smith-McMillan decomposition is also widely used to determine the poles and zeros of the transfer matrix of a broadband MIMO system [74, 75].

2.4.3 FIR Lossless System Decomposition

In the signal processing literature, a lossless FIR system which satisfies causality and stability can be described by a paraunitary matrix with finite degree. Vaidyanathan [69] has shown that any finite degree paraunitary matrix can be factorised into a set of paraunitary matrices consisting of delay and rotation matrices. At each step of this

2.4 Other Polynomial Matrix Decomposition Techniques

process, an elementary delay and a Givens rotation matrix [8] are factorised out of the FIR lossless system $\underline{\mathbf{H}}_L(z)$ (of degree L), which results in a lossless (paraunitary) system whose degree, L , has been reduced by unity [67].

For example, for a 2×2 lossless FIR system $\underline{\mathbf{H}}_L(z) = \sum_{l=0}^L \mathbf{H}[l]z^{-l}$, where $\mathbf{H}[l] \in \mathbb{R}^{2 \times 2}$ and its determinant $\det\{\underline{\mathbf{H}}_L(z)\} = cz^{-L}$, where $c \neq 0$, the first step of this decomposition is to find an elementary delay matrix $\underline{\Delta}(z)$ and a Givens rotation matrix \mathbf{Q}_L such that

$$\underline{\mathbf{H}}_L(z) = \underbrace{\begin{bmatrix} \cos(\theta_L) & \sin(\theta_L) \\ -\sin(\theta_L) & \cos(\theta_L) \end{bmatrix}}_{\mathbf{Q}_L} \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & z^{-1} \end{bmatrix}}_{\underline{\Delta}(z)} \underline{\mathbf{H}}_{L-1}(z), \quad (2.30)$$

where $\underline{\mathbf{H}}_{L-1}(z)$ denotes a lossless FIR system whose determinant degree has been reduced by unity. This results in the degree of the whole system being reduced. Following on this, a lossless degree-one block has been extracted to obtain $\underline{\mathbf{H}}_{L-1}(z)$. This process is repeated until a degree-zero block is obtained. Thus, $\underline{\mathbf{H}}_L(z)$ can be factorised as

$$\underline{\mathbf{H}}_L(z) = \mathbf{Q}_L \underline{\Delta}(z) \mathbf{Q}_{L-1} \underline{\Delta}(z) \cdots \mathbf{Q}_1 \underline{\Delta}(z) \mathbf{Q}_0 \begin{bmatrix} \alpha & 0 \\ 0 & \pm\alpha \end{bmatrix}, \quad (2.31)$$

where $\alpha \neq 0$ is included to account for the ambiguity in this decomposition [69, 67].

Vaidyanathan [69] has generalised this 2×2 paraunitary matrix to paraunitary matrices of any dimension, provided that they are of fixed degree. The drawback of this decomposition method is that it only performs a unit delay at each step, which is not efficient when the paraunitary matrix to be factorised is of a high degree but most of the coefficient matrices are zero-valued. For example, if the paraunitary matrix is of degree 10, but contains only two non-zero coefficient matrices ($\mathbf{H}[0]$ and $\mathbf{H}[10]$), this decomposition would require ten steps, which is computationally inefficient. However, for the PEVD algorithms discussed in Chapters 3 and 4, this would not be a problem.

This FIR lossless decomposition method was adopted by Regalia and Huang [76] to obtain a two-channel lossless FIR filter for optimal data compaction. In effect, this

leads to an optimal paraunitary matrix which can be used to compute the PEVD of a 2×2 matrix. Although they re-formulated the problem using a state space approach and developed an iteration method which avoids the problems of local minima associated with gradient descent techniques, their approach still suffers from the inherent inefficiency of the unit delay step in the FIR lossless decomposition method and is only applicable to the 2×2 case.

2.4.4 Lambert's Approach for Multichannel Blind Deconvolution

In [77, 78], Lambert proposed a solution to address the problem of convolutive BSS. By generalising some standard linear algebra and control techniques from the complex number field to the field of rational functions, he developed a method to compute the EVD of FIR polynomial matrices. The eigenvalue routine adopted in his work is based on Householder reflections [8]. Specifically, his method transforms the polynomial mixing matrix to the frequency domain using the discrete Fourier transform (DFT) and involves the approximate inversion of the FIR filters in the frequency domain. For this reason, his method is very different from the PEVD algorithms discussed in this thesis.

2.4.5 Approximate Polynomial Eigenvalue Decomposition

In [79], Tkacenko proposed a solution for computing the approximate polynomial EVD of para-Hermitian matrices. His method uses Householder reflections to successively construct first order FIR paraunitary transformations as required for annihilating the off-diagonal energy. As all off-diagonal energy must be brought onto the coefficient matrix of order one before it can be transferred onto the diagonal, it does not seem to converge as the number of iterations increases. This may lead to a poor approximation in some cases.

Similar to Tkacenko's approach, an approximate PEVD algorithm developed by Redif et al. [80] exploits Householder-like paraunitary matrices and the same parame-

ter optimisation schemes; however, the essential difference is that Redif's method has the freedom to search the entire polynomial matrix for the maximum off-diagonal energy. In this respect, this is much like the second order sequential best rotation (SBR2) algorithm [7], which will be reviewed in the next chapter.

2.5 Chapter Summary

This chapter has provided a brief overview of scalar matrix decomposition techniques, including EVD and SVD, and has discussed some relevant applications of these techniques in the context of narrowband sensor array processing, where the sensor signals are instantaneously mixed. Following on that, the convolutive mixing model was introduced, where the sensor signals are correlated with one another not only at the same time instant but also over a range of time delays due to the multipath effect. Subsequently, some existing techniques for addressing the broadband sensor array processing were briefly discussed. In particular, the idea of the polynomial EVD was introduced in terms of diagonalising the space-time covariance matrix. The next chapter will primarily focus on further discussions of the existing algorithms for calculating the PEVD.

Chapter 3

Iterative Algorithms for Polynomial Eigenvalue Decomposition

3.1 Introduction

The idea of polynomial eigenvalue decomposition (PEVD) was generalised from the conventional EVD for para-Hermitian matrices, and it was initially developed as the preliminary (second order) stage of a multi-stage convolutive blind source separation (BSS) method [22]. However, the PEVD algorithms can also be used alone as a time-domain approach for decorrelating convolutively mixed signals received from a set of sensors [7, 27] and for identifying signal and noise subspaces, also known as broadband subspace decomposition [12, 17, 81].

This chapter aims to investigate the state-of-the-art algorithmic developments for PEVD and provides a comprehensive discussion of existing PEVD algorithms from the following two perspectives: (i) what algorithms are available for computing PEVD and (ii) how they work differently for diagonalising para-Hermitian matrices. We begin by answering basic questions concerning under what circumstances a PEVD exists and whether the solution is unique. Some generalities of the PEVD algorithms are then introduced. Following on that, the most established PEVD algorithm, known as

second order sequential best rotation (SBR2) [7], is discussed. In particular, we take a fresh look at the SBR2 algorithm in terms of its potential for optimising the subband coding gain. As a result, the spectral majorisation property can be proved.

Apart from the SBR2 algorithm, other existing PEVD algorithms, including the sequential matrix diagonalisation (SMD), maximum element SMD (ME-SMD), and multiple shift ME-SMD (MSME-SMD) algorithms, are also briefly discussed. In addition, polynomial order truncation methods are introduced to shorten the unnecessarily large order of the resulting polynomial matrices obtained from the PEVD algorithms.

3.2 Generality of Iterative PEVD Algorithms

This section first discusses the uniqueness and ambiguity of a para-Hermitian matrix EVD problem. Then, some general definitions of existing PEVD algorithms are introduced in order to facilitate a detailed discussion in the following sections.

3.2.1 PEVD Uniqueness and Ambiguity

To clarify the problem, we can rewrite the PEVD expression of the cross spectral density (CSD) matrix $\underline{\mathbf{R}}(z) \in \underline{\mathbb{C}}^{M \times M}$ in (2.26) as

$$\underline{\mathbf{R}}(z) \approx \underline{\tilde{\mathbf{H}}}(z)\underline{\mathbf{D}}(z)\underline{\mathbf{H}}(z). \quad (3.1)$$

Icart and Comon claim in [82] that such decomposition at least exists in approximation with the paraunitary matrix $\underline{\mathbf{H}}(z)$ of sufficiently high order. Assuming the PEVD in (3.1) holds with equality, we want to find out whether a second decomposition

$$\underline{\mathbf{R}}(z) = \underline{\tilde{\tilde{\mathbf{H}}}}(z)\underline{\tilde{\mathbf{D}}}(z)\underline{\tilde{\mathbf{H}}}(z) \quad (3.2)$$

exists.

To answer the above question, we consider $\underline{\mathbf{R}}(z)$ as the CSD matrix of a set of convolatively mixed signals $\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$ in a subband coding application. Vaidyanathan [27] proved that the optimal subband coders (with maximum coding gain) can be realised if and only if $\underline{\tilde{\mathbf{D}}}(z)$ is strongly decorrelated (diagonalised) and spectrally majorised. In other words, the unique solution $\underline{\tilde{\mathbf{D}}}(z)$, which directly corresponds to the maximum coding gain, exists and therefore it follows that $\underline{\tilde{\mathbf{D}}}(z) = \underline{\mathbf{D}}(z)$. Note that the spectral majorisation property of PEVD can be seen as the extension of ordered EVD for para-Hermitian matrices, and further details will be discussed in Section 3.4.

However, there is ambiguity with respect to the paraunitary matrix $\underline{\tilde{\mathbf{H}}}(z)$ [83, 84]. Let $\underline{\tilde{\mathbf{H}}}(z) = \underline{\Gamma}(z)\underline{\mathbf{H}}(z)$, where $\underline{\Gamma}(z) \in \underline{\mathbb{C}}^{M \times M}$ is a diagonal paraunitary matrix satisfying $\underline{\tilde{\Gamma}}(z)\underline{\Gamma}(z) = \mathbf{I}$; then (3.2) can be rewritten as

$$\underline{\mathbf{R}}(z) = \underline{\tilde{\mathbf{H}}}(z)\underline{\tilde{\Gamma}}(z)\underline{\mathbf{D}}(z)\underline{\Gamma}(z)\underline{\mathbf{H}}(z). \quad (3.3)$$

Thus, it follows that $\underline{\tilde{\Gamma}}(z)\underline{\mathbf{D}}(z)\underline{\Gamma}(z) = \underline{\mathbf{D}}(z)$ provided $\underline{\mathbf{D}}(z)$ is precisely diagonal. Here $\underline{\Gamma}(z)$ can be seen as a lossless filter bank which does not affect the norm of $\underline{\mathbf{D}}(z)$, and its diagonal elements only comprise time-shift and phase adjustment terms, i.e.,

$$\underline{\Gamma}(z) = \text{diag}\{e^{j\alpha_1}z^{-\tau_1}, e^{j\alpha_2}z^{-\tau_2}, \dots, e^{j\alpha_M}z^{-\tau_M}\}. \quad (3.4)$$

Therefore, even if the diagonal para-Hermitian matrix $\underline{\mathbf{D}}(z)$ is unique, the paraunitary matrix $\underline{\tilde{\mathbf{H}}}(z)$ is ambiguous. The benefit of the ambiguity has been exploited in the row-shift corrected truncation method for the paraunitary matrix [83, 85] in order to further reduce its polynomial order.

3.2.2 Anatomy of the PEVD Algorithms

All the existing PEVD algorithms, including the SBR2 [7, 28] and SMD [30, 35] algorithm classes, operate by applying a sequence of elementary paraunitary matrices to the input para-Hermitian matrix $\underline{\mathbf{R}}(z)$ in order to produce an approximately diago-

nal polynomial matrix $\underline{\mathbf{D}}(z)$. This process comprises a number of iterations, and each iteration aims to annihilate a certain number of the off-diagonal elements in $\underline{\mathbf{R}}(z)$.

Despite the differences between the PEVD algorithms in terms of implementation, all algorithms generally consist of three common steps at each iteration. For the i -th iteration, the first step is to search for the remaining off-diagonal elements of the para-Hermitian matrix $\underline{\mathbf{R}}^{(i-1)}(z)$ obtained from the previous iteration. Note that the search strategy varies with different algorithms. Then, part of the off-diagonal elements in $\underline{\mathbf{R}}^{(i-1)}(z)$ are brought onto the zero-lag coefficient matrix in the second step using an elementary paraunitary shift matrix $\underline{\mathbf{B}}^{(i)}(z)$, which results in the transformed polynomial matrix

$$\underline{\mathbf{R}}^{(i)'}(z) = \underline{\mathbf{B}}^{(i)}(z)\underline{\mathbf{R}}^{(i-1)}(z)\tilde{\underline{\mathbf{B}}}^{(i)}(z), \quad (3.5)$$

where $\underline{\mathbf{B}}^{(i)}(z)$ is algorithm dependent and is determined by the search strategy of the specific PEVD algorithm, as detailed in the following sections. The final step is to transfer the off-diagonal energy which was shifted in step two onto the diagonal of the zero-lag matrix. This is implemented by means of a unitary rotation matrix $\mathbf{Q}^{(i)}$, which is applied to all lags in the para-Hermitian matrix $\underline{\mathbf{R}}^{(i)'}(z)$, such that

$$\underline{\mathbf{R}}^{(i)}(z) = \mathbf{Q}^{(i)}\underline{\mathbf{R}}^{(i)'}(z)\mathbf{Q}^{(i)\text{H}}. \quad (3.6)$$

Note that the construction of the unitary rotation matrix $\mathbf{Q}^{(i)}$ is also defined by the specific PEVD algorithm. By combining the shift and rotation steps in (3.5) and (3.6), we have

$$\underline{\mathbf{R}}^{(i)}(z) = \mathbf{Q}^{(i)}\underline{\mathbf{B}}^{(i)}(z)\underline{\mathbf{R}}^{(i-1)}(z)\tilde{\underline{\mathbf{B}}}^{(i)}(z)\mathbf{Q}^{(i)\text{H}}, \quad (3.7)$$

and the resulting elementary paraunitary matrix $\underline{\mathbf{G}}^{(i)}(z)$ at the i -th iteration can be expressed as

$$\underline{\mathbf{G}}^{(i)}(z) = \mathbf{Q}^{(i)}\underline{\mathbf{B}}^{(i)}(z). \quad (3.8)$$

Thus, the i -th iteration of any iterative PEVD algorithm is then accomplished by

$$\underline{\mathbf{R}}^{(i)}(z) = \underline{\mathbf{G}}^{(i)}(z)\underline{\mathbf{R}}^{(i-1)}(z)\tilde{\underline{\mathbf{G}}}^{(i)}(z). \quad (3.9)$$

Each of the PEVD algorithms stops when either the off-diagonal energy of $\underline{\mathbf{R}}^{(i)}(z)$ falls below a predefined threshold or a specified number of iterations, I , have been conducted. Once the algorithm is complete, it returns the approximate polynomial eigenvalues in the diagonalised para-Hermitian matrix $\underline{\mathbf{R}}^{(I)}(z)$ and the approximate polynomial eigenvectors in $\underline{\mathbf{H}}^{(I)}(z)$ such that

$$\underline{\mathbf{R}}^{(I)}(z) = \underline{\mathbf{H}}^{(I)}(z)\underline{\mathbf{R}}(z)\tilde{\underline{\mathbf{H}}}^{(I)}(z), \quad (3.10)$$

where the paraunitary matrix $\underline{\mathbf{H}}^{(I)}(z)$ is simply the product of the unitary rotation matrices $\underline{\mathbf{Q}}^{(i)}$ and the paraunitary shift matrices $\underline{\mathbf{B}}^{(i)}(z)$ for $i = 1, \dots, I$, i.e.,

$$\underline{\mathbf{H}}^{(I)}(z) = \underline{\mathbf{G}}^{(I)}(z)\underline{\mathbf{G}}^{(I-1)}(z)\dots\underline{\mathbf{G}}^{(1)}(z). \quad (3.11)$$

Provided I is chosen to be reasonably large, the approximate diagonalised polynomial matrix $\underline{\mathbf{R}}^{(I)}(z)$ is considered as the solution of PEVD, i.e., $\underline{\mathbf{R}}^{(I)}(z) \approx \underline{\mathbf{D}}(z)$. Furthermore, the paraunitary transformation is norm preserving, such that

$$\left\| \underline{\mathbf{H}}^{(I)}(z)\underline{\mathbf{R}}(z)\tilde{\underline{\mathbf{H}}}^{(I)}(z) \right\|_{\text{F}}^2 = \|\underline{\mathbf{R}}(z)\|_{\text{F}}^2, \quad (3.12)$$

where the notation $\|\cdot\|_{\text{F}}^2$ represents the squared Frobenius norm.

Note that the paraunitary shift operation $\underline{\mathbf{B}}^{(i)}(z)$ for all PEVD algorithms will increase both the polynomial order of $\underline{\mathbf{R}}^{(I)}(z)$ and $\underline{\mathbf{H}}^{(I)}(z)$. This is problematic, as the order growth of the para-Hermitian matrix $\underline{\mathbf{R}}^{(I)}(z)$ will lead to a significant increase in the computational complexity of the PEVD algorithms as the number of iterations increases. To tackle this problem, polynomial order truncation methods [7, 86] for

para-Hermitian matrices have been developed to reduce the complexity of the iterative PEVD algorithms. Although the growing order of the paraunitary matrix $\underline{\mathbf{H}}^{(l)}(z)$ does not affect the algorithms' complexity during iterations, the application cost of the final generated paraunitary matrix can be high for some applications, including paraunitary filter bank design for channel coding [17], broadband MIMO decoupling [31, 34], and broadband angle of arrival estimation [12], etc. Therefore, different methods of truncating the order of the paraunitary matrix have been proposed in [83, 87]. Details of order truncation techniques will be discussed in Section 3.6.

3.3 Second Order Sequential Best Rotation Algorithm

3.3.1 Outline of the SBR2 Algorithm

The SBR2 algorithm [7] is the first iterative algorithm for computing PEVD. At the i -th iteration, the algorithm starts by searching for the off-diagonal coefficient with maximum magnitude in $\underline{\mathbf{R}}^{(i-1)}(z)$. According to the para-Hermitian property in (2.18), the search space of the maximum off-diagonal element can be restricted either to the upper triangular region across all the lags in $\underline{\mathbf{R}}^{(i-1)}(z)$ or to the positive (negative) lags plus the zero-lag, excluding all the on-diagonal elements; the algorithm convergence remains the same for both methods. Assuming that the maximum off-diagonal element found at the i -th iteration is represented by $r_{jk}^{(i)}[\tau]$, its location parameters satisfy

$$\{j^{(i)}, k^{(i)}, \tau^{(i)}\} = \arg \max_{j,k,\tau} \left\| \underline{\mathbf{R}}^{(i-1)}[\tau] \right\|_{\infty}, \quad \forall \tau, \quad (3.13)$$

where $j^{(i)}$, $k^{(i)}$, and $\tau^{(i)}$ denote the corresponding row, column, and lag indices of $r_{jk}^{(i)}[\tau]$. The notation $\|\cdot\|_{\infty}$ represents L_{∞} norm. Note that for the results presented in this thesis, the search space of the maximum off-diagonal element is restricted to the upper triangular area of $\underline{\mathbf{R}}^{(i-1)}(z)$, i.e., $j < k$.

3.3 Second Order Sequential Best Rotation Algorithm

described in (3.9). For a more intuitive illustration, Figure 3.1 demonstrates a single iteration of SBR2 when applied to a 5×5 para-Hermitian matrix of order 4.

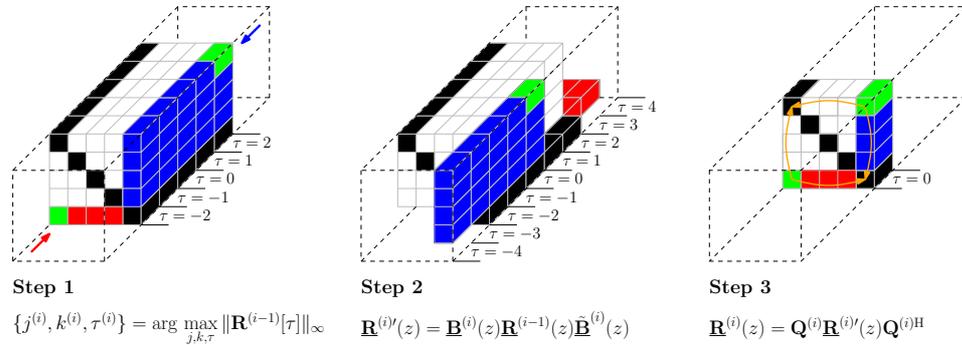


Figure 3.1 A 3D illustration showing a single iteration of the SBR2 algorithm when applied to diagonalising a 5×5 para-Hermitian matrix example, where **Step 1** identifies the location of $r_{jk}^{(i)}[\tau]$ (marked in green), **Step 2** describes the corresponding row and column shift operations, and **Step 3** transfers the pairwise maximum elements onto the diagonal (only the zero-lag coefficient matrix is shown here for visibility purposes) [31].

The algorithm continues by repeating the iterative process mentioned above until the magnitude of the maximum off-diagonal element $|r_{jk}^{(i)}[\tau]|$ is found to be smaller than a given threshold ε , which can be set to a very small value to achieve sufficient accuracy. Assuming that the algorithm has converged by the I -th iteration, the diagonalised para-Hermitian matrix $\underline{\mathbf{D}} \in \mathbb{C}^{M \times M}$ takes the form

$$\underline{\mathbf{D}}(z) = \text{diag} \{ \underline{d}_{11}(z), \underline{d}_{22}(z), \dots, \underline{d}_{MM}(z) \}, \quad (3.17)$$

and the resulting paraunitary matrix $\underline{\mathbf{H}} \in \mathbb{C}^{M \times M}$ can be calculated using (3.11). For further details of the SBR2 algorithm, see [7].

3.3.2 Applications of the SBR2 Algorithm

Strong Decorrelation

In Section 2.3.2, we discussed the convolutive mixing model in the context of broadband sensor array processing. As a result of the convolutive mixing, the sensor signals

3.3 Second Order Sequential Best Rotation Algorithm

$\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$ for $t = 0, \dots, T - 1$, as shown in (2.21), are generally correlated with one another, not just at the same time instant but also over a range of time delays. Specifically, the correlation between each pair of sensors at the same time instant is referred to as the spatial correlation, while the correlation over a chosen range of time delays represents the temporal correlation. The conventional EVD or SVD can only be used to remove the spatial correlation in the case of a narrowband sensor array. To remove both spatial and temporal correlations between different sensor signals, strong decorrelation is required. The idea of strong decorrelation (or total decorrelation) [27] is to find a matrix of suitably chosen FIR filters $\underline{\mathbf{H}}(z)$ which can be applied to minimise the cross-correlations between different sensors. Figure 3.2 is a schematic diagram which illustrates the idea of decorrelating a set of convolutively mixed signals $\mathbf{x}[t]$ by means of a paraunitary filter bank $\underline{\mathbf{H}}(z)$. Assuming that the sensor outputs $\mathbf{x}[t]$ have zero mean,

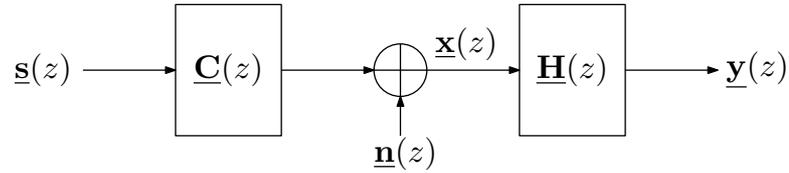


Figure 3.2 Strong decorrelation for a set of convolutively mixed sensor signals.

the space-time covariance matrix can be defined as $\mathbf{R}_{xx}[\tau] = E\{\mathbf{x}[t]\mathbf{x}^H[t - \tau]\}$ for $\tau \in \mathbb{Z}$, and its z -transform, $\underline{\mathbf{R}}_{xx}(z) \triangleq \sum_{\tau=-\infty}^{\infty} \mathbf{R}_{xx}[\tau]z^{-\tau}$, yields the CSD matrix

$$\underline{\mathbf{R}}_{xx}(z) = \sum_{\tau=-\infty}^{\infty} \mathbf{R}_{xx}[\tau]z^{-\tau}. \quad (3.18)$$

Since the sensors signals $\mathbf{x}[t]$ are correlated, $\mathbf{R}_{xx}[\tau]$ will not be diagonal $\forall \tau$ and hence $\underline{\mathbf{R}}_{xx}(z)$ will not be diagonal. In practice, the CSD matrix can be estimated as [7]

$$\hat{\underline{\mathbf{R}}}_{xx}(z) \triangleq \sum_{\tau=-W}^W \hat{\mathbf{R}}_{xx}[\tau]z^{-\tau}, \quad (3.19)$$

where

$$\hat{\mathbf{R}}_{xx}[\tau] \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \mathbf{x}[t] \mathbf{x}^H[t - \tau] \quad (3.20)$$

and W represents the correlation window parameter, which is often measured experimentally. The following assumptions are made when calculating $\hat{\mathbf{R}}_{xx}[\tau]$:

1. $\hat{\mathbf{R}}_{xx}[\tau] = 0$ for $|\tau| > W$,
2. $T \gg W$ and
3. $\mathbf{x}[t] = 0$ for $t \notin [0, 1, \dots, T - 1]$.

Since the individual elements in $\hat{\mathbf{R}}_{xx}[\tau]$ are the auto and cross-correlation sequences of the sensor signals $x_k[t]$ and $x_l[t]$ for $k, l = 1, \dots, M$, it follows that

$$r_{x_k x_l}[\tau] = r_{x_l x_k}^*[-\tau], \quad |\tau| \leq W \text{ and } \tau \in \mathbb{Z}. \quad (3.21)$$

Thus, the polynomial matrix $\hat{\mathbf{R}}_{xx}(z)$ is para-Hermitian by construction. Therefore, the SBR2 algorithm [7] can be used to generate the required paraunitary matrix $\mathbf{H}(z)$ such that

$$\mathbf{H}(z) \hat{\mathbf{R}}_{xx}(z) \tilde{\mathbf{H}}(z) = \hat{\mathbf{D}}(z), \quad (3.22)$$

where $\hat{\mathbf{D}}(z)$ is approximately diagonal. The strong decorrelation of the sensor signals $\mathbf{x}(z)$ is then implemented by the transformation

$$\mathbf{y}(z) = \mathbf{H}(z) \mathbf{x}(z), \quad (3.23)$$

where $\mathbf{y}(z)$ denotes the transformed signals. To a good approximation, the CSD matrix of $\mathbf{y}(z)$ can be estimated by

$$\hat{\mathbf{R}}_{yy}(z) = \mathbf{H}(z) \hat{\mathbf{R}}_{xx}(z) \tilde{\mathbf{H}}(z) = \hat{\mathbf{D}}(z), \quad (3.24)$$

which is an approximately diagonal matrix. This means that the correlations between different sensor signals are eliminated. In this thesis, an improved version of SBR2, called MS-SBR2, is proposed and its effectiveness in strong decorrelation of a set of sensor signals is demonstrated by a numerical example in Section 4.7.

Spectral Majorisation

The decorrelated signals $\mathbf{y}[t] \in \mathbb{C}^{M \times 1}$ produced by the SBR2 algorithm also satisfy the spectral majorisation property [27], which states that the power spectral densities (PSDs) of the on-diagonal elements $\underline{d}_{mm}(e^{j\Omega}) = \underline{d}_{mm}(z)|_{z=e^{j\Omega}}$ for $m = 1, \dots, M$ in the diagonalised CSD matrix $\underline{\mathbf{D}}(z)$ satisfy

$$\underline{d}_{11}(e^{j\Omega}) \geq \underline{d}_{22}(e^{j\Omega}) \geq \dots \geq \underline{d}_{mm}(e^{j\Omega}), \quad \forall \Omega \in [-\pi, \pi]. \quad (3.25)$$

This means that the expected powers in $y_m[t]$ for $m = 1, \dots, M$ are arranged in descending order at every frequency. This is analogous to an ordered EVD in the narrowband case. It is important to note that the combined power of the received signals is invariant under a paraunitary transformation [69], i.e.,

$$\text{trace}\{\underline{\mathbf{R}}_{yy}(e^{j\Omega})\} = \text{trace}\{\underline{\mathbf{H}}(e^{j\Omega})\underline{\mathbf{R}}_{xx}(e^{j\Omega})\underline{\mathbf{H}}^{\dagger}(e^{j\Omega})\} = \text{trace}\{\underline{\mathbf{R}}_{xx}(e^{j\Omega})\}, \quad (3.26)$$

and using a paraunitary transformation can only redistribute the spectral power between channels.

The spectral majorisation property provides a very useful tool in applications that are based on the broadband subspace decomposition, such as broadband beamforming and BSS. As spectrally majorised signals tend to have most of the related signal energy focused in as few channels as possible [7, 88], the signal and noise subspaces can be easily identified. This feature will be demonstrated by a numerical example in Section 4.7, where the proposed multiple shift SBR2 algorithm is used to decorrelate a set of convolutively mixed signals.

3.4 A Novel Insight into the SBR2 Algorithm

In this section, we take a fresh look at the SBR2 algorithm in terms of its potential for optimising the subband coding gain. We demonstrate how every iteration of the SBR2 algorithm must lead to an increase in the subband coding gain until it comes arbitrarily close to its maximum possible value. This leads to the much desired proof that the SBR2 algorithm does indeed converge towards a spectrally majorised solution. A new quantity γ associated with the coding gain optimisation is introduced, and its monotonic behaviour brings new insight to the convergence of the SBR2 algorithm. Based on this novel insight, a modified SBR2 algorithm is explicitly designed to maximise the coding gain.

3.4.1 Coding Gain

The subband coder is a generalisation of the transform coder and has been used in the area of data compression [27]. It aims to maximise the coding gain, i.e., to minimise the mean square reconstruction error due to subband quantisation. The necessary and sufficient conditions for maximising the coding gain are [27, 89]:

1. strong decorrelation – this means that the CSD matrix $\underline{\mathbf{R}}(z)$ has been diagonalised, or equivalently that the subband signals $\mathbf{y}[t] = \sum_{\tau=0}^T \mathbf{H}[\tau] \mathbf{x}[t - \tau]$ are totally uncorrelated, i.e., $E\{y_k[t] y_l^*[t - \tau]\} = 0, k \neq l, \forall t$ and τ .
2. spectral majorisation – the PSDs of the on-diagonal elements in $\underline{\mathbf{D}}(z)$ satisfy the spectral majorisation property, as described in (3.25).

Given a CSD matrix $\underline{\mathbf{R}}(z) = \sum_{\tau} \mathbf{R}[\tau] z^{-\tau}$ of the sensor signals $\mathbf{x}[t] \in \mathbb{C}^{M \times 1}$, the coding gain, whose maximisation requires diagonalisation and spectral majorisation, is measured as the ratio of the arithmetic and geometric means of the channel variances. At the i -th iteration, the variance of the m -th channel is given by $r_{mm}^{(i)}[0]$, where $m =$

$1, \dots, M$, so the coding gain is defined as [35]

$$G^{(i)} = \frac{\frac{1}{M} \sum_{m=1}^M r_{mm}^{(i)}[0]}{\left(\prod_{m=1}^M r_{mm}^{(i)}[0] \right)^{\frac{1}{M}}}. \quad (3.27)$$

Note that the trace

$$\text{trace}\{\mathbf{R}^{(i)}[0]\} = \sum_{m=1}^M r_{mm}^{(i)}[0] = \text{trace}\{\mathbf{R}[0]\} = \text{trace}\{\mathbf{D}[0]\} \quad (3.28)$$

is invariant under paraunitary transformations, so maximising the coding gain is equivalent to minimising the product of variances in the denominator of (3.27).

3.4.2 Proof of Spectral Majorisation

The SBR2 algorithm has been adopted successfully in the design of the paraunitary (or-thonormal) filter banks for subband coding [17, 21]. In effect, it has demonstrated the capability of a principal component filter bank (PCFB) by achieving the optimal coding gain. However, there is no proof in the existing literature that the SBR2 algorithm will always produce the necessary spectral majorisation. In the rest of this section a proof of this important property will be derived.

Theorem (Spectral Majorisation of the SBR2 Algorithm): If strong decorrelation is achieved using the SBR2 algorithm, the resulting PSDs of the on-diagonal elements in the diagonalised para-Hermitian matrix must also be spectrally majorised.

Proof: To prove this theorem, let us define two polynomial matrices $\mathbf{R}'(z)$ and $\mathbf{R}''(z)$, which respectively represent the resulting polynomial matrices after the elementary delay operation and the Jacobi rotation in SBR2, and introduce the parameter

$$\gamma \triangleq \prod_{m=1}^M r_{mm}[0], \quad (3.29)$$

3.4 A Novel Insight into the SBR2 Algorithm

where $r_{mm}[0]$ for $m = 1, \dots, M$ represent the diagonal element of $\mathbf{R}[0]$. Following the elementary delay step, the SBR2 algorithm employs a Jacobi rotation as shown in (3.16) to transfer the energy of the off-diagonal element $r'_{jk}[0] = r_{jk}[\tau]$ and its conjugate $r'_{kj}[0] = r_{kj}[-\tau]$ onto the diagonal by choosing the rotation parameters such that

$$\begin{bmatrix} c & se^{j\phi} \\ -se^{-j\phi} & c \end{bmatrix} \begin{bmatrix} r'_{jj}[0] & r'_{jk}[0] \\ r'_{kj}[0] & r'_{kk}[0] \end{bmatrix} \begin{bmatrix} c & -se^{j\phi} \\ se^{-j\phi} & c \end{bmatrix} = \begin{bmatrix} r''_{jj}[0] & 0 \\ 0 & r''_{kk}[0] \end{bmatrix}, \quad (3.30)$$

where c and s denote the $\cos \theta$ and $\sin \theta$, respectively. As a Jacobi rotation only affects rows j and k , columns j and k , only the 2×2 sub-matrices of the elementary rotation step, as described in (3.6), are presented. Note that the rest of the diagonal elements in $\mathbf{R}'[0]$ will not change under the Jacobi rotation. Since the transformations are unitary, it follows that

$$\det \left\{ \begin{bmatrix} r'_{jj}[0] & r'_{jk}[0] \\ r'_{kj}[0] & r'_{kk}[0] \end{bmatrix} \right\} = \det \left\{ \begin{bmatrix} r''_{jj}[0] & 0 \\ 0 & r''_{kk}[0] \end{bmatrix} \right\}. \quad (3.31)$$

Thus,

$$r''_{jj}[0]r''_{kk}[0] = r'_{jj}[0]r'_{kk}[0] - |r'_{jk}[0]|^2 = r_{jj}[0]r_{kk}[0] - |r_{jk}[\tau]|^2, \quad (3.32)$$

where we have taken account the fact that $r'_{jj}[0] = r_{jj}[0]$, $r'_{kk}[0] = r_{kk}[0]$, and $r'_{jk}[0] = r_{jk}[\tau]$. Since $r_{jk}[\tau] \neq 0$, it follows that

$$r''_{jj}[0]r''_{kk}[0] < r_{jj}[0]r_{kk}[0]. \quad (3.33)$$

Also, since only the j -th and k -th diagonal elements are altered during the iteration, we have

$$\gamma'' \triangleq \prod_{m=1}^M r''_{mm}[0] < \gamma. \quad (3.34)$$

Clearly the denominator in (3.27), which is directly related to $\gamma^{(i)}$, is monotonically reduced at each iteration in SBR2, i.e., $\gamma^{(i)} < \gamma^{(i-1)}$, until no further reduction is possible

($|r_{jk}[\tau]| < \varepsilon$). It follows that the coding gain $G^{(i)}$ increases monotonically to attain its maximum value $G^{(l)}$.

It was clearly demonstrated by Vaidyanathan [27] that the optimum coding gain (which requires a PCFB) can be achieved if and only if strong decorrelation and spectral majorisation have been obtained. Thus it follows that the SBR2 algorithm, which was explicitly designed to achieve strong decorrelation, must not only achieve that objective, but also produce spectral majorisation. ■

3.4.3 Modified SBR2 Algorithm

Instead of finding the maximum element $|r_{jk}^{(i)}[\tau]|$ at the i -th iteration, it is now possible to consider the coding gain $G^{(i)}$ as a convergence indicator for the SBR2 algorithm. This gives us a useful new insight whereby the SBR2 algorithm converges uniformly due to the monotonic behaviour of $\gamma^{(i)}$, in contrast with the original convergence factor $|r_{jk}^{(i)}[\tau]|$ whose value does not reduce monotonically. An alternative approach, therefore, is to monitor the gradient of the coding gain, which is defined as $\rho^{(i)} = G^{(i)} - G^{(i-1)}$. As the value of $\rho^{(i)}$ is not guaranteed to reduce monotonically, the average value $\hat{\rho}$ over a suitably chosen range $W \in \mathbb{Z}$ can be used to assess the convergence behaviour, i.e.,

$$\hat{\rho} = \frac{1}{W} \sum_{n=i-W+1}^i \rho^{(n)}, \quad i \geq W. \quad (3.35)$$

Then the iterative process stops when the value of $\hat{\rho}$ is sufficiently small. The modified SBR2 algorithm is summarised in Algorithm 1.

Algorithm 1 Modified SBR2 Algorithm for Optimum Coding Gain

Input: para-Hermitian matrix $\underline{\mathbf{R}}(z) \in \mathbb{C}^{M \times M}$, maximum number of iterations I , convergence parameter ε' , and trim parameter μ

Output: diagonalised para-Hermitian matrix $\underline{\mathbf{D}}(z)$ and paraunitary matrix $\underline{\mathbf{H}}(z)$

- 1: **Initialisation:** $iter \leftarrow 0$, $\underline{\mathbf{H}}(z) \leftarrow \mathbf{I}_M$, $\hat{\rho} \leftarrow 1 + \varepsilon'$ and assign a value to W
- 2: **while** $iter < I$ and $\hat{\rho} > \varepsilon'$ **do**
- 3: locate the maximum off-diagonal element $r_{jk}[\tau]$;
- 4: $g \leftarrow |r_{jk}[\tau]|$;
- 5: **if** $iter = 0$ and $g = 0$ **then**
- 6: break;
- 7: **else**
- 8: % Time-shift the maximum off-diagonal element
- 9: $\underline{\mathbf{R}}'(z) \leftarrow \underline{\mathbf{B}}^{(k,\tau)}(z)\underline{\mathbf{R}}(z)\tilde{\underline{\mathbf{B}}}^{(k,\tau)}(z)$; $\underline{\mathbf{H}}'(z) \leftarrow \underline{\mathbf{B}}^{(k,\tau)}(z)\underline{\mathbf{H}}(z)$;
- 10: % Perform the Jacobi rotation step
- 11: compute rotation parameters θ and ϕ ;
- 12: $\underline{\mathbf{R}}(z) \leftarrow \mathbf{Q}^{(j,k)}(\theta, \phi)\underline{\mathbf{R}}'(z)\mathbf{Q}^{(j,k)\text{H}}(\theta, \phi)$; $\underline{\mathbf{H}}(z) \leftarrow \mathbf{Q}^{(j,k)}(\theta, \phi)\underline{\mathbf{H}}'(z)$;
- 13: $iter \leftarrow iter + 1$;
- 14: % Trim polynomial matrix order
- 15: $\underline{\mathbf{R}}(z) \leftarrow \text{TrimPH}(\underline{\mathbf{R}}(z), \mu)$; $\underline{\mathbf{H}}(z) \leftarrow \text{TrimPU}(\underline{\mathbf{H}}(z), \mu)$
using [86, 87]
- 16: compute $G^{(iter)}$ and $\rho^{(iter)}$ according to (3.27);
- 17: **if** $iter \geq W$ **then**
- 18: $\hat{\rho} \leftarrow \frac{1}{W} \sum_{n=iter-W+1}^{iter} \rho^{(n)}$;
- 19: **end if**
- 20: **end if**
- 21: **end while**

Numerical Example

To demonstrate the modified SBR2 algorithm in terms of the coding gain optimisation, one of the test examples used in the original SBR2 paper [7] is chosen, where a set of convolutively mixed signals $\mathbf{x}[t] \in \mathbb{R}^{3 \times 1}$ is generated from a 2×3 convolutive MIMO channel model with a 3×2 polynomial mixing matrix $\underline{\mathbf{A}}(z)$ whose entries comprise 5-th order finite impulse response (FIR) filters. Gaussian random noise was also added to each sensor output, resulting in a signal-to-noise ratio (SNR) of 5.3 dB.

Figure 3.3(a) plots the coefficients of the CSD matrix $\underline{\mathbf{R}}(z)$ of the mixed signals $\mathbf{x}[t]$. The modified SBR2 algorithm is then applied to this matrix, and it stops when the

3.5 Sequential Matrix Diagonalisation Algorithms

average gradient $\hat{\rho}$ reaches the predefined lower bound value, i.e., $\hat{\rho} \leq \epsilon' = 10^{-5}$. Here the parameter W is set equal to 10. The algorithm converged in 110 iterations to a point where the average gradient $\hat{\rho} = 0.96 \times 10^{-5}$. The diagonalised CSD matrix is plotted in Figure 3.3(b), where its polynomial order has been truncated to the same length as the input CSD matrix. As the product of the subband variances $\gamma^{(i)}$ is monotonically reduced as the iterations continue, the coding gain $G^{(i)}$ is monotonically increasing, as shown in Figure 3.4(a). As opposed to the original SBR2 algorithm, Figure 3.4(b) shows the behaviour of the convergence factor $g = |r_{jk}^{(i)}[\tau]|$, for which it converged at $g = 0.0224$. It is obvious that the conventional convergence parameter g used in the SBR2 algorithm does not possess monotonic behaviour in comparison with the coding gain in the modified SBR2 algorithm. The monotonic behaviour of the coding gain provides a more reliable test of convergence for the algorithm.

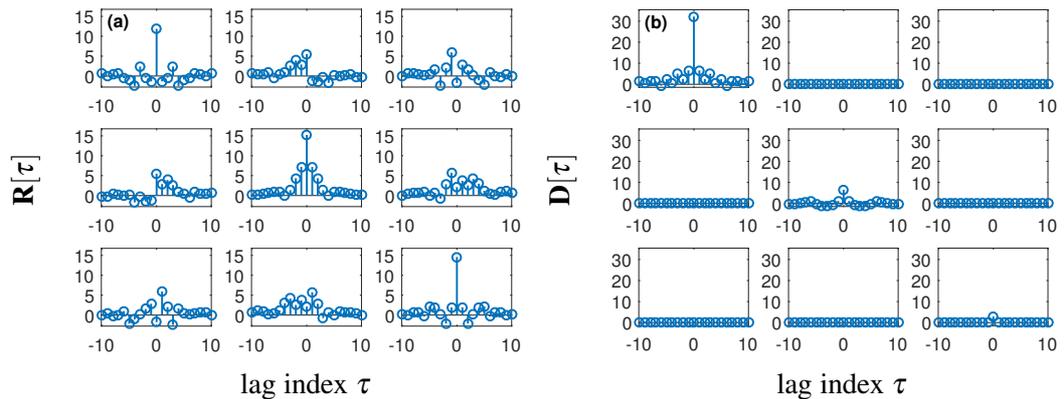


Figure 3.3 The stem plot of (a) the CSD matrix and (b) the diagonalised CSD matrix obtained from the modified SBR2.

3.5 Sequential Matrix Diagonalisation Algorithms

Recently, the sequential matrix diagonalisation (SMD) algorithm [35] and its derivative versions, including maximum element SMD (ME-SMD) [35] and multiple shift maximum element SMD (MSME-SMD) [30, 90, 91] algorithms, have been developed to calculate the PEVD of para-Hermitian matrices. These variants are different only in

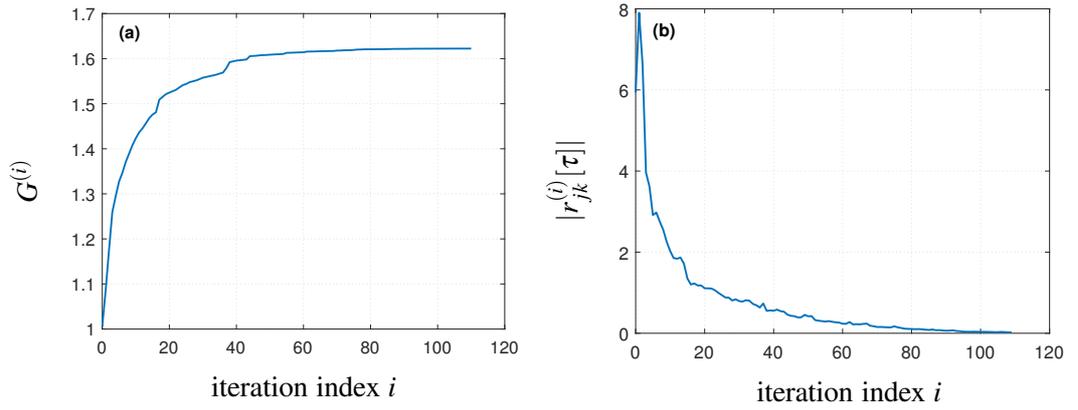


Figure 3.4 Convergence of the SBR2 algorithm when applied to the example CSD matrix in Figure 3.3, showing (a) the behaviour of the coding gain $G^{(i)}$ and (b) the behaviour of the magnitude of the maximum off-diagonal element found at the i -th iteration.

the search strategies that are used to transfer off-diagonal energy onto the zero-lag coefficient matrix. Unlike the SBR2 algorithm, the SMD family requires an initialisation step to diagonalise the zero-lag coefficient matrix $\mathbf{R}[0] \in \mathbb{C}^{M \times M}$ before all iterations. This is implemented by computing a full EVD to $\mathbf{R}[0]$, i.e.,

$$\mathbf{S}^{(0)}[0] = \mathbf{Q}^{(0)}\mathbf{R}[0]\mathbf{Q}^{(0)H}, \quad (3.36)$$

and then applying the modal matrix $\mathbf{Q}^{(0)}$, which is a unitary matrix, to all lags of the para-Hermitian matrix $\underline{\mathbf{R}}(z)$, such that

$$\underline{\mathbf{S}}^{(0)}(z) = \mathbf{Q}^{(0)}\underline{\mathbf{R}}(z)\mathbf{Q}^{(0)H}. \quad (3.37)$$

Following on that, each iteration of the SMD algorithms can be summarised into three common steps operating on $\underline{\mathbf{S}}^{(i-1)}(z)$ for $i = 1, \dots, I$:

- step 1: searching the maximum column norm (or the maximum off-diagonal element) in $\underline{\mathbf{S}}^{(i-1)}(z)$;

- step 2: shifting column(s) and row(s) onto the zero-lag coefficient matrix using a paraunitary shift matrix $\underline{\mathbf{B}}^{(i)}(z)$, such that

$$\underline{\mathbf{S}}^{(i)'}(z) = \underline{\mathbf{B}}^{(i)}(z)\underline{\mathbf{S}}^{(i-1)}(z)\tilde{\underline{\mathbf{B}}}^{(i)}(z), \quad (3.38)$$

where $\underline{\mathbf{S}}^{(i)'}(z)$ denotes the intermediate matrix after the paraunitary shift operation; and

- step 3: calculating a full EVD of the zero-lag coefficient matrix $\underline{\mathbf{S}}^{(i)'}[0]$ and applying the corresponding modal matrix to all lags of $\underline{\mathbf{S}}^{(i)'}(z)$,

$$\underline{\mathbf{S}}^{(i)}(z) = \mathbf{Q}^{(i)}\underline{\mathbf{S}}^{(i)'}(z)\mathbf{Q}^{(i)H}(z). \quad (3.39)$$

The search step described above is algorithm dependent and will be discussed separately for each version of the SMD algorithm in the following sub-sections. Note that the unitary matrix $\mathbf{Q}^{(i)}$ is different from the elementary rotation matrix in the SBR2 algorithm, as it transfers all of the off-diagonal energy onto the diagonal at the zero-lag plane instead of just the energy of a single element onto the diagonal like SBR2. For this reason, the SMD family is much more complex than the SBR2 algorithm at each iteration.

3.5.1 The SMD Algorithm

For the i -th iteration, the original SMD algorithm [35] starts by searching the column that contains the maximum off-diagonal energy, which is identified by the parameter set

$$\{k^{(i)}, \tau^{(i)}\} = \arg \max_{k, \tau} \left\| \hat{\mathbf{s}}_k^{(i-1)}[\tau] \right\|_2, \quad \forall \tau, \quad (3.40)$$

3.5 Sequential Matrix Diagonalisation Algorithms

where the notation $\|\cdot\|_2$ denotes the L_2 norm, and the vector $\hat{\mathbf{s}}_k^{(i-1)}[\boldsymbol{\tau}]$ contains all elements in the $k^{(i-1)}$ -th column of $\mathbf{S}^{(i-1)}[\boldsymbol{\tau}]$ except for the diagonal element, i.e.,

$$\left\| \hat{\mathbf{s}}_k^{(i-1)}[\boldsymbol{\tau}] \right\|_2 = \sqrt{\sum_{j=1, j \neq k}^M \left| s_{jk}^{(i-1)}[\boldsymbol{\tau}] \right|^2}. \quad (3.41)$$

Based on the location information $\{k^{(i)}, \tau^{(i)}\}$, the paraunitary shift matrix $\underline{\mathbf{B}}^{(i)}(z)$ can be expressed as

$$\underline{\mathbf{B}}^{(i)}(z) = \text{diag} \left\{ \underbrace{1, \dots, 1}_{k^{(i)}-1}, z^{-\tau^{(i)}}, \underbrace{1, \dots, 1}_{M-k^{(i)}} \right\}, \quad (3.42)$$

and then the SMD algorithm shifts the corresponding row and column pair onto the zero-lag plane by means of the paraunitary shift operation, as described in (3.38).

In the final step, rather than just using a single Jacobi rotation as in the SBR2 algorithm, the SMD algorithm computes a full EVD for the zero-lag matrix $\mathbf{S}^{(i)'}[0]$ using the unitary matrix $\mathbf{Q}^{(i)}$. Like the initialisation step, the unitary matrix $\mathbf{Q}^{(i)}$ has to be applied to the entire para-Hermitian matrix $\underline{\mathbf{S}}^{(i)'}(z)$ according to (3.39).

Similar to the SBR2 algorithm, the SMD algorithm stops when either a set number of iterations, I , have been conducted or the found column norm is less than a predefined threshold. Once the algorithm stops, it returns the approximate polynomial eigenvalues in the diagonalised para-Hermitian matrix $\underline{\mathbf{S}}^{(I)}(z)$ and the approximate polynomial eigenvectors in the paraunitary matrix $\underline{\mathbf{H}}^{(I)}(z)$. The generated paraunitary matrix is simply the product of the unitary energy transfer matrices $\mathbf{Q}^{(i)}$ and paraunitary delay matrices $\underline{\mathbf{B}}^{(i)}(z)$ from each of the I iterations,

$$\underline{\mathbf{H}}^{(I)}(z) = \underline{\mathbf{G}}^{(I)}(z) \cdots \underline{\mathbf{G}}^{(2)}(z) \underline{\mathbf{G}}^{(1)}(z), \quad (3.43)$$

where the elementary paraunitary matrix $\underline{\mathbf{G}}^{(i)}(z)$ can be expressed as

$$\underline{\mathbf{G}}^{(i)}(z) = \mathbf{Q}^{(i)}\underline{\mathbf{B}}^{(i)}(z), \quad i = 1, \dots, I. \quad (3.44)$$

3.5.2 Maximum Element SMD Algorithm

The ME-SMD algorithm [35] is introduced for the purpose of reducing the search cost in the SMD algorithm as described in Section 3.5.1. The only difference between them is that the ME-SMD algorithm replaces the L_2 norm in (3.40) with the L_∞ , i.e.,

$$\{k^{(i)}, \tau^{(i)}\} = \arg \max_{k, \tau} \left\| \hat{\mathbf{s}}_k^{(i-1)}[\tau] \right\|_\infty, \quad \forall \tau, \quad (3.45)$$

which is essentially the same as the search process in the SBR2 algorithm.

3.5.3 Multiple Shift Maximum Element SMD Algorithm

Instead of finding a single maximum off-diagonal element at each iteration, as in ME-SMD, the MSME-SMD algorithm [30] takes a more aggressive step by introducing a “multiple-shift” strategy which aims to search more off-diagonal elements and shift them to the zero-lag plane. This makes the MSME-SMD algorithm converge faster than both SBR2 and SMD in terms of reducing the off-diagonal energy.

For any $M \times M$ para-Hermitian matrix, MSME-SMD can find and shift a total of $M - 1$ maximum elements onto the zero-lag plane at every iteration by using a set of reduced search spaces. The location of each maximum element is constrained by the same condition as shown in (3.45), which is the same as that used in both SBR2 and ME-SMD. However, the maximum element search at the i -th iteration is carried out $M - 1$ times in total in order to maximise the off-diagonal energy brought to the zero-lag plane. After the $M - 1$ maximum elements have been found, they are then

transferred onto the zero-lag plane using the paraunitary shift matrix

$$\hat{\mathbf{B}}^{(i)}(z) = \mathbf{B}^{(1,i)}(z)\mathbf{B}^{(2,i)}(z)\cdots\mathbf{B}^{(M-1,i)}(z), \quad (3.46)$$

whereby each elementary shift matrix $\mathbf{B}^{(p,i)}(z)$ for $p = 1, \dots, M-1$ takes the form of

$$\mathbf{B}^{(p,i)}(z) = \text{diag} \left\{ \underbrace{1, \dots, 1}_{k^{(p,i)}-1}, z^{-\tau^{(p,i)}}, \underbrace{1, \dots, 1}_{M-k^{(p,i)}} \right\}, \quad (3.47)$$

and it is determined by the specific location parameter set for that shift. Note that $k^{(p,i)} \in \{1, \dots, M\}$ denotes the column index of the maximum off-diagonal element found at the p -th shift step of the i -th iteration, and $k^{(p,i)}, \forall p$ are different from each other according to the principle of the reduced search space method in MSME-SMD (see [30] for details). The diagonalisation step in MSME-SMD involves applying a full EVD to the zero-lag plane and subsequently the modal matrix to all lags in $\mathbf{S}^{(i)'}(z)$ according to (3.39), which is the same as for other members in the SMD family.

Although the MSME-SMD has the advantage over other PEVD algorithms in terms of the convergence speed, it comes at the expense of higher computational complexity, especially when the input para-Hermitian matrix is large. A detailed comparison of the PEVD algorithms will be presented by means of numerical examples in Chapter 5.

3.6 Polynomial Order Shortening Methods

3.6.1 Limitations of the PEVD Algorithms

To recap from (3.10), at the end of the i -th iteration of the PEVD algorithms [7, 28, 30, 29, 35], the decomposition takes the form of

$$\mathbf{R}^{(i)}(z) = \mathbf{H}^{(i)}(z)\mathbf{R}(z)\tilde{\mathbf{H}}^{(i)}(z), \quad (3.48)$$

where $\underline{\mathbf{R}}(z) \in \underline{\mathbb{C}}^{M \times M}$ denotes the input para-Hermitian matrix to the algorithm, $\underline{\mathbf{R}}^{(i)}(z)$ is the transformed para-Hermitian matrix after i iterations, and $\underline{\mathbf{H}}^{(i)}(z) \in \underline{\mathbb{C}}^{M \times M}$ is the resulting paraunitary transformation matrix. Note that $\underline{\mathbf{R}}^{(i)}(z)$ will be approximately diagonal, provided a sufficient number of iterations have been performed. The polynomial orders of $\underline{\mathbf{R}}^{(i)}(z)$ and $\underline{\mathbf{H}}^{(i)}(z)$ can increase due to the application of the paraunitary shift in (3.5). For example, in the SBR2 algorithm, each pairwise shift matrices $\underline{\mathbf{B}}^{(i)}(z)$ and $\tilde{\underline{\mathbf{B}}}^{(i)}(z)$ create new coefficient matrices at both ends of the matrix $\underline{\mathbf{R}}^{(i-1)}(z)$ to accommodate the shifted coefficients, which results in an increased order of $\underline{\mathbf{R}}^{(i)'}(z)$ compared to $\underline{\mathbf{R}}^{(i-1)}(z)$ from the previous iteration. These new coefficient matrices consist entirely of zeros except for those coefficients positioned in either the k -th row or k -th column of the matrix. A similar problem is encountered with the paraunitary matrix, but its order only grows from either end of the coefficient matrices of $\underline{\mathbf{H}}^{(i-1)}(z)$ depending on how the paraunitary shift operates. The continuous growth of the polynomial order in the para-Hermitian matrix is problematic, as it can lead to a significant increase in the computational complexity of the PEVD algorithm as iterations go on. In other words, this can result in a computationally slow algorithm. Once the algorithm converges, the order of the diagonal matrix $\underline{\mathbf{D}}(z)$ becomes unnecessarily large, and most of the elements in its outer coefficient matrices are zeros, with the remainder accounting for only a small proportion of the Frobenius norm of the input matrix $\underline{\mathbf{R}}(z)$. A worked example will be presented in Chapter 4 to highlight this limitation. In addition, the final paraunitary matrix with large polynomial order can cause costly implementation for some applications, such as paraunitary filter bank based subband coding [21], precoding and equalisation design for broadband MIMO systems [34], convolutive blind source separation [23], spectral factorisation [20], etc.

In general, many of the outer coefficient matrices of $\underline{\mathbf{R}}^{(i)}(z)$ can be truncated after each iteration without seriously compromising the accuracy of the decomposition. By doing this, not only can the PEVD algorithm provide fast convergence but also provides a large reduction of the computational load for the steps of searching for the

dominant off-diagonal element (or the dominant column norm in the SMD algorithm) and the paraunitary rotation. Similarly, truncating the paraunitary matrix can also be conducted after each iteration. We will now review some of the existing polynomial order shortening methods.

3.6.2 Truncation Methods for Para-Hermitian Matrices

This section introduces two existing order shortening methods which can be used to truncate the para-Hermitian matrices obtained from the PEVD algorithms to ensure that the polynomial orders do not grow unnecessarily large. Through the order shortening process, both the computational load and the execution time of the algorithm can be significantly reduced.

Energy Based Truncation

For a para-Hermitian matrix $\underline{\mathbf{R}}(z) = \sum_{\tau=-T/2}^{T/2} \mathbf{R}[\tau]z^{-\tau}$, the energy based truncation method [7, 86] is to find the smallest value for the lag parameter $T_{\text{lim}} > 0$ such that

$$\frac{2 \sum_{\tau=T_{\text{lim}}}^{T/2} \|\mathbf{R}[\tau]\|_{\text{F}}^2}{\|\underline{\mathbf{R}}(z)\|_{\text{F}}^2} \leq \mu, \quad (3.49)$$

where μ defines the proportion of energy (squared Frobenius norm) of $\underline{\mathbf{R}}(z)$ which is allowed to be removed due to the truncation. Once a value for T_{lim} has been found, the truncated para-Hermitian matrix $\underline{\mathbf{R}}_{\text{tr}}(z)$ can be calculated as

$$\underline{\mathbf{R}}_{\text{tr}}(z) = \sum_{\tau=-T_{\text{lim}}+1}^{T_{\text{lim}}-1} \mathbf{R}[\tau]z^{-\tau}. \quad (3.50)$$

Note that due to the para-Hermitian property of $\underline{\mathbf{R}}(z)$, the same number of coefficient matrices must be truncated from both ends of the matrix to ensure that the matrix remains para-Hermitian.

Lag Bound Fixed Truncation

The energy based truncation method would not be a suitable choice for some cases where a fixed bound on the order of the diagonalised para-Hermitian matrix is needed, as the choice of the truncation parameter μ in (3.49) simply cannot guarantee a specific order in advance. To truncate a para-Hermitian matrix $\underline{\mathbf{R}}(z)$ using the lag bound fixed method [67], a suitable lag parameter $L > 0$ has to be chosen to ensure that any outer coefficient matrices $\mathbf{R}[\tau]$ for $|\tau| > L/2$ are removed from $\underline{\mathbf{R}}(z)$, which results in the truncated para-Hermitian matrix given by

$$\underline{\mathbf{R}}_{\text{tr}}(z) = \sum_{\tau=-L/2}^{L/2} \mathbf{R}[\tau]z^{-\tau}. \quad (3.51)$$

It is generally not appropriate to apply this truncation method as part of an iterative routine within the PEVD algorithm, as it is difficult to determine the choice of L in advance. Therefore, this truncation technique is normally used just once after convergence to obtain the prescribed order of the diagonal matrix. Again, it should be noticed that the parameter L must be an even number to ensure that the truncated matrix remains para-Hermitian.

For the simulation results presented in Chapter 4 and Chapter 5, the energy based truncation method will be used. As for the results used to illustrate the potential application of spectral factorisation in Chapter 6, where the PEVD algorithm can be used to convert the multichannel spectral factorisation problem into a set of independent single-channel problems, the lag bound fixed truncation is adopted.

3.6.3 Truncation Methods for Paraunitary Matrices

To reduce the cost of applying the final paraunitary matrix $\underline{\mathbf{H}}(z)$ to polynomial subspace-based applications as mentioned above, we briefly review two existing approaches which will be used in this thesis.

Lag Based Truncation

The first approach for truncating paraunitary matrices [87] follows the idea of the energy based truncation method for para-Hermitian matrices proposed in [7, 86]. However, instead of symmetrically omitting the outer coefficient matrices as for $\underline{\mathbf{R}}(z)$, the lag based truncation method [87] sequentially removes the leading and trailing coefficient matrices from $\underline{\mathbf{H}}(z)$. In other words, the trimming removes coefficient matrices from the front end or back end of $\underline{\mathbf{H}}(z)$, whichever has the smallest Frobenius norm, until the proportion of removed energy reaches a predefined proportion of the total energy in $\underline{\mathbf{H}}(z)$. For a paraunitary matrix $\underline{\mathbf{H}}(z) \in \mathbb{C}^{M \times M \times L}$ with coefficient matrices $\mathbf{H}[\tau]$ for $\tau = 0, \dots, L-1$, the proportion of removed energy is expressed as

$$\kappa_{\text{lag}} = 1 - \frac{\sum_{\tau} \|f_{\text{lag}}(\mathbf{H}[\tau])\|_{\text{F}}^2}{\|\underline{\mathbf{H}}(z)\|_{\text{F}}^2} = 1 - \frac{1}{M} \sum_{\tau} \|f_{\text{lag}}(\mathbf{H}[\tau])\|_{\text{F}}^2, \quad (3.52)$$

where the trim function $f_{\text{lag}}(\mathbf{H}[\tau])$ means removing the leading L_1 and trailing L_2 coefficient matrices from $\underline{\mathbf{H}}(z)$, and it is given by

$$f_{\text{lag}}(\mathbf{H}[\tau]) = \begin{cases} \mathbf{H}[\tau + L_1], & 0 \leq \tau < L - L_1 - L_2 \\ \mathbf{0}, & \text{otherwise} \end{cases}. \quad (3.53)$$

Thus, the truncated paraunitary matrix is expressed as

$$\underline{\mathbf{H}}_{\text{tr}}(z) = \sum_{\tau=L_1}^{L-L_2-1} \mathbf{H}[\tau]z^{-\tau}. \quad (3.54)$$

Here, the lag based truncation was designed to maximise the number of lags removed, $L_1 + L_2$, whilst keeping the proportion of removed energy below an upper bound μ , hence it leads to the constrained optimisation problem:

$$\begin{aligned} & \text{maximise} && (L_1 + L_2) \\ & \text{subject to} && \kappa_{\text{lag}} \leq \mu. \end{aligned} \quad (3.55)$$

Row-Shift Corrected Truncation

The row-shift corrected truncation method [83] is capable of reducing the order of paraunitary matrices by exploiting the paraunitary ambiguity as described in (3.3). This ambiguity permits $\underline{\mathbf{H}}(z)$ to be replaced by $\tilde{\underline{\mathbf{H}}}(z)$, where $\tilde{\underline{\mathbf{H}}}(z) = \underline{\Gamma}(z)\underline{\mathbf{H}}(z)$ and for the truncation purpose $\underline{\Gamma}(z)$ only contains time-shift terms, i.e.,

$$\underline{\Gamma}(z) = \text{diag}\{z^{-\tau_1}, z^{-\tau_2}, \dots, z^{-\tau_M}\}. \quad (3.56)$$

Here the diagonal elements $z^{-\tau_m}$ for $m = 1, \dots, M$ can shift the m -th row of $\underline{\mathbf{H}}(z)$ by τ_m samples, and these row shifts can be used to align the coefficients in each row so that the overall paraunitary matrix can be truncated further.

Let $\underline{\mathbf{h}}_m(z)$, $m = 1, \dots, M$ represent the constituent row vectors of $\underline{\mathbf{H}}(z)$; thus, its paraconjugate is given by

$$\tilde{\underline{\mathbf{H}}}(z) = [\tilde{\underline{\mathbf{h}}}_1(z), \tilde{\underline{\mathbf{h}}}_2(z), \dots, \tilde{\underline{\mathbf{h}}}_M(z)], \quad (3.57)$$

where $\underline{\mathbf{h}}_i(z)\tilde{\underline{\mathbf{h}}}_j(z) = \delta(i-j)$ due to the paraunitary property. For the row-shift corrected truncation method, each row vector $\underline{\mathbf{h}}_m(z)$ is now truncated individually based on

$$f_{\text{shift}}(\underline{\mathbf{h}}_m[\tau]) = \begin{cases} \underline{\mathbf{h}}_m[\tau + L_{1,m}], & 0 \leq \tau < T_m \\ \mathbf{0}, & \text{otherwise} \end{cases}, \quad (3.58)$$

where $T_m = L - L_{1,m} - L_{2,m}$ denotes the length of the truncated vector. $L_{1,m}$ and $L_{2,m}$ are the length of the leading and trailing coefficient vectors being removed from $\underline{\mathbf{h}}_m(z)$ of lag length L . Since $\underline{\mathbf{h}}_m(z)$, $\forall m = 1, \dots, M$ has unit energy, it appears sensible to truncate the same proportion of energy from each vector. Therefore, the proportion of energy removed from $\underline{\mathbf{h}}_m(z)$ is given by

$$\kappa_{\text{shift},m} = 1 - \sum_{\tau} \|f_{\text{shift}}(\underline{\mathbf{h}}_m[\tau])\|_{\text{F}}^2. \quad (3.59)$$

Hence, the optimum truncation based on row-shift correction is given by the following constrained problem:

$$\begin{aligned} & \text{maximise} && \min_m (L_{1,m} + L_{2,m}) \\ & \text{subject to} && \kappa_{\text{shift},m} \leq \frac{\mu'}{M} \quad \forall m = 1, \dots, M, \end{aligned} \tag{3.60}$$

where μ' defines the upper bound of the proportion of energy removed from $\underline{\mathbf{H}}(z)$. Once each row vector has been truncated, the row shifts $\tau_m = L_{1,m}$, $m = 1, \dots, M$ can be identified and applied to align the truncation via $\underline{\Gamma}(z)$ in (3.56).

The order of the truncated paraunitary matrix $\underline{\mathbf{H}}_{\text{tr}}(z)$ will be equal to the order of the truncated row vector whose length is the maximum, i.e., $\max_m T_m$, which is determined by $\min_m (L_{1,m} + L_{2,m})$ in (3.60). In effect, the process of truncating each individual row vector in $\underline{\mathbf{H}}(z)$ is equivalent to the lag based truncation method [87]. These two truncation method for paraunitary matrices will be applied to different PEVD algorithms to examine their performances, and results will be presented in Chapter 5.

3.7 Chapter Summary

In this chapter, we have briefly reviewed the state-of-the-art algorithms for calculating the PEVD of para-Hermitian matrices. These algorithms are categorised into two families, i.e., the SBR2 family and the SMD family. In particular, a first proof of the spectral majorisation property in SBR2 has been presented based on the idea of maximising the coding gain. The monotonically increasing behaviour of the coding gain has been exploited to obtain a more reliable test of convergence for the SBR2 algorithm. Furthermore, different polynomial order truncation methods are introduced to address the order growth problem in the polynomial matrices produced by the PEVD algorithms.

Chapter 4

Multiple Shift SBR2 Algorithm for Polynomial Eigenvalue Decomposition

4.1 Introduction

As mentioned in Chapter 3, the sequential matrix diagonalisation (SMD) family, including the original SMD [35], maximum element SMD (ME-SMD) [35], and multiple shift ME-SMD (MSME-SMD) [30] algorithms, are designed to achieve better diagonalisation than the SBR2 algorithm at the expense of higher computational complexity. Therefore, the aim of this chapter is to see whether some of the ideas in MSME-SMD can be harnessed to create a version of SBR2 that converges more rapidly while still enjoying the low complexity benefit of SBR2.

In this chapter, we propose an improved version of the SBR2 algorithm for calculating the eigenvalue decomposition (EVD) of para-Hermitian matrices. The improved algorithm is entitled multiple shift SBR2 (MS-SBR2) and is developed based on the original SBR2 algorithm. It is capable of performing the diagonalisation faster than the conventional SBR2 algorithm by means of transferring more off-diagonal energy onto the diagonal at each iteration; this is extremely important in some real-time applications as mentioned previously. Furthermore, the MS-SBR2 algorithm seems to

be more accurate in term of reconstructing the para-Hermitian matrix from the inverse decomposition $\hat{\mathbf{R}}(z) = \tilde{\mathbf{H}}(z)\mathbf{D}(z)\mathbf{H}(z)$ in our studied numerical examples. The convergence of the MS-SBR2 algorithm has been proven akin to that of the SBR2 algorithm.

We have also investigated two different time-shift strategies which arise in the MS-SBR2 algorithm, including the conventional (k -constrained) and direction-fixed time-shifts. For the computer simulations, we will first demonstrate how the proposed MS-SBR2 algorithm performs for each of the time-shift methods using numerical examples, and results will be compared to that of the original SBR2 algorithm based on several performance metrics. Furthermore, we will investigate how different source models can impact the proposed MS-SBR2 algorithm based on defining a dynamic range of the underlying sources, and different relations between the source signals' power spectral densities. Last but not least, an applied example will be presented to show how the MS-SBR2 algorithm can be used to strongly decorrelate a set of convolutively mixed sensor signals and how it performs compared to the original SBR2 algorithm.

4.2 Outline of the MS-SBR2 Algorithm

The MS-SBR2 algorithm was developed based on the SBR2 algorithm, but with an additional off-diagonal energy transferred onto the diagonal by means of a “multiple shift” in every iteration step, which is akin to the MSME-SMD algorithm. By shifting multiple elements, the MS-SBR2 algorithm can achieve the diagonalisation with fewer iterations than SBR2.

Given a para-Hermitian matrix $\mathbf{R}(z) \in \mathbb{C}^{M \times M}$, at each iteration the MS-SBR2 algorithm can bring at most $\lfloor M/2 \rfloor$ off-diagonal elements onto the zero-lag plane, which then can be annihilated by a sequence of Jacobi rotations. The reason $\lfloor M/2 \rfloor$ elements at most can be transferred at each iteration is that each Jacobi rotation affects both rows j and k and columns j and k . In other words, the search space of the next maximum off-diagonal element has to exclude these two rows and two columns in order

4.2 Outline of the MS-SBR2 Algorithm

to avoid affecting previous maxima. Note that for the matrix dimension parameter $M \leq 3$, the MS-SBR2 algorithm simply reduces to the SBR2 algorithm. Details about the search strategy will be discussed in the following section. There are two main stages involved at each iteration in MS-SBR2. The first stage aims to bring as many as $\lfloor M/2 \rfloor$ off-diagonal elements to the zero-lag plane by using the established time-shift method, and the second stage is to apply a sequence of Jacobi rotations to the zero-lag matrix $\mathbf{R}^{(i)'}[0]$ in order to eliminate those elements brought to the plane at the first stage. Based on the formulation described in (3.8) the elementary paraunitary matrix in MS-SBR2, therefore, can be defined as

$$\begin{aligned}\underline{\hat{\mathbf{G}}}^{(i)}(z) &= \mathbf{Q}^{(L^{(i)},i)} \mathbf{Q}^{(L^{(i)-1},i)} \dots \mathbf{Q}^{(1,i)} \underline{\mathbf{B}}^{(L^{(i)},i)}(z) \underline{\mathbf{B}}^{(L^{(i)-1},i)}(z) \dots \underline{\mathbf{B}}^{(1,i)}(z) \\ &= \hat{\mathbf{Q}}^{(i)} \hat{\underline{\mathbf{B}}}^{(i)}(z),\end{aligned}\quad (4.1)$$

where $\hat{\mathbf{Q}}^{(i)} = \prod_{l=1}^{L^{(i)}} \mathbf{Q}^{(l,i)}$, $\hat{\underline{\mathbf{B}}}^{(i)}(z) = \prod_{l=1}^{L^{(i)}} \underline{\mathbf{B}}^{(l,i)}(z)$, and $L^{(i)} \leq \lfloor M/2 \rfloor$ denotes the total number of off-diagonal elements shifted to the zero-lag coefficient matrix at the i -th iteration. Accordingly, the resulting para-Hermitian matrix at this iteration can be computed by performing the paraunitary similarity transformations as

$$\underline{\mathbf{R}}^{(i)'}(z) = \hat{\underline{\mathbf{B}}}^{(i)}(z) \underline{\mathbf{R}}^{(i-1)}(z) \tilde{\hat{\underline{\mathbf{B}}}}^{(i)}(z) \quad (4.2)$$

and

$$\underline{\mathbf{R}}^{(i)}(z) = \hat{\mathbf{Q}}^{(i)} \underline{\mathbf{R}}^{(i)'}(z) \hat{\mathbf{Q}}^{(i)H}. \quad (4.3)$$

Similar to SBR2, the MS-SBR2 algorithm converges when the stopping condition suffices, i.e., the magnitude of the maximum off-diagonal element $|r_{jk}^{(1,i)}[\tau]|$, found at the i -th iteration with a full search space, is smaller than the pre-defined threshold ε . Assuming the algorithm stops at the I -th iteration, the generated paraunitary matrix can be calculated as

$$\underline{\mathbf{H}}^{(I)}(z) = \hat{\underline{\mathbf{G}}}^{(I)}(z) \hat{\underline{\mathbf{G}}}^{(I-1)}(z) \dots \hat{\underline{\mathbf{G}}}^{(1)}(z). \quad (4.4)$$

4.2 Outline of the MS-SBR2 Algorithm

The MS-SBR2 algorithm is summarised in Algorithm 2.

Algorithm 2 The MS-SBR2 Algorithm

Input: para-Hermitian matrix $\mathbf{R}(z) \in \mathbb{C}^{M \times M}$, maximum number of iterations I , convergence parameter ε , and truncation parameter μ

Output: diagonalised matrix $\mathbf{D}(z)$ and the paraunitary matrix $\mathbf{H}(z)$

- 1: **Initialisation:** $iter \leftarrow 0$, $g \leftarrow 1 + \varepsilon$, $\mathbf{H}^{(0)}(z) \leftarrow \mathbf{I}_M$, and $\mathbf{R}^{(0)}(z) \leftarrow \mathbf{R}(z)$
- 2: **while** $iter < I$ and $g > \varepsilon$ **do**
- 3: % Locate and time-shift multiple off-diagonal elements
- 4: **Initialisation:** the total number of off-diagonal elements shifted at the i -th iteration, $L^{(i)} \leftarrow 0$
- 5: **while** $g > \varepsilon$ and the search space exists **do**
- 6: locate the maximum off-diagonal element $r_{jk}^{(i)}[\tau]$ in the current search space;
- 7: $g \leftarrow |r_{jk}^{(i)}[\tau]|$;
- 8: **if** $g > \varepsilon$ **then**
- 9: $\mathbf{R}^{(i)'}(z) \leftarrow \mathbf{B}^{(i)}(z)\mathbf{R}^{(i-1)}(z)\tilde{\mathbf{B}}^{(i)}(z)$; $\mathbf{H}^{(i)'}(z) \leftarrow \mathbf{B}^{(i)}(z)\mathbf{H}^{(i-1)}(z)$
 according to (3.5)
- 10: $L^{(i)} \leftarrow L^{(i)} + 1$;
- 11: **end if**
- 12: update the search space for remaining off-diagonal elements;
- 13: **end while**
- 14: % Perform a sequence of Jacobi rotations
- 15: **for** $l = 1 : L^{(i)}$ **do**
- 16: $\mathbf{R}^{(i)}(z) \leftarrow \mathbf{Q}^{(l,i)}\mathbf{R}^{(i)'}(z)\mathbf{Q}^{(l,i)H}$; $\mathbf{H}^{(i)}(z) \leftarrow \mathbf{Q}^{(l,i)}\mathbf{H}^{(i)'}(z)$ according to (4.3)
- 17: **end for**
- 18: $iter \leftarrow iter + 1$;
- 19: % Trim polynomial matrix order
- 20: $\mathbf{R}^{(i)}(z) \leftarrow \text{TrimPH}(\mathbf{R}^{(i)}(z), \mu)$; $\mathbf{H}^{(i)}(z) \leftarrow \text{TrimPU}(\mathbf{H}^{(i)}(z), \mu)$
- 21: **end while**

The search strategy for MS-SBR2 differs from the one employed in SBR2, and it is based on a set of reduced search spaces. As each Jacobi rotation operation will act on both the columns and rows j and k of the maximum off-diagonal element $r_{jk}^{(i)'}[0]$ at the zero-lag plane, the search space for the next off-diagonal element will have to exclude those two columns and two rows to avoid affecting the previous off-diagonal element.

Assuming the input para-Hermitian matrix has dimension 6×6 , the first off-diagonal element at the i -th iteration can be located according to (3.13). Once the first element a and its complex conjugate a^* have been shifted to the zero-lag matrix as shown in

4.3 Proof of Convergence

Figure 4.1(a), the grey areas shown in Figure 4.1(b) will be eliminated from the search space of the next off-diagonal element, which leads the white parts of the upper triangular area to be the search space of the second element. By continuing from Figure 4.1(b), if the second element b was found at row 3 and column 6, its complex conjugate b^* will be at row 6 and column 3 according to the para-Hermitian property. After bringing them to the zero-lag matrix as shown in Figure 4.1(c), the search space for the third element c^* will be the remaining position row 1 and column 5 shown in Figure 4.1(d). This search strategy applies to a general $M \times M$ para-Hermitian matrix. Generally speaking, there are at most $\lfloor M/2 \rfloor$ off-diagonal elements which can be located at each iteration. Therefore, in the case of $M \leq 3$, the MS-SBR2 algorithm is identical to the SBR2 algorithm. In other words, if there is only one off-diagonal element shifted to the zero-lag coefficient matrix at each iteration, i.e., $L^{(i)} = 1$, then MS-SBR2 reduces to SBR2.

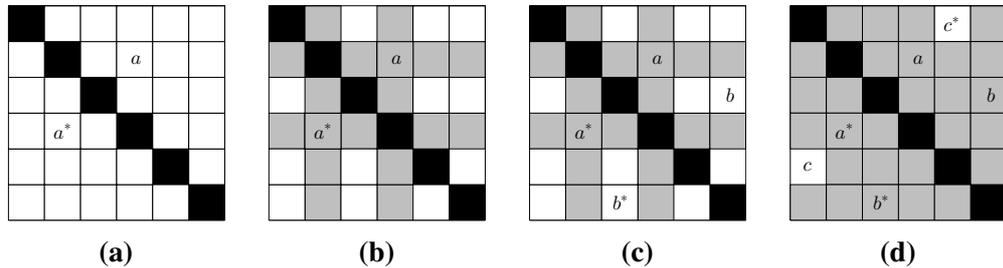


Figure 4.1 Illustration of the search space for a 6×6 para-Hermitian matrix example in the MS-SBR2 algorithm showing (a) the location of the first maximum off-diagonal element a and its conjugate a^* , (b) the reduced search space (marked in white tiles), (c) the location of the second pairwise off-diagonal elements b and b^* , and (d) the location of the last pairwise off-diagonal element c and c^* .

4.3 Proof of Convergence

Theorem 1 (Convergence of the MS-SBR2 Algorithm): With a sufficiently large number of iterations I , the MS-SBR2 algorithm can approximately diagonalise the para-

4.3 Proof of Convergence

Hermitian matrix $\mathbf{R}(z) \in \mathbb{C}^{M \times M}$ and reduce the energy of the off-diagonal elements to an arbitrarily low threshold $\psi > 0$.

Proof: The proof of *Theorem 1* is very similar to that of the original SBR2 algorithm [7]. To prove *Theorem 1*, we first define the following quantities:

$$\mathcal{N}_1\{\mathbf{R}^{(i)}(z)\} \triangleq \sum_{m=1}^M |r_{mm}^{(i)}[0]|^2, \quad (4.5)$$

$$\mathcal{N}_2\{\mathbf{R}^{(i)}(z)\} \triangleq \|\mathbf{R}^{(i)}[0]\|_{\mathbb{F}}^2, \quad (4.6)$$

$$\mathcal{N}_3\{\mathbf{R}^{(i)}(z)\} \triangleq \mathcal{N}_2\{\mathbf{R}^{(i)}(z)\} - \mathcal{N}_1\{\mathbf{R}^{(i)}(z)\}, \quad (4.7)$$

$$\mathcal{N}_4\{\mathbf{R}^{(i)}(z)\} \triangleq \|\mathbf{R}^{(i)}(z)\|_{\mathbb{F}}^2 = \sum_{\tau} \|\mathbf{R}^{(i)}[\tau]\|_{\mathbb{F}}^2. \quad (4.8)$$

The equations above denote, respectively, the squares of: the trace norm at the zero-lag ($\tau = 0$) of the transformed polynomial matrix $\mathbf{R}^{(i)}(z)$; the Frobenius norm at $\tau = 0$; the off-diagonal F-norm at $\tau = 0$; and the Frobenius norm of $\mathbf{R}^{(i)}(z)$. Note that $\mathcal{N}_1\{\cdot\}$ is invariant under the paraunitary delay operations, as in (4.2), i.e.,

$$\mathcal{N}_1\{\mathbf{R}^{(i)'}(z)\} = \mathcal{N}_1\{\hat{\mathbf{B}}^{(i)}(z)\mathbf{R}^{(i-1)}(z)\tilde{\mathbf{B}}^{(i)}(z)\} = \mathcal{N}_1\{\mathbf{R}^{(i-1)}(z)\}. \quad (4.9)$$

Further, $\mathcal{N}_2\{\cdot\}$ is invariant under the Jacobi rotations in (4.3), i.e.,

$$\mathcal{N}_2\{\mathbf{R}^{(i)}(z)\} = \mathcal{N}_2\{\hat{\mathbf{Q}}^{(i)}\mathbf{R}^{(i)'}(z)\hat{\mathbf{Q}}^{(i)\text{H}}\} = \mathcal{N}_2\{\mathbf{R}^{(i)'}(z)\}. \quad (4.10)$$

Due to the multiple-shift effect in MS-SBR2, the off-diagonal elements $r_{jk}^{(l,i)}[\tau]$ for $l = 1, \dots, L^{(i)}$ found at the i -th iteration will all be brought onto the zero-lag matrix. Thus, the norm of these elements is given by

$$\gamma^{(i)} = \sum_{l=1}^{L^{(i)}} |r_{jk}^{(l,i)}[\tau]|^2. \quad (4.11)$$

By performing the paraunitary delay step in (4.2) and the rotation step in (4.3), the total energy transferred onto the diagonal of the zero-lag matrix $\mathbf{R}^{(i)}[0]$ will be $2\gamma^{(i)}$,

such that

$$\mathcal{N}_3\{\underline{\mathbf{R}}^{(i)'}(z)\} \geq 2\gamma^{(i)}, \quad (4.12)$$

and

$$\mathcal{N}_1\{\underline{\mathbf{R}}^{(i)}(z)\} = \mathcal{N}_1\{\underline{\mathbf{R}}^{(i)'}(z)\} + 2\gamma^{(i)} = \mathcal{N}_1\{\underline{\mathbf{R}}^{(i-1)}(z)\} + 2\gamma^{(i)}. \quad (4.13)$$

Since $\gamma^{(i)} > 0$, $\mathcal{N}_1\{\underline{\mathbf{R}}^{(i)}(z)\}$ increases monotonically with each iteration. Also, due to

$$\mathcal{N}_1\{\underline{\mathbf{R}}^{(i)}(z)\} \leq \mathcal{N}_4\{\underline{\mathbf{R}}^{(i)}(z)\} \quad \forall i, \quad (4.14)$$

and the overall energy, $\mathcal{N}_4\{\underline{\mathbf{R}}^{(i)}(z)\}$, is invariant over iterations. There must exist a supremum S for $\mathcal{N}_1\{\underline{\mathbf{R}}^{(i)}(z)\}$, such that

$$S = \sup_i \mathcal{N}_1\{\underline{\mathbf{R}}^{(i)}(z)\}. \quad (4.15)$$

It follows that for any $\psi > 0$ there must be an iteration number I for which $S - \mathcal{N}_1\{\underline{\mathbf{R}}^{(I)}(z)\} < \psi$, so the increase $2\gamma^{(I+i)}$, $i > 0$, at any subsequent stages must satisfy

$$2\gamma^{(I+i)} \leq S - \mathcal{N}_1\{\underline{\mathbf{R}}^{(I)}(z)\} < \psi. \quad (4.16)$$

Hence, for any $\psi > 0$, there must be an iteration I at which $\gamma^{(I)}$ is bounded by ψ . ■

4.4 Time-Shift Strategies of the MS-SBR2 Algorithm

This section introduces two different ways of operating the paraunitary shift step in the MS-SBR2 algorithm.

4.4.1 The Conventional Time-Shift Method

At each elementary delay stage of MS-SBR2, the conventional time-shift method [28] operates by shifting the $k^{(l,i)}$ -th row of $\underline{\mathbf{R}}^{(l,i)}(z)$ in either the positive ($\tau^{(l,i)} > 0$) or

negative ($\tau^{(l,i)} < 0$) lag direction therefore shifting the $k^{(l,i)}$ -th column to the opposite direction. This idea of the conventional time-shift method is summarised in Algorithm 3. For a more intuitive explanation, a 3D illustration of the MS-SBR2 algorithm with the conventional time-shift method is given in Figure 4.2, where a 5×5 para-Hermitian matrix example is used to demonstrate the procedure of a single iteration of the conventional MS-SBR2 algorithm.

Algorithm 3 Conventional Time-Shift Method for MS-SBR2

Input: para-Hermitian matrix $\underline{\mathbf{R}}^{(i-1)}(z)$, paraunitary matrix $\underline{\mathbf{H}}^{(i-1)}(z)$, and location indices $\{k^{(l,i)}, \tau^{(l,i)}\}$
Output: shifted para-Hermitian matrix $\underline{\mathbf{R}}^{(i)'}(z)$ and paraunitary matrix $\underline{\mathbf{H}}^{(i)'}(z)$

- 1: **Initialisation:** $\underline{\mathbf{R}}^{(1,i)}(z) \leftarrow \underline{\mathbf{R}}^{(i-1)}(z)$, $\underline{\mathbf{H}}^{(1,i)}(z) \leftarrow \underline{\mathbf{H}}^{(i-1)}(z)$
- 2: **for** $l = 1 : L^{(i)}$ **do**
- 3: **if** $\tau^{(l,i)} > 0$ **then**
- 4: shift the $k^{(l,i)}$ -th row of $\underline{\mathbf{R}}^{(l,i)}(z)$ and $\underline{\mathbf{H}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the positive lag direction;
- 5: shift the $k^{(l,i)}$ -th column of $\underline{\mathbf{R}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the negative lag direction;
- 6: **else if** $\tau^{(l,i)} < 0$ **then**
- 7: shift the $k^{(l,i)}$ -th row of $\underline{\mathbf{R}}^{(l,i)}(z)$ and $\underline{\mathbf{H}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the negative lag direction;
- 8: shift the $k^{(l,i)}$ -th column of $\underline{\mathbf{R}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the positive lag direction;
- 9: **else**
- 10: $\underline{\mathbf{R}}^{(l,i)}(z) \leftarrow \underline{\mathbf{R}}^{(l,i)}(z)$, $\underline{\mathbf{H}}^{(l,i)}(z) \leftarrow \underline{\mathbf{H}}^{(l,i)}(z)$
- 11: **end if**
- 12: **end for**
- 13: $\underline{\mathbf{R}}^{(i)'}(z) \leftarrow \underline{\mathbf{R}}^{(L^{(i)},i)}(z)$, $\underline{\mathbf{H}}^{(i)'}(z) \leftarrow \underline{\mathbf{H}}^{(L^{(i)},i)}(z)$

In some cases, the directions of row (column) shifts at different delay stages within one iteration might be different, which will result in some non-zero elements being shifted further away from the zero-lag plane and causing unnecessary order growth of the polynomial matrices.

4.4.2 The Direction-Fixed Time-Shift Method

To avoid the unnecessary polynomial order growth imposed by the conventional time-shift method, this section presents a new method of controlling the order growth of

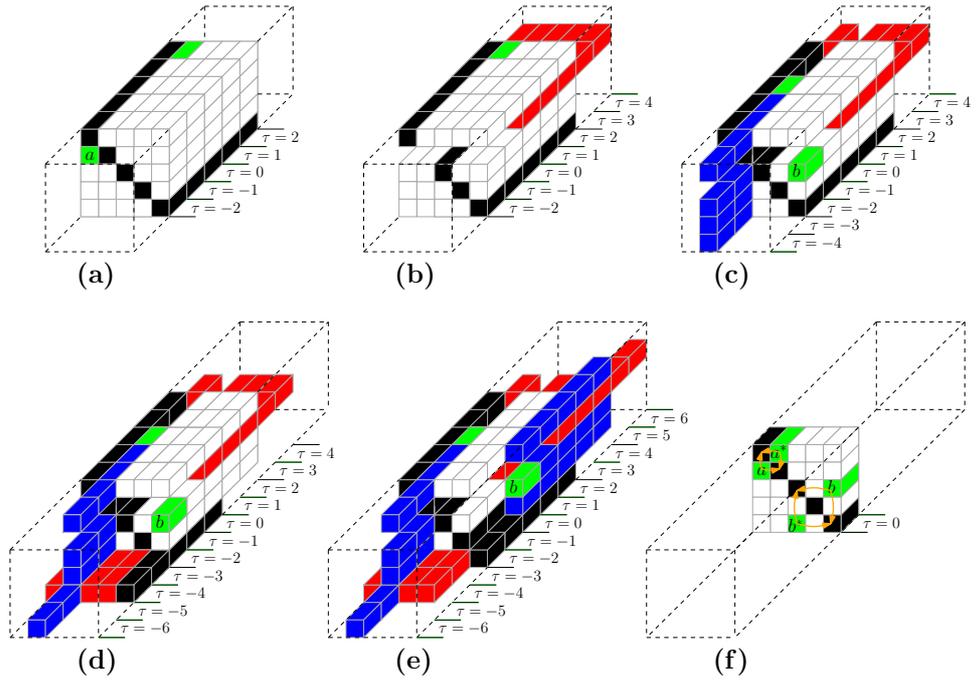


Figure 4.2 A 3D illustration of the MS-SBR2 algorithm with the conventional time-shift method for diagonalising a 5×5 example para-Hermitian matrix, showing a single iteration of (a) the first maximum off-diagonal element a^* (marked in green colour) found at the location of $\{1, 2, 2\}$; (b) time-shifting row 2 (marked in red colour) towards the positive-lag direction by 2 lags; (c) time-shifting column 2 (marked in blue colour) towards the negative-lag direction by 2 lags and the second off-diagonal element b found at the location of $\{3, 5, -2\}$; (d) time-shifting row 5 towards negative-lag direction by 2 lags; (e) time-shifting column 5 towards the positive-lag direction by 2 lags; and (f) performing a sequence of Jacobi rotations to transfer the off-diagonal elements ($\tau = 0$) onto the diagonal. Note that the generated paraunitary rotation matrix is also applied to the remaining lags.

polynomial matrices in the MS-SBR2 algorithm. In effect, the proposed method introduces a new elementary delay strategy which keeps all the row (column) shifts in the same direction throughout each iteration, thereby giving us the flexibility to control the polynomial order growth by selecting shifts that ensure non-zero coefficients are kept closer to the zero-lag plane. By restricting the direction of all row (column) shifts throughout iterations, there will be no interference between the subsequent delay stages. All the zero-filled outer lags of polynomial matrices can be precisely tracked and removed without compromising the diagonalisation performance and accuracy of

the algorithm. Thus, the whole scheme is named order-controlled MS-SBR2 (OCMS-SBR2) algorithm.

For the results presented in this thesis, the direction of all the row shifts is confined towards the positive time lag, while the direction of all the column shifts is towards the negative time lag. A summary of the direction-fixed delay strategy is shown in Algorithm 4. Accordingly, a 3D illustration of the OCMS-SBR2 algorithm is given in Figure 4.3.

Algorithm 4 Direction-Fixed Time-Shift Method for MS-SBR2

Input: para-Hermitian matrix $\underline{\mathbf{R}}^{(i-1)}(z)$, paraunitary matrix $\underline{\mathbf{H}}^{(i-1)}(z)$, and location indices $\{j^{(l,i)}, k^{(l,i)}, \tau^{(l,i)}\}$

Output: shifted para-Hermitian matrix $\underline{\mathbf{R}}^{(i)'}(z)$ and paraunitary matrix $\underline{\mathbf{H}}^{(i)'}(z)$

- 1: **Initialisation:** $\underline{\mathbf{R}}^{(1,i)}(z) \leftarrow \underline{\mathbf{R}}^{(i-1)}(z)$, $\underline{\mathbf{H}}^{(1,i)}(z) \leftarrow \underline{\mathbf{H}}^{(i-1)}(z)$
- 2: **for** $l = 1 : L^{(i)}$ **do**
- 3: **if** $\tau^{(l,i)} > 0$ **then**
- 4: shift the $k^{(l,i)}$ -th row of $\underline{\mathbf{R}}^{(l,i)}(z)$ and $\underline{\mathbf{H}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the positive lag direction;
- 5: shift the $k^{(l,i)}$ -th column of $\underline{\mathbf{R}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the negative lag direction;
- 6: **else if** $\tau^{(l,i)} < 0$ **then**
- 7: shift the $j^{(l,i)}$ -th row of $\underline{\mathbf{R}}^{(l,i)}(z)$ and $\underline{\mathbf{H}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the positive lag direction;
- 8: shift the $j^{(l,i)}$ -th column of $\underline{\mathbf{R}}^{(l,i)}(z)$ by $|\tau^{(l,i)}|$ lags towards the negative lag direction;
- 9: **else**
- 10: $\underline{\mathbf{R}}^{(l,i)}(z) \leftarrow \underline{\mathbf{R}}^{(l,i)}(z)$, $\underline{\mathbf{H}}^{(l,i)}(z) \leftarrow \underline{\mathbf{H}}^{(l,i)}(z)$
- 11: **end if**
- 12: **end for**
- 13: $\underline{\mathbf{R}}^{(i)'}(z) \leftarrow \underline{\mathbf{R}}^{(L^{(i)},i)}(z)$, $\underline{\mathbf{H}}^{(i)'}(z) \leftarrow \underline{\mathbf{H}}^{(L^{(i)},i)}(z)$

Assuming no order truncation scheme is involved when using the OCMS-SBR2 algorithm for computing the PEVD, the order growth on $\underline{\mathbf{R}}^{(i)}(z)$ and $\underline{\mathbf{H}}^{(i)}(z)$ are now bounded by the maximum modulus of the delay $|\tau^{(l_{\max},i)}|$ at the i -th iteration, where

$$l_{\max} = \arg \max_l |\tau^{(l,i)}|, \forall l = 1 \cdots L^{(i)} \quad , \quad (4.17)$$

and $|\tau^{(l,i)}|$ denotes the modulus of the delay needed to bring the maximum element onto the zero-lag at the l -th time-shift step within the i -th iteration. With zero-filled

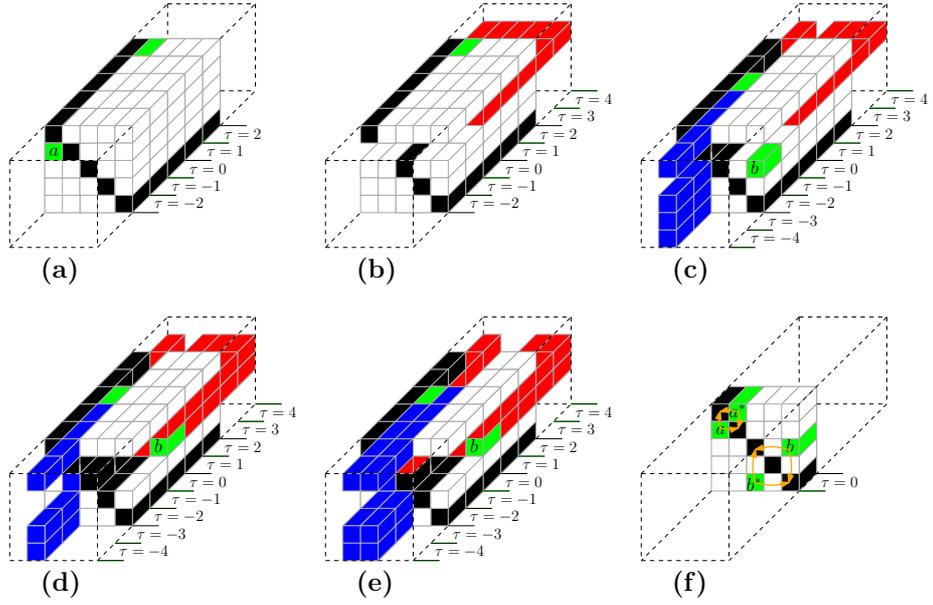


Figure 4.3 A 3D illustration of the MS-SBR2 algorithm with the direction-fixed time-shift method for diagonalising a 5×5 example para-Hermitian matrix, showing a single iteration of (a) the first maximum off-diagonal element a^* (marked in green colour) found at the location of $\{1, 2, 2\}$; (b) time-shifting row 2 (marked in red colour) towards the positive-lag direction by 2 lags; (c) time-shifting column 2 (marked in blue colour) towards the negative-lag direction by 2 lags and the second off-diagonal element b found at the location of $\{3, 5, -2\}$; (d) time-shifting row 3 towards the positive-lag direction by 2 lags; (e) time-shifting column 3 towards the negative-lag direction by 2 lags; and (f) performing a sequence of Jacobi rotations to transfer the off-diagonal elements ($\tau = 0$) onto the diagonal. Note that the generated paraunitary rotation matrix is also applied to the remaining lags.

outer matrices being removed, the final polynomial orders can be estimated as

$$\begin{aligned}
 O_{\text{PH}}^{(I)} &= O_{\text{PH}}^{(0)} + 2 \sum_{i=1}^I |\tau^{(l_{\max}, i)}|, \\
 O_{\text{PU}}^{(I)} &= \sum_{i=1}^I |\tau^{(l_{\max}, i)}|,
 \end{aligned} \tag{4.18}$$

where $O_{\text{PH}}^{(I)}$ denotes the order of the transformed para-Hermitian matrix $\mathbf{R}^{(I)}(z)$ at the I -th iteration, with its initial order given by $O_{\text{PH}}^{(0)}$, and $O_{\text{PU}}^{(I)}$ denotes the order of the corresponding paraunitary matrix $\mathbf{H}^{(I)}(z)$. Bear in mind that the benefit of using the direction-fixed delay strategy can only be reflected when $L^{(i)} \geq 2, \exists i = 1, 2, \dots, I$, meaning that in order to make the most out of the direction-fixed shift strategy in

the OCMS-SBR2 algorithm, there must exist at least one iteration at which two or more shifts occur. For example, given a para-Hermitian matrix $\underline{\mathbf{R}}(z)$ with the matrix dimension $M = 6$, the best-case scenario is that there are $\lfloor M/2 \rfloor = 3$ off-diagonal elements which can be shifted and annihilated at each iteration.

Due to the common nature of how the PEVD algorithms operate, the transformed para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$ at the i -th iteration usually has highly sparse outer coefficient matrices which generally account for a small proportion of the total energy of the input para-Hermitian matrix $\underline{\mathbf{R}}(z)$. To truncate the negligibly small amount of energy and also to reduce the computational complexity, the para-Hermitian [86] and paraunitary [87] truncation approaches are respectively applied to $\underline{\mathbf{R}}^{(i)}(z)$ and $\underline{\mathbf{H}}^{(i)}(z)$ with pre-defined truncation parameters μ_{PH} and μ_{PU} , whose values indicate the proportion of the total energy of $\underline{\mathbf{R}}(z)$ and $\underline{\mathbf{H}}(z)$ to be truncated. Further details about how the OCMS-SBR2 algorithm performs after introducing the truncation schemes will be presented in the following sections.

4.5 Numerical Examples

To demonstrate the performance of these two different time-shift strategies as mentioned in Algorithms 3 and 4, numerical examples are designed to evaluate the corresponding versions of MS-SBR2, including the conventional MS-SBR2 and OCMS-SBR2 algorithms in terms of the chosen performance metrics, which will now be discussed. Simulation results are also benchmarked against the original SBR2 algorithm.

4.5.1 Performance Metrics

Diagonalisation Measure

To monitor the convergence of the PEVD algorithms, one of the important measures is the normalised off-diagonal energy $\eta^{(i)}$ at the i -th iteration,

$$\eta^{(i)} = \frac{\sum_{\tau} \sum_{m \neq n}^M |r_{mn}^{(i)}[\tau]|^2}{\sum_{\tau} \|\mathbf{R}[\tau]\|_{\text{F}}^2}, \quad m, n = 1, 2, \dots, M, \quad (4.19)$$

where the numerator denotes the off-diagonal energy of the transformed para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$ at the i -th iteration and the denominator denotes the energy of the input para-Hermitian matrix $\underline{\mathbf{R}}(z)$. Compared to the convergence metric g (magnitude of the maximum off-diagonal element) for the SBR2 algorithm as discussed in Chapter 3, this $\eta^{(i)}$ metric is more commonly used across all different PEVD algorithms as it can effectively reflect how much off-diagonal energy remains in $\underline{\mathbf{R}}^{(i)}(z)$ after each iteration. For this reason, this performance metric will also be adopted for the results presented in the following chapters.

Para-Hermitian Matrix Reconstruction Error

To assess the accuracy of the decomposition performed by the PEVD algorithms, especially when the truncation methods in Section 3.6.2 are employed, the para-Hermitian matrix reconstruction error can be used as the performance metric, which is defined as

$$\xi_{\text{PH}} = \frac{\|\underline{\mathbf{R}}(z) - \hat{\underline{\mathbf{R}}}(z)\|_{\text{F}}^2}{\|\underline{\mathbf{R}}(z)\|_{\text{F}}^2}, \quad (4.20)$$

where $\underline{\mathbf{R}}(z)$ is the input para-Hermitian matrix and $\hat{\underline{\mathbf{R}}}(z)$ denotes the reconstructed matrix obtained from the inverse transformation $\hat{\underline{\mathbf{R}}}(z) = \tilde{\underline{\mathbf{H}}}(z)\underline{\mathbf{D}}(z)\underline{\mathbf{H}}(z)$.

Paraunitary Matrix Reconstruction Error

The paraunitary property, i.e. $\underline{\mathbf{H}}(z)\tilde{\underline{\mathbf{H}}}(z) = \mathbf{I}_M$, is lost after applying the order truncation as mentioned in Section 3.6.3. Considering $\underline{\mathbf{H}}(z)$ as a filter bank, the loss manifests itself as reconstruction error [69], and the difference to a paraunitary system can be defined as

$$\underline{\Phi}(z) = \mathbf{I}_M - \underline{\mathbf{H}}_{\text{tr}}(z)\tilde{\underline{\mathbf{H}}}_{\text{tr}}(z), \quad (4.21)$$

where $\underline{\mathbf{H}}_{\text{tr}}(z)$ denotes the truncated paraunitary matrix. Thus, the loss of the paraunitarity can be calculated as

$$\xi_{\text{PU}} = \frac{1}{M} \|\underline{\Phi}(z)\|_{\text{F}}^2 = \frac{1}{M} \sum_{\tau} \|\Phi[\tau]\|_{\text{F}}^2, \quad (4.22)$$

where M is the matrix dimension parameter, and the coefficient $\frac{1}{M}$ is used for normalisation purposes based on the fact that the squared Frobenius norm of any paraunitary matrices $\underline{\mathbf{H}}(z) \in \underline{\mathbb{C}}^{M \times M}$ is equal to M .

4.5.2 Simulation Scenario

To assess the proposed MS-SBR2 algorithms based on the performance metrics mentioned above, the diagonalisation is conducted using Monte Carlo simulations over an ensemble of 2000 random para-Hermitian matrices $\underline{\mathbf{R}}(z)$ of order 4. Each instance of $\underline{\mathbf{R}}(z)$ is generated as $\underline{\mathbf{R}}(z) = \underline{\mathbf{A}}(z)\tilde{\underline{\mathbf{A}}}(z)$, where $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{6 \times 6}$ is a random polynomial matrix of order 2 with independent and identically distributed (i.i.d.) zero mean unit variance complex Gaussian entries. Each algorithm is set to run for 100 iterations, with the performance metrics recorded after every iteration. The experiments are carried out for two different cases: (i) no truncation method is used during iterations and (ii) the energy based truncation method [86] is applied to the transformed para-Hermitian matrix at each iteration, whilst the lag based truncation method [87] is applied to the

corresponding paraunitary transformation matrix, both with the truncation parameters $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-4}$.

4.5.3 Algorithm Convergence

The diagonalisation performance versus iterations is shown in Figure 4.4(a) for the different versions of SBR2. Clearly, using the multiple-shift operation, both versions of MS-SBR2 converge significantly faster than the original SBR2 algorithm. However, it should be noted that both MS-SBR2 and OCMS-SBR2 require $M/2 = 3$ times as many Jacobi rotations as SBR2 needs at each iteration, as shown in Figure 4.4(b), which makes the computational loads higher than that of SBR2. Figure 4.4(c) presents the normalised remaining off-diagonal energy as a function of the time taken to calculate the decomposition ¹. Compared to SBR2, the MS-SBR2 algorithms take more time to execute 100 iterations. However, for a specific level of diagonalisation, the MS-SBR2 algorithms appear to run much faster than SBR2. Furthermore, using the direction fixed shift method, the OCMS-SBR2 algorithm requires less execution time than does the conventional MS-SBR2, which uses the k -constrained shift method as mentioned in Section 4.4. Note that to make a fair comparison, the execution time is measured without using any truncation methods. Overall, we can see that the two different time-shift strategies used in MS-SBR2 have no impact on the diagonalisation performance, and the OCMS-SBR2 algorithm outperforms the conventional MS-SBR2 algorithm in terms of the computation time.

4.5.4 Polynomial Matrix Order

Case 1: No Truncation

Without using any truncation methods, the average polynomial orders of $\underline{\mathbf{H}}^{(i)}(z)$ and $\underline{\mathbf{R}}^{(i)}(z)$ versus iterations, obtained from different versions of SBR2, are illustrated in

¹Computations undertaken on a PC with Intel(R) Core(TM) i7-3770T CPU @ 2.50GHz and 16 GB RAM.

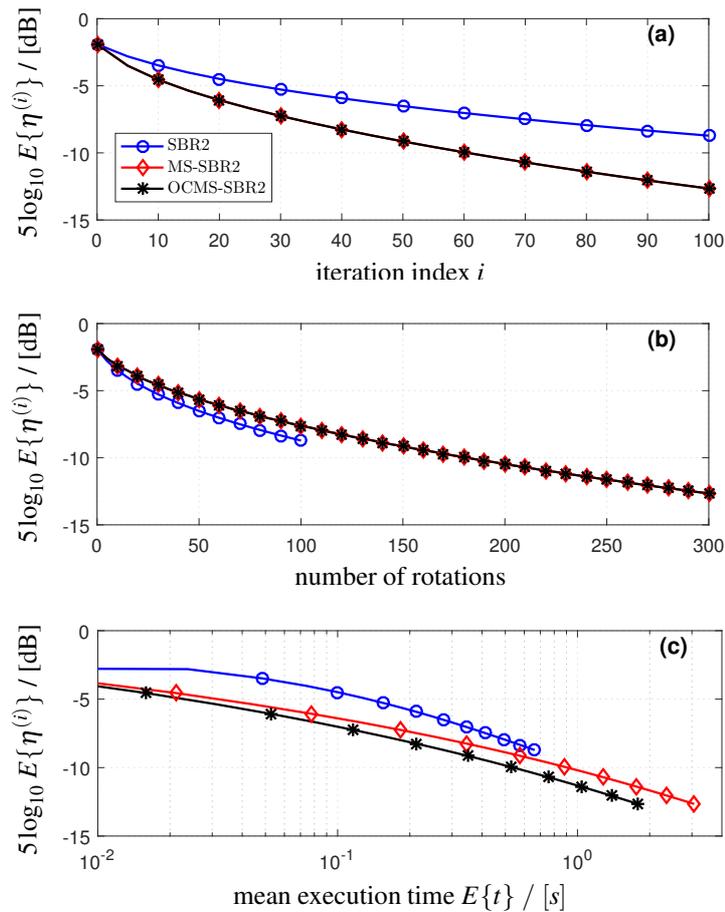


Figure 4.4 Convergence comparison of different versions of SBR2 for diagonalising 2000 randomly generated para-Hermitian matrices, showing (a) the ensemble average of normalized off-diagonal energy $\eta^{(i)}$ versus iterations, (b) number of Jacobi rotations, and (c) mean execution time $E\{t\}$ over 100 iterations.

Figures 4.5(a) and 4.5(b), respectively. Due to the multiple-shift operation, both versions of MS-SBR2 generate the transformed para-Hermitian and paraunitary matrices with much larger order than does SBR2. By using the direction-fixed shift method, the OCMS-SBR2 algorithm can remove all the zero-filled outer coefficient matrices generated at each iteration, which effectively controls the unnecessary growth in polynomial order. In this example, the polynomial order obtained from OCMS-SBR2 has been reduced by half compared to MS-SBR2.

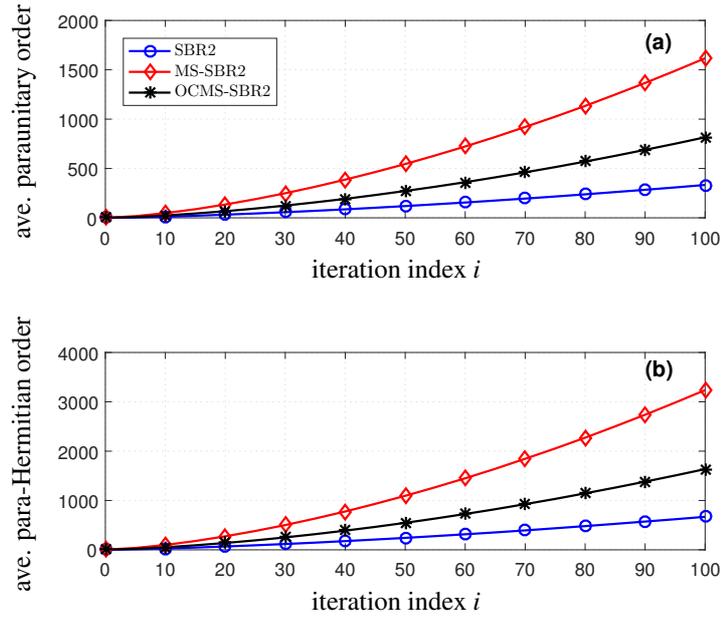


Figure 4.5 Comparison of polynomial orders among different versions of SBR2 without using any truncation methods, showing (a) the average order of $\underline{\mathbf{H}}^{(i)}(z)$ versus iterations, and (b) the average order of $\underline{\mathbf{R}}^{(i)}(z)$ versus iterations.

Case 2: With Truncation Methods

To see the order reductions after applying the truncation methods and the resulting reconstruction errors $\xi_{\text{PU}}^{(i)}$ and $\xi_{\text{PH}}^{(i)}$ at the i -th iteration, we make the algorithms retain the untrimmed matrices $\underline{\mathbf{R}}^{(i)}(z)$ and $\underline{\mathbf{H}}^{(i)}(z)$ through all iterations whilst generating the truncated copy $\underline{\mathbf{R}}_{\text{tr}}^{(i)}(z)$ with a chosen truncation parameter $\mu_{\text{PH}} = 10^{-4}$ and $\underline{\mathbf{H}}_{\text{tr}}^{(i)}(z)$ with $\mu_{\text{PU}} = 10^{-4}$. At every iteration, the average order of $\underline{\mathbf{H}}_{\text{tr}}^{(i)}(z)$ is recorded and depicted in Figure 4.6(a), together with the average reconstruction error $E\{\xi_{\text{PU}}^{(i)}\}$ shown in Figure 4.6(b). Initially, the error curves start very low but they quickly increase as the truncation methods begin to remove a portion of the energy. Although the same paraunitary error metrics are found between the two versions of MS-SBR2, the OCMS-SBR2 algorithm presents slightly better results than the MS-SBR2 algorithm in terms of the order reduction.

Similarly, the order of the truncated para-Hermitian matrix $\underline{\mathbf{R}}_{\text{tr}}^{(i)}(z)$ and the average para-Hermitian matrix reconstruction errors $E\{\xi_{\text{PH}}^{(i)}\}$ are presented in Figures 4.7(a) and (b), respectively. When the truncation method is applied to the different versions

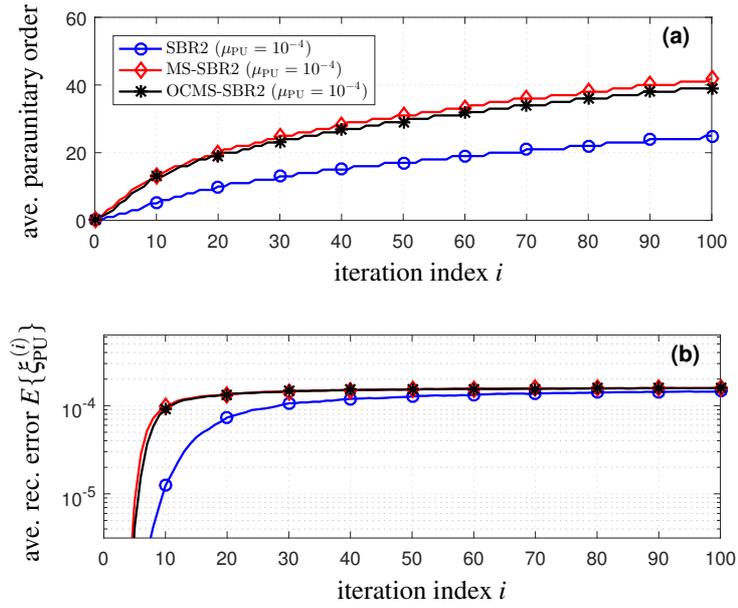


Figure 4.6 Results obtained from different SBR2 algorithms for the lag based truncation method with $\mu_{\text{PU}} = 10^{-4}$, showing (a) the average order after truncation of $\underline{\mathbf{H}}^{(i)}(z)$ versus iterations, and (b) the average reconstruction error $E\{\xi_{\text{pU}}^{(i)}\}$ versus iterations.

of SBR2 with the same truncation value, the OCMS-SBR2 algorithm shows a marginal advantage over the conventional MS-SBR2 algorithm in terms of the order reduction. By comparing it with the untrimmed case, as shown in Figure 4.5(b), this suggests that most of the outer coefficient matrices of $\underline{\mathbf{R}}^{(i)}(z)$ obtained from MS-SBR2 are essentially filled up with zeros. As the zero-filled outer matrices have already been removed at each iteration in OCMS-SBR2, the truncation process can only remove the non-zero coefficients. For this reason, the advantage of OCMS-SBR2 becomes less obvious. Furthermore, the reconstructed para-Hermitian matrix $\hat{\underline{\mathbf{R}}}^{(i)}(z) = \tilde{\underline{\mathbf{H}}}_{\text{tr}}^{(i)}(z)\underline{\mathbf{R}}_{\text{tr}}^{(i)}(z)\underline{\mathbf{H}}_{\text{tr}}^{(i)}(z)$ from OCMS-SBR2, appears to be a bit more accurate than that from MS-SBR2 according to the error results illustrated in Figure 4.7(b). Initially, the reconstruction error $\xi_{\text{pH}}^{(i)}$ of SBR2 is larger than those of both versions of MS-SBR2, however, as the iterations continue, this error metric of SBR2 is gradually decreasing and overtaken by both MS-SBR2 and OCMS-SBR2.

Overall, with a relatively small proportion of energy being removed, both orders of $\underline{\mathbf{H}}^{(i)}(z)$ and $\underline{\mathbf{R}}^{(i)}(z)$ have been drastically reduced for all different SBR2 algorithms

4.5 Numerical Examples

compared to the case in which no truncation method is used. In particular, the OCMS-SBR2 algorithm shows a slight performance advantage over MS-SBR2 when the truncation methods [86, 87] are used. Note that for the results presented in this section, the truncation methods are not implemented as part of an iterative routine in the SBR2 algorithms, and therefore the computational loads have not been affected. To investigate how the truncation methods can be used to reduce the computational load of the proposed MS-SBR2 algorithm, a more realistic application in strong decorrelation will be discussed in Section 4.7.

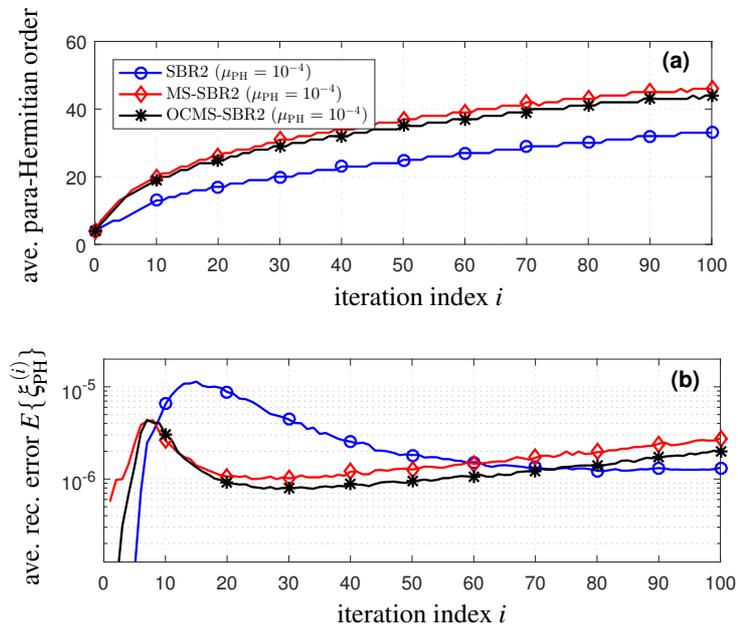


Figure 4.7 Results obtained from different SBR2 algorithms for the energy based truncation method with $\mu_{PH} = 10^{-4}$, showing (a) the average order after truncation of $\mathbf{R}^{(i)}(z)$ versus iterations, and (b) the average reconstruction error $E\{\xi_{PH}^{(i)}\}$ versus iterations.

In conclusion, not only can the OCMS-SBR2 algorithm produce the para-Hermitian and paraunitary matrices with lower polynomial order than using the conventional shift method in MS-SBR2, but also it is slightly advantageous in terms of the algorithm's accuracy, as suggested by the reconstruction error metrics of the studied examples. Furthermore, as shown in Figure 4.4(c), the OCMS-SBR2 algorithm is computationally faster than both SBR2 and MS-SBR2 due to zero-filled outer coefficient matrices

being removed at each iteration. Note that the diagonalisation performance is not affected when using the two different time-shift methods in MS-SBR2. Therefore, with the demonstrated advantages of the OCMS-SBR2 algorithm, all experimental results which involve the MS-SBR2 algorithm in the remainder of this thesis will be generated based on the order controlled MS-SBR2 algorithm, i.e., OCMS-SBR2.

4.6 Impact of Source Model Matrix Conditioning

In the previous section, we examined the proposed MS-SBR2 algorithm based on various performance metrics by means of randomly generated para-Hermitian matrix examples. Each of those para-Hermitian matrices $\underline{\mathbf{R}}(z) \in \underline{\mathbb{C}}^{M \times M}$ is derived from a randomly generated polynomial matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{M \times N}$ as $\underline{\mathbf{R}}(z) = \underline{\mathbf{A}}(z)\tilde{\underline{\mathbf{A}}}(z)$, where all the entries of $\underline{\mathbf{A}}(z)$ satisfy the same Gaussian distribution. In practical experiments, the relative performance of the PEVD algorithms depends quite significantly on the type of para-Hermitian matrix that is to be factorised. In [35], a source model was proposed to generate the para-Hermitian matrix for testing the PEVD algorithms, where the ground truth PEVD with finite order paraunitary factors and equality in (3.1) is guaranteed. This source model idea was later generalised by Corr et al. in [92] to further investigate how to control the conditioning of para-Hermitian matrices to be diagonalised by the PEVD algorithms. This includes defining the dynamic range of the underlying sources, which is related the eigenvalue spread or condition number of a para-Hermitian matrix generalised from the area of Hermitian matrices. In addition to that, different relations between the sources' power spectral densities (PSDs) are defined, including

- spectrally majorised PSDs, as demonstrated in (3.25);
- not spectrally majorised PSDs, i.e., with overlapping PSDs.

In this section, the source matrix conditioning method from [92] is adopted to investigate how different source models can affect the proposed MS-SBR2 algorithm.

We begin by briefly introducing the source model, followed by the relevant simulation results.

4.6.1 Source Model

The source model is illustrated by the block diagram shown in Figure 4.8. A total of N mutually uncorrelated source signals $s_n[t], n = 1, \dots, N$ are generated by emitting uncorrelated, zero mean unit variance complex Gaussian signals $u_n[t]$ through N innovation filters $\underline{f}_n(z)$ of order L . The individual PSDs of the source signals $s_n[t]$ are therefore given by $\underline{r}_{s,n}(e^{j\Omega}) = \underline{f}_n(e^{j\Omega})\tilde{\underline{f}}_n(e^{j\Omega})$ where $\underline{f}_n(e^{j\Omega})$ represents the Fourier transform of the n -th innovation filter [35, 93]. By carefully controlling the maximum radius of zeros and the filter gain, the dynamic PSD range of the source signals can be obtained as well as different relations of the PSDs of the sources [92]. The convolutive mixed sources $x_n[t], n = 1, \dots, N$ are then formed by transmitting $s_n[t]$ through a random paraunitary matrix $\underline{\mathbf{A}}(z) \in \underline{\mathbb{C}}^{N \times N}$ of order K , which can be generated as [35, 69]

$$\underline{\mathbf{A}}(z) = \mathbf{A}_0 \prod_{k=1}^K \underline{\mathbf{A}}_k(z) = \mathbf{A}_0 \prod_{k=1}^K (\mathbf{I} - \mathbf{a}_k \mathbf{a}_k^H + \mathbf{a}_k \mathbf{a}_k^H z^{-1}), \quad (4.23)$$

where $\mathbf{A}_0 \in \mathbb{C}^{N \times N}$ denotes a unitary matrix, and $\mathbf{a}_k \in \mathbb{C}^N, k = 1, \dots, K$ are random vectors with unit norm, which are used to construct a random first order paraunitary matrix $\underline{\mathbf{A}}_k(z)$.

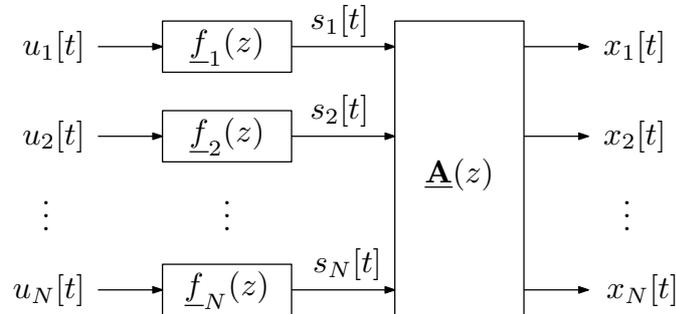


Figure 4.8 Block diagram of a source model consisting of N independent zero mean unit variance complex Gaussian sources $u_n[t], n = 1, \dots, N$, innovation filters with transfer functions $\underline{f}_n(z)$, and a paraunitary convolutive mixing system $\underline{\mathbf{A}}(z)$.

The cross spectral density (CSD) matrix of the output signals $x_n[t]$, $n = 1, \dots, N$ is therefore expressed as

$$\begin{aligned} \underline{\mathbf{R}}(z) &= \sum_{\tau} \underline{\mathbf{R}}[\tau] z^{-\tau} = \sum_{\tau} E\{\mathbf{x}[t] \mathbf{x}^H[t - \tau]\} z^{-\tau} \\ &= \underline{\mathbf{A}}(z) \underline{\mathbf{R}}_s(z) \tilde{\underline{\mathbf{A}}}(z) = \underline{\mathbf{A}}(z) \underline{\mathbf{F}}(z) \tilde{\underline{\mathbf{F}}}(z) \tilde{\underline{\mathbf{A}}}(z), \end{aligned} \quad (4.24)$$

where $\underline{\mathbf{R}}_s(z)$ denotes the PSD matrix of $s_n[t]$, which is determined by the diagonal matrix $\underline{\mathbf{F}}(z) \tilde{\underline{\mathbf{F}}}(z) = \text{diag}\{f_{\underline{1}}(z) \tilde{f}_{\underline{1}}(z), \dots, f_{\underline{N}}(z) \tilde{f}_{\underline{N}}(z)\}$.

Clearly, if $\underline{\mathbf{R}}_s(z)$ or $\underline{\mathbf{F}}(z)$ is spectrally majorised, the PEVD $\underline{\mathbf{R}}(z) = \tilde{\underline{\mathbf{H}}}(z) \underline{\mathbf{D}}(z) \underline{\mathbf{H}}(z)$ in (3.1) exists with equality, and it satisfies $\tilde{\underline{\mathbf{H}}}(z) = \underline{\mathbf{A}}(z)$ and $\underline{\mathbf{D}}(z) = \underline{\mathbf{F}}(z) \tilde{\underline{\mathbf{F}}}(z)$. However, if $\underline{\mathbf{R}}_s(z)$ is not spectrally majorised, the PEVD of $\underline{\mathbf{R}}(z)$ often requires the order of the factors of $\underline{\mathbf{H}}(z)$ and $\underline{\mathbf{D}}(z)$ to be higher than in the spectrally majorised case in order to achieve both strong decorrelation (or diagonalisation) and spectral majorisation. We will now demonstrate how different source model matrices can affect the performance of the MS-SBR2 algorithm by means of numerical examples.

4.6.2 Simulation Scenario

To generate the CSD matrices, the average dynamic PSD range of the source model is chosen to be either 10 dB or 30 dB, together with different relations of the sources' PSDs. This leads to four specific cases: (i) spectrally majorised sources with 10 dB dynamic range, (ii) not spectrally majorised sources with 10 dB dynamic range, (iii) spectrally majorised sources with 30 dB dynamic range, and (iv) not spectrally majorised sources with 30 dB dynamic range. In each case, results are averaged over an ensemble of 200 random realisations. For each realisation, the source model is created based on $N = 5$ independent sources and the same number of sensors. The order of the innovation filters and the paraunitary mixing matrices are set to be $L = 20$ and $K = 20$, respectively. Thus, each randomly generated CSD matrix $\underline{\mathbf{R}}(z)$ is 5×5 with order $2(L + K) = 80$.

4.6.3 Algorithm Convergence and Paraunitary Order

The chosen PEVD algorithms, including SBR2 and MS-SBR2, are applied to each CSD matrix for a total of 500 iterations. The lag based truncation method, as described in Section 3.6.3, is applied to the resulting paraunitary matrices obtained from both algorithms after each iteration, with $\mu_{\text{PU}} = 10^{-4}$.

Figures 4.9(a) and (b) show how the different SBR2 algorithms converge for the CSD matrix examples obtained from different source models with dynamic ranges of 10 dB and 30 dB, respectively. For both dynamic ranges, the spectrally unmajorised sources initially converge faster than the the majorised sources for all algorithms. However, as the iterations continue, the unmajorised sources are gradually overtaken by the majorised sources in terms of the convergence speed, and after 500 iterations, there is an obvious difference between the two source models, where the majorised sources present better performance in terms of reducing the off-diagonal energy. Furthermore, the results shown in Figure 4.9(b) with higher dynamic range are worse than those in Figure 4.9(a) with lower dynamic range. Undoubtedly, the MS-SBR2 algorithm outperforms SBR2 in terms of the diagonalisation measure, as multiple off-diagonal elements are annihilated at each iteration.

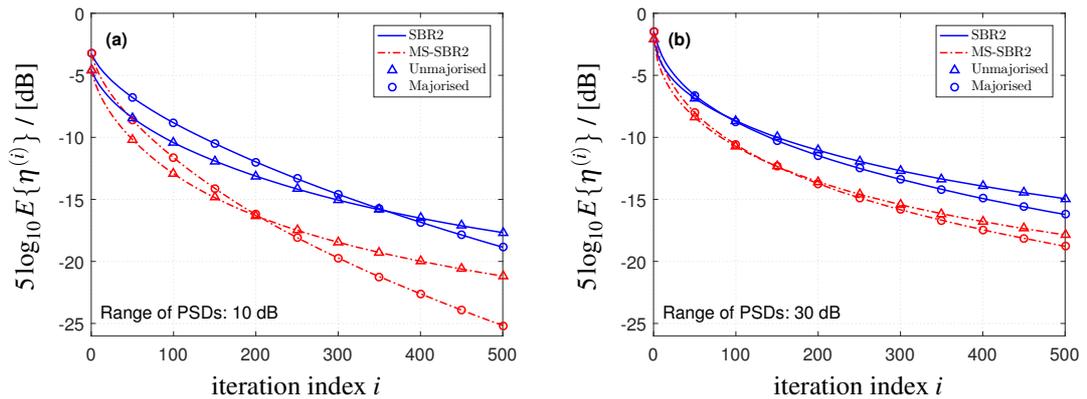


Figure 4.9 Remaining off-diagonal energy versus iterations for the SBR2 and MS-SBR2 algorithms over 200 random realisations, showing both majorisation types with (a) 10 dB dynamic range, and (b) 30 dB dynamic range.

Apart from the convergence comparison for different source models, another important performance metric is the order of the resulting paraunitary matrices obtained from different algorithms. Figure 4.10(a) presents the average remaining off-diagonal energy versus the paraunitary matrix order for both algorithms, obtained from the majorised and unmajorised source models with 10 dB dynamic range, demonstrating that the SBR2 and MS-SBR2 algorithms perform similarly in terms of the growth rate of the paraunitary order. However, when the dynamic range is increased to 30 dB, the paraunitary order from the MS-SBR2 algorithm increases faster than from SBR2 for the majorised sources, as shown in Figure 4.10(b). This reflects that the multiple shifts of the MS-SBR2 algorithm cause the paraunitary order to grow faster in the case of higher dynamic range. Nonetheless, for both dynamic ranges, Figure 4.10 shows that the unmajorised sources tend to have larger paraunitary order than the majorised sources for both the SBR2 and MS-SBR2 algorithms.

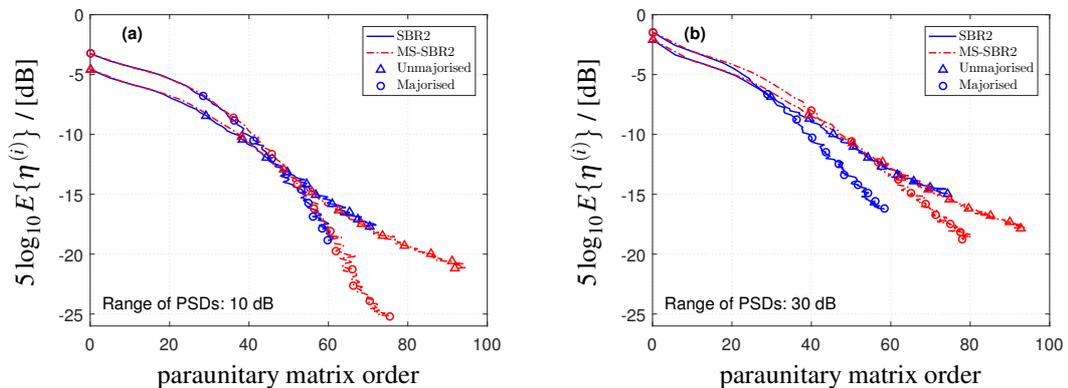


Figure 4.10 Paraunitary matrix order for the SBR2 and MS-SBR2 algorithms over an ensemble of 200 random realisations, showing both majorisation types with (a) 10 dB dynamic range, and (b) 30 dB dynamic range.

4.6.4 Spectral Ordering

As the simulations from the above section are based on an ensemble average of 200 realisations, the resulting PSDs obtained from applying the PEVD algorithms to different source model matrices are not presented. In this section, a single source model is

chosen from each of the four specific cases as the scenario to investigate the PSDs, and the MS-SBR2 algorithm is applied to each source model matrix for 150 iterations. In the case of the 10 dB dynamic range, Figures 4.11(a) and (b) illustrate the on-diagonal PSDs of $\underline{\mathbf{D}}(z)$ for the majorised and unmajorised source models, respectively. Note that the ground truth (or ideal) PSDs are shown in light grey shading. For the majorised source model, the MS-SBR2 algorithm has converged to a spectrally majorised solution, which closely matches the underlying ground truth PSDs. However, for the unmajorised source, the results are approximately spectrally majorised by re-ordering the spectral components of $\underline{\mathbf{F}}(z)$ using a paraunitary matrix, which is likely to increase the order of the polynomial factors in PEVD.

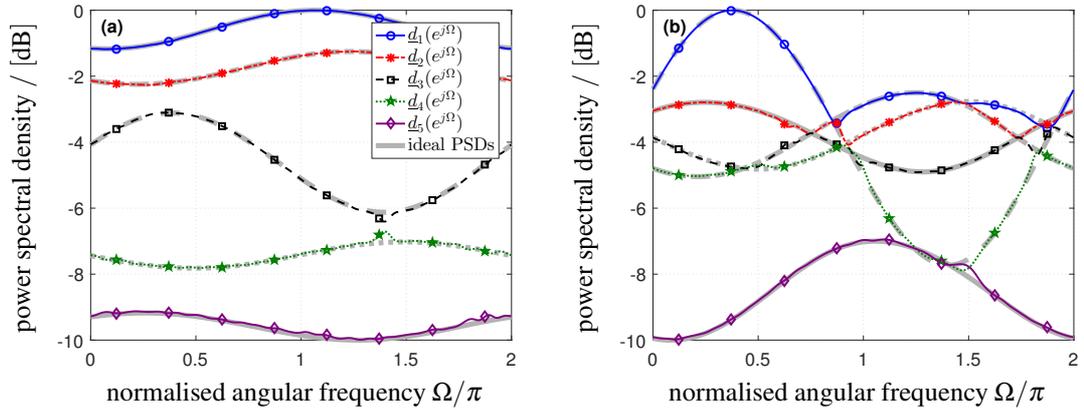


Figure 4.11 PSDs of the on-diagonal polynomials of $\underline{\mathbf{D}}(z)$ for (a) a strictly majorised source model, and (b) an unmajorised source model, both with 10 dB dynamic range obtained from MS-SBR2 after 150 iterations, superimposed on light grey ideal PSDs.

Figures 4.12(a) and (b) respectively show the PSDs for the majorised and unmajorised sources, both with 30 dB dynamic range. With the larger dynamic range, the MS-SBR2 algorithm is not able to produce spectrally majorised results within 150 iterations. As can be seen in Figures 4.12(a) and (b), some cross-talk occurs in lower power channels for both majorised and unmajorised source models. Compared to the majorised source model, the MS-SBR2 algorithm appears to produce better spectral ordering results for the unmajorised source model.

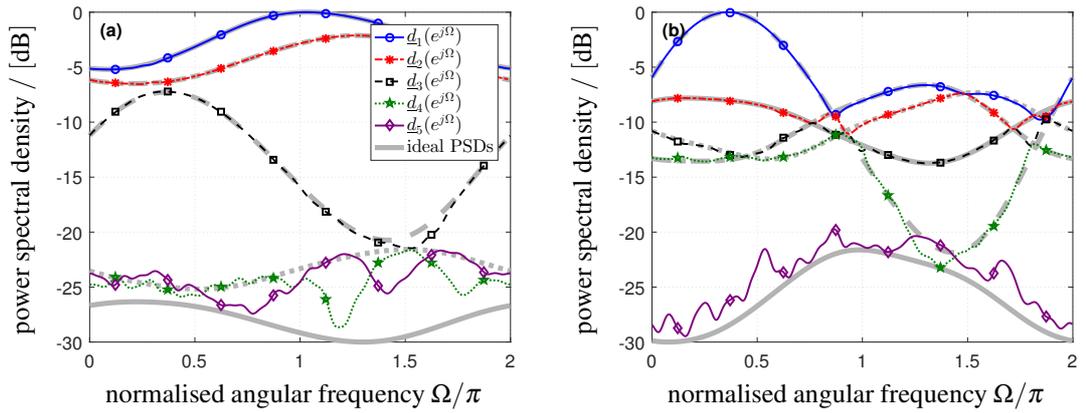


Figure 4.12 PSDs of the on-diagonal polynomials of $\underline{\mathbf{D}}(z)$ for (a) a strictly majorised source model, and (b) an unmajorised source model, both with 30 dB dynamic range obtained from MS-SBR2 after 150 iterations, superimposed on light grey ideal PSDs.

4.7 Strong Decorrelation of Convolutively Mixed Signals

This section provides a numerical example to demonstrate how the proposed MS-SBR2 algorithm can be used as a technique for enforcing strong decorrelation upon a set of convolutively mixed signals and as a tool for identifying the signal and noise subspaces. This example also allows the comparison of the performance of the MS-SBR2 algorithm to that of its predecessor, SBR2 [7].

4.7.1 Simulation Scenario

To set up the experiment, a (4×5) convolutive MIMO system, with the channel mixing matrix $\underline{\mathbf{C}}(z) \in \mathbb{R}^{5 \times 4}$, is designed to emulate the propagation of four signals onto five sensors. Each of the polynomial entries $c_{jk}(z)$, for $j = 1, \dots, 5$ and $k = 1, \dots, 4$ was chosen to be a fifth order finite impulse response (FIR) filter with the coefficients $c_{jk}[\tau]$, for $\tau = 0, \dots, 5$ drawn randomly from a uniform distribution within the range $[-1, 1]$.

Four independent binary phase shift keying (BPSK) source signals $\mathbf{s}[t] \in \mathbb{R}^{4 \times 1}$ are then generated and emitted through the convolutive mixing channel according to (2.21). Each source signal has a length of 1000 independent and identically distributed (i.i.d.)

sequences for which each sample was assigned the value ± 1 with probability $1/2$. With the source signals chosen to have unit variance, the CSD matrix of the source signals satisfies $\underline{\mathbf{R}}_{ss}(z) = \mathbf{I}_N$. As a result of the mixing process in (2.23), the sensor outputs $\mathbf{x}[t] \in \mathbb{R}^{5 \times 1}$ will generally be correlated with one another over a range of time delays. Thus, the expected CSD matrix of the signals $\mathbf{x}[t]$ is represented by

$$\begin{aligned} \underline{\mathbf{R}}_{xx}(z) &= \underline{\mathbf{C}}(z)\underline{\mathbf{R}}_{ss}(z)\tilde{\underline{\mathbf{C}}}(z) + \sigma^2\mathbf{I}_M \\ &= \underline{\mathbf{C}}(z)\tilde{\underline{\mathbf{C}}}(z) + \sigma^2\mathbf{I}_M, \end{aligned} \quad (4.25)$$

where σ^2 denotes the variance of the noise and for this experiment σ is set equal to 1.5. Thus, the signal-to-noise ratio (SNR) at the receiver can be estimated by

$$\text{SNR} = 10 \log_{10} \left(\frac{\text{trace} \{ (\underline{\mathbf{C}}(z)\tilde{\underline{\mathbf{C}}}(z)) |_{\tau=0} \}}{M \sigma^2} \right), \quad (4.26)$$

where the notation $(\cdot)|_{\tau=0}$ denotes the zero-lag coefficient matrix of the polynomial matrix inside the bracket, and the number of sensors, M , is equal to five. Hence, the value of the SNR is calculated as 5.08 dB for this specific realisation.

4.7.2 Results and Analysis

The estimated space-time covariance matrix $\hat{\mathbf{R}}_{xx}[\tau]$ can be calculated in accordance with (3.20) where the correlation window parameter W was set to 10. As the mixing matrix $\underline{\mathbf{C}}(z)$ has order of 5, $\hat{\mathbf{R}}_{xx}[\tau]$ will be approximately zero for all lags $|\tau| > 5$ and any deviation in these lags will be due to sample estimation errors. For this reason, the choice of $W = 10$ is more than sufficient. A graphical representation of the estimated CSD matrix $\hat{\underline{\mathbf{R}}}_{xx}(z)$ is plotted in Figure 4.13.

Both the SBR2 and MS-SBR2 algorithms are then applied to diagonalise this matrix with the same stopping condition

$$g \leq \varepsilon = \delta \sqrt{\frac{1}{M} \sum_{m=1}^M |r_{mm}[0]|^2}, \quad (4.27)$$

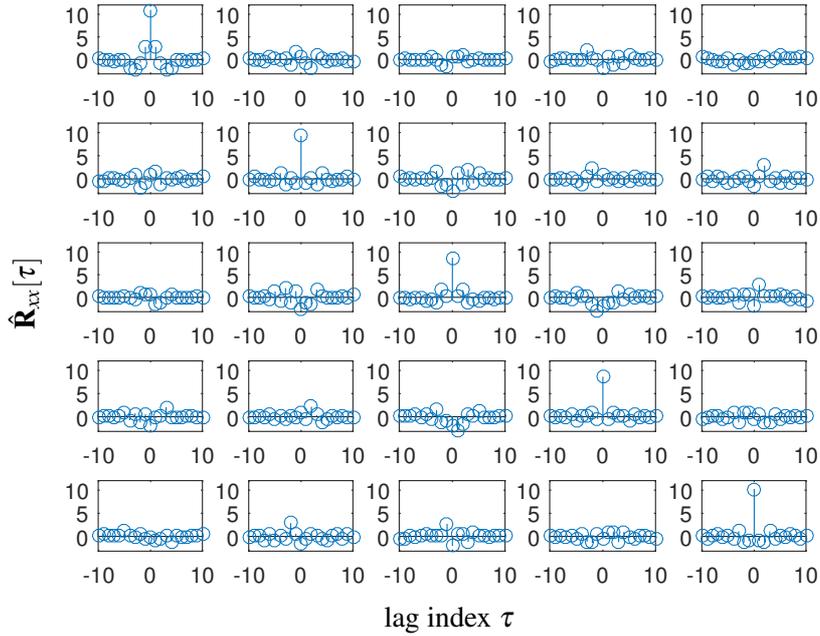


Figure 4.13 The stem plot of the estimated space-time covariance matrix $\hat{\mathbf{R}}_{xx}[\tau] \circ \bullet \hat{\mathbf{R}}_{xx}(z)$ of the convolutively mixed signals $\mathbf{x}[t]$ with the chosen SNR of 5.08 dB.

where $g = |r_{jk}^{(i)}[\tau]|$ denotes the magnitude of the maximum off-diagonal element found at the i -th iteration and $r_{mm}[0]$ for $m = 1, \dots, M$ represent the diagonal elements of the zero-lag coefficient matrix $\hat{\mathbf{R}}_{xx}[0]$. In this experiment, δ is set to 10^{-2} , which results in $\varepsilon = 0.0962$.

Case 1: No Truncation

We have considered two different situations of the simulation depending on whether the truncation methods are used or not. For the simple case of no truncation, the SBR2 algorithm took 288 iterations to converge to a point where $g = |r_{jk}^{(288)}[\tau]| = 0.0955$, while the MS-SBR2 algorithm only took 147 iterations to reach a similar convergence level. Figure 4.14 shows the behaviour of $|r_{jk}^{(i)}[\tau]|$ versus iterations for these two different PEVD algorithms. The polynomial matrices $\mathbf{D}(z)$ and $\mathbf{H}(z)$ obtained by applying the diagonalisation to $\mathbf{R}_{xx}(z)$ using MS-SBR2 are plotted in Figure 4.15 and Figure 4.16, respectively. Upon inspection of the plots, the polynomial orders of these matrices are

unnecessarily large, with most of the coefficients positioned in the outer lags amounting to a small proportion of the squared Frobenius norm of the diagonal matrix $\mathbf{D}(z)$.

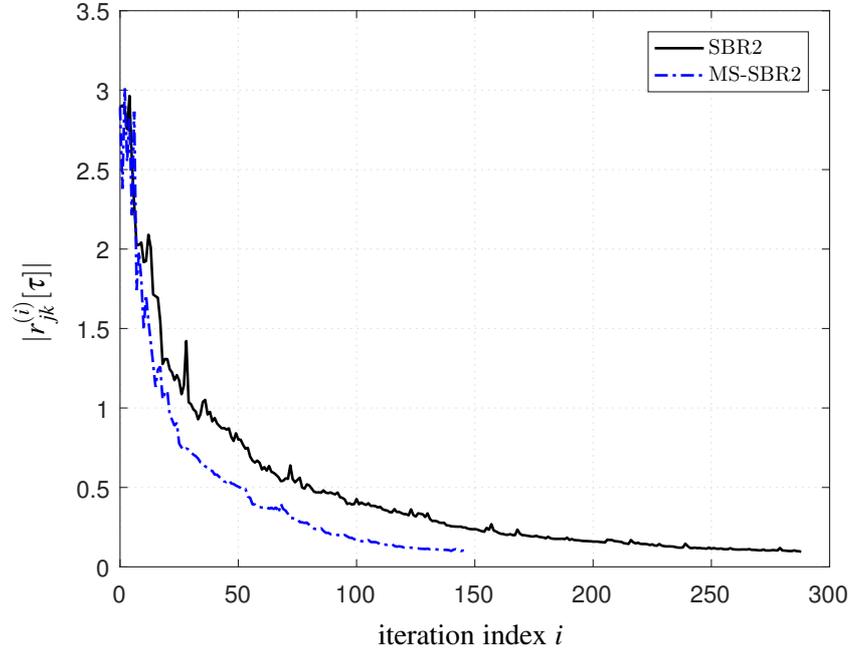


Figure 4.14 The magnitude of the maximum off-diagonal element $|r_{jk}^{(i)}[\tau]|$ found at the i -th iteration for the SBR2 and MS-SBR2 algorithms.

The performance metrics are recorded in Table 4.1, which shows the complete comparisons between the proposed MS-SBR2 algorithm and the original SBR2 algorithm. Using the same stopping condition mentioned in (4.27), MS-SBR2 can generate the diagonal matrix $\mathbf{D}(z)$ with the remaining off-diagonal energy $\|\text{off}\{\mathbf{D}(z)\}\|_{\text{F}}^2 = 1.8433$, which is less than that of SBR2; it also yields the matrices $\mathbf{D}(z)$ and $\mathbf{H}(z)$ with lower order. As to the error metrics defined in (4.22) and (4.20), both the paraunitary reconstruction error ξ_{PU} and the para-Hermitian reconstruction error ξ_{PH} are very small (around 10^{-30}). In fact, these deviations are due to round-off errors caused by the machine precision. In theory, they should be equal to 0 as no truncation method is used. Although the cost per iteration in MS-SBR2 is higher than that of SBR2, the total execution time taken for MS-SBR2 to calculate the decomposition is less than that of SBR2.

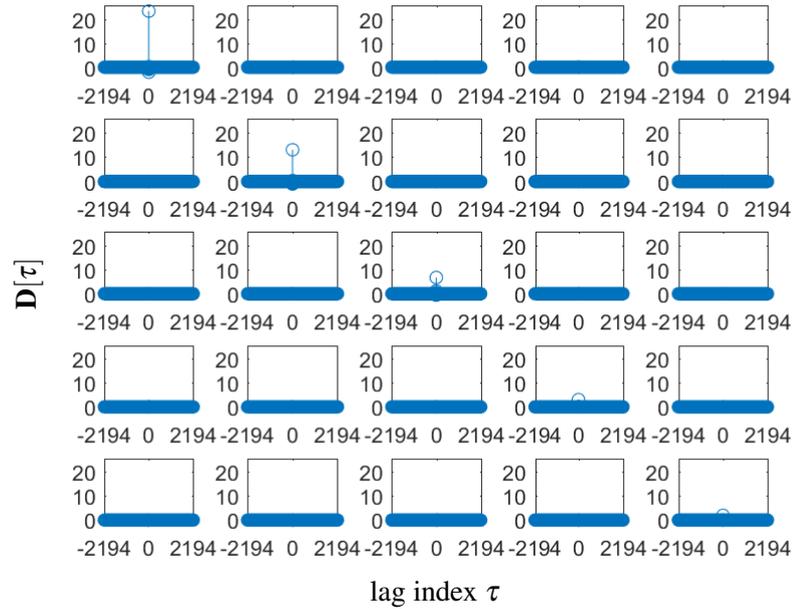


Figure 4.15 The stem plot of the strongly decorrelated CSD matrix $\underline{\mathbf{D}}(z)$ after applying the MS-SBR2 algorithm to the CSD matrix $\underline{\mathbf{R}}(z)$ shown in Figure 4.13, with no polynomial order truncation involved.

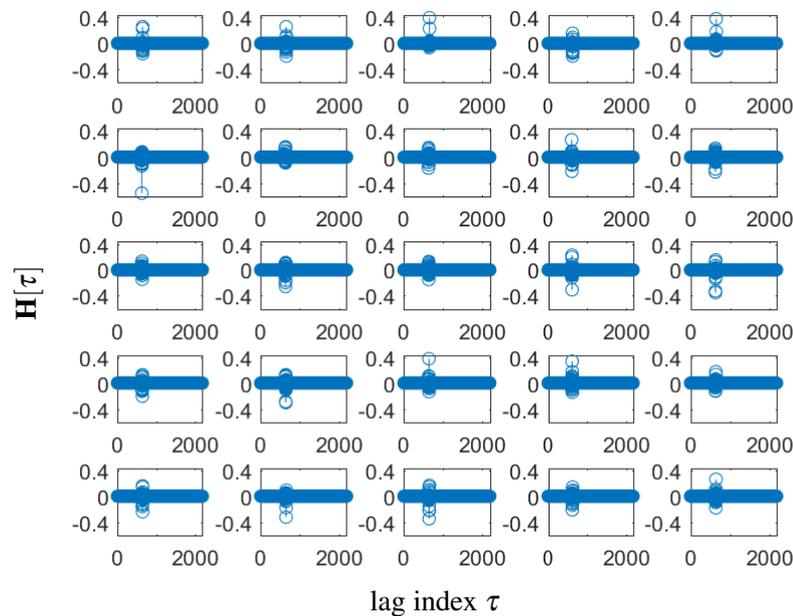


Figure 4.16 The stem plot of the generated paraunitary polynomial matrix $\underline{\mathbf{H}}(z)$ after applying the MS-SBR2 algorithm to the CSD matrix $\underline{\mathbf{R}}(z)$ shown in Figure 4.13, with no polynomial order truncation involved.

Table 4.1 Performance comparisons between the SBR2 and MS-SBR2 algorithms when applied to the same CSD matrix example in Figure 4.13, without implementing any order truncation process.

Measures	SBR2	MS-SBR2
converged value g	0.0955	0.0960
number of iterations	288	147
$\ \underline{\mathbf{D}}(z)\ _{\text{F}}^2$	792.5783	792.5783
$\ \text{off}\{\underline{\mathbf{D}}(z)\}\ _{\text{F}}^2$	2.0865	1.8433
order of $\underline{\mathbf{H}}(z)$	3086	2184
order of $\underline{\mathbf{D}}(z)$	6192	4388
ξ_{PU}	1.4959×10^{-30}	1.7045×10^{-30}
ξ_{PH}	1.9674×10^{-30}	2.5679×10^{-30}
computational time (sec.) ¹	0.7729	0.5516

¹ Computations undertaken on a PC with Intel(R) Core(TM) i7-3770T CPU @ 2.50GHz and 16 GB RAM.

Figure 4.17(a) shows the PSD of the mixed signals $\underline{\mathbf{x}}(z)$. Following the strong decorrelation using the paraunitary matrix $\underline{\mathbf{H}}(z)$ obtained from the MS-SBR2 algorithm, the PSD of the decorrelated signals $\underline{\mathbf{y}}(z)$ is plotted in Figure 4.17(b). Clearly, the decorrelated signals are spectrally majorised as the PSDs satisfy $\underline{d}_1(e^{j\Omega}) \geq \underline{d}_2(e^{j\Omega}) \geq \dots \geq \underline{d}_5(e^{j\Omega})$ at every frequency. More importantly, the spectrally majorised signals tend to have most of the related signal energy focused in the smallest number of signal channels [7]. Therefore the noise subspace can be identified as the the signal with the lowest spectra, i.e., $y_5[t]$.

It has been proven that the total input signal power, for all frequencies, is invariant under a paraunitary transformation [69]. This can be seen in Figure 4.18, where the total PSD of all signals is plotted before and after applying the paraunitary transformation matrix $\underline{\mathbf{H}}(z)$ obtained from the MS-SBR2 algorithm. Note that a paraunitary transformation can redistribute the power across channels, but it cannot amplify or attenuate the total power.

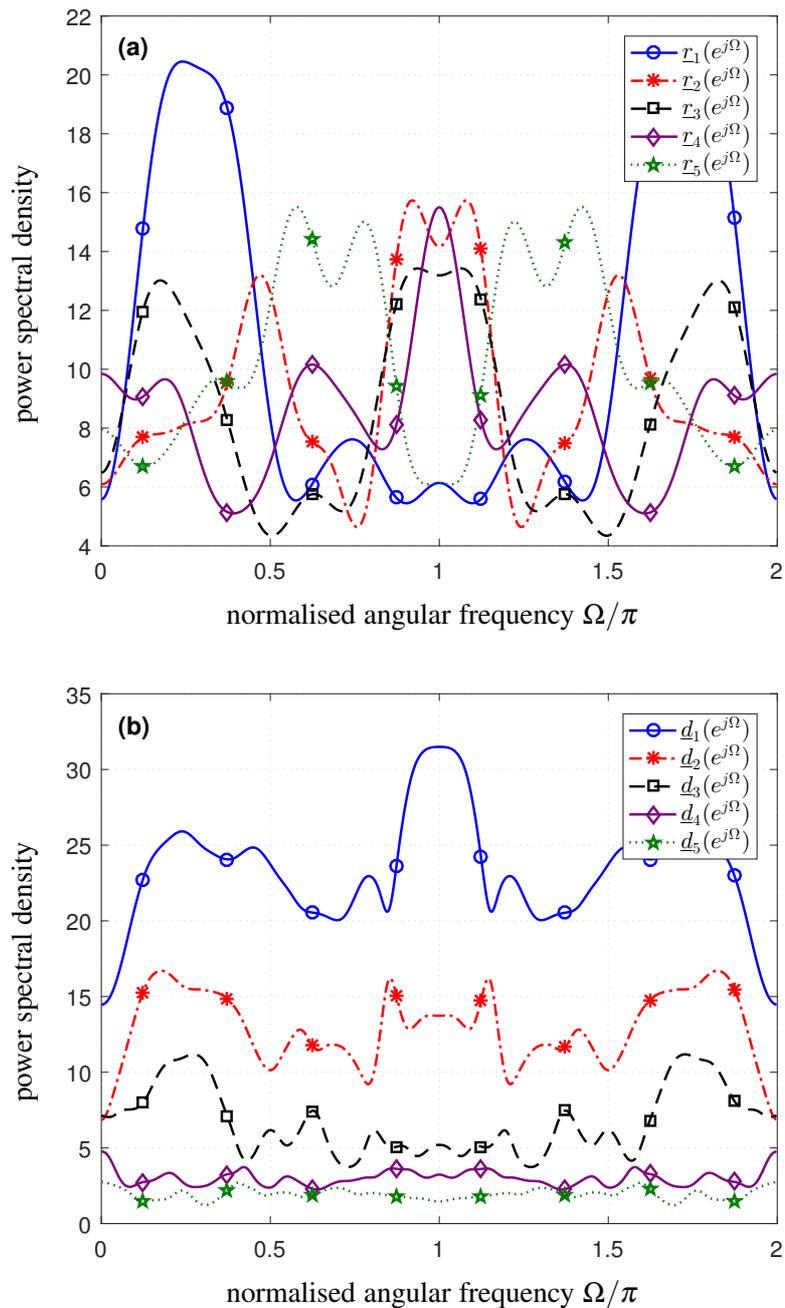


Figure 4.17 Power spectral densities of (a) the convolutively mixed signals $\mathbf{x}[t]$, and (b) the strongly decorrelated signals $\mathbf{y}[t]$ generated using the paraunitary matrix $\mathbf{H}(z)$ obtained from the MS-SBR2 algorithm.

Case 2: With Truncation Methods

The above example demonstrates the ability of the MS-SBR2 algorithm to calculate the PEVD of the estimated CSD matrix $\hat{\mathbf{R}}_{xx}(z)$ and consequently strongly decorrelate the received signals $\mathbf{x}[t]$. However, this example also illustrates the unnecessarily large

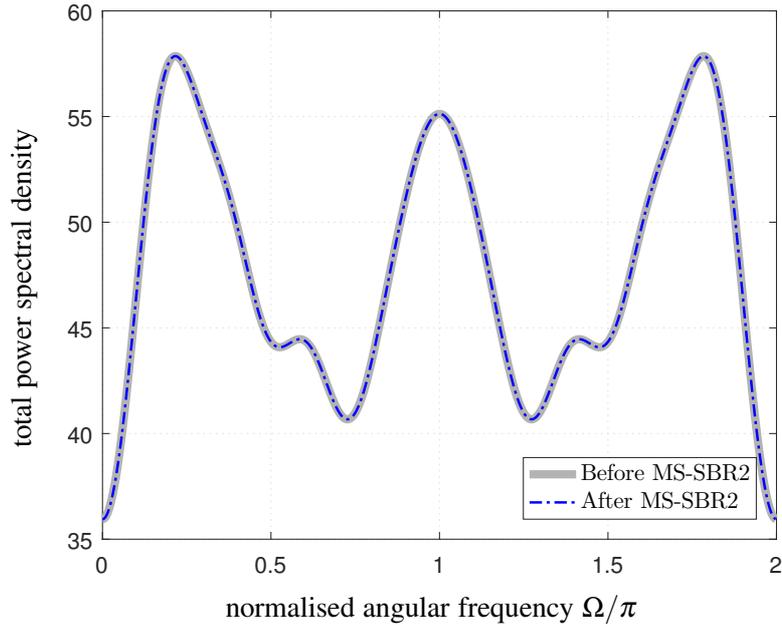


Figure 4.18 Total power spectral density of the expected signals, before and after applying the paraunitary transformation matrix $\underline{\mathbf{H}}(z)$ obtained from the MS-SBR2 algorithm.

orders of the diagonalised matrix $\underline{\mathbf{D}}(z)$ and paraunitary matrix $\underline{\mathbf{H}}(z)$ obtained from the MS-SBR2 algorithm. The order growth in the para-Hermitian matrix is problematic as it will lead to a significant increase in the computational complexity of the algorithm. Also, it would be very costly to apply the final paraunitary matrix of large order to subspace-based applications, as mentioned in Section 3.6.1.

To demonstrate how the truncation methods can be used to address these limitations, the proposed MS-SBR2 algorithm was again applied to this example. However, this time, the truncation methods are implemented at the end of each iteration of the algorithm. Specifically, the energy based truncation method [86] is employed to generate the truncated para-Hermitian matrix $\underline{\mathbf{R}}_{\text{tr}}^{(i)}(z)$ at the i -th iteration, whilst the lag based truncation method [87] is used to obtain the truncated paraunitary matrix $\underline{\mathbf{H}}_{\text{tr}}^{(i)}(z)$. This experiment was carried out for four different choices of the para-Hermitian truncation parameter μ_{PH} and the paraunitary truncation parameter μ_{PU} : (i) $\mu_{\text{PH}} = \mu_{\text{PU}} = 0$, (ii) $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-8}$, (iii) $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-6}$, and (iv) $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-4}$. Note that μ_{PH} can also be set differently from μ_{PU} . In each case, the same stopping condition

defined in (4.27) was adopted. The results observed for the four cases are presented in Table 4.2.

By using the truncation methods, the orders of the diagonalised matrix $\underline{\mathbf{D}}_{\text{tr}}(z)$ and the corresponding paraunitary matrix $\underline{\mathbf{H}}_{\text{tr}}(z)$, obtained from the MS-SBR2 algorithm, are drastically reduced. Even removing the outer coefficient matrices with norm equal to zero (to computational precision) by setting $\mu_{\text{PH}} = 0$ significantly reduces the order of the diagonalised matrix from 4388 (see Table 4.1) to 2438 without compromising the accuracy of the decomposition, i.e. the para-Hermitian reconstruction error ξ_{PH} remains unaffected. Similarly, by setting $\mu_{\text{PU}} = 0$, the paraunitary matrix has been truncated without losing the paraunitarity, and its order has been reduced from 2184 to 1372. Furthermore, if both μ_{PH} and μ_{PU} are set larger than zero, then the polynomial orders can be further reduced. However, the transformation performed is no longer norm preserving and will therefore result in some errors. The error measures ξ_{PU} and ξ_{PH} will increase as the truncation parameters go up.

Another advantage of truncating the polynomial matrices is that the computational load and memory storage requirements of the algorithm are reduced, which reduces the time required for the algorithm to converge. This is clearly demonstrated by the measures of the computational time shown in Table 4.2. With suitable choices of μ_{PH} and μ_{PU} , the speed of the algorithm can be optimised whilst maintaining the error metrics ξ_{PU} and ξ_{PH} at a relatively low level. In practise, it is difficult to know in advance what values to choose for the truncation parameters μ_{PH} and μ_{PU} , and proper choices of these parameters will entirely depend on the requirements of the decomposition for the specified application. If computational time and memory are not considered as the most important factors when applying the MS-SBR2 algorithm, it is better to assign very small values to the truncation parameters, for example $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-8}$, as this not only brings down the polynomial order, but also minimises the error of the decomposition.

Table 4.2 Performance measures of the MS-SBR2 algorithm when applied to the same CSD matrix example in Figure 4.13, implementing the energy based para-Hermitian truncation and lag based paraunitary truncation methods for different values of μ_{PH} and μ_{PU} .

Measures	Truncation parameter $\mu_{\text{PH}} = \mu_{\text{PU}} =$			
	0	10^{-8}	10^{-6}	10^{-4}
converged value g	0.0960	0.0959	0.0925	0.0957
number of iterations	147	147	151	121
$\ \underline{\mathbf{D}}_{\text{tr}}(z)\ _{\text{F}}^2$	792.5783	792.5777	792.5102	787.8571
$\ \text{off}\{\underline{\mathbf{D}}_{\text{tr}}(z)\}\ _{\text{F}}^2$	1.8433	1.8440	1.7734	0.3288
order of $\underline{\mathbf{H}}_{\text{tr}}(z)$	1372	105	99	44
order of $\underline{\mathbf{D}}_{\text{tr}}(z)$	2438	132	134	30
ξ_{PU}	1.7045×10^{-30}	1.7636×10^{-6}	1.5986×10^{-4}	0.0071
ξ_{PH}	2.5679×10^{-30}	7.5229×10^{-10}	1.7834×10^{-6}	0.0031
computational time (sec.) ¹	0.6166	0.1422	0.1338	0.0851

¹ Computations undertaken on a PC with Intel(R) Core(TM) i7-3770T CPU @ 2.50GHz and 16 GB RAM.

Figure 4.19(a) and (b) respectively illustrate how the orders of the paraunitary matrix $\underline{\mathbf{H}}^{(i)}(z)$ and the transformed para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$ increase as iterations continue in the MS-SBR2 algorithm, for each of the five truncation cases recorded in Tables 4.1 and 4.2. It can be seen clearly from the figure that the polynomial orders are continuously increasing when no truncation method is used, which leads to a very large order for both matrices. Furthermore, by comparing the individual plots for each truncation value, it is clear that the order can be significantly reduced whilst still maintaining an accurate level of decomposition. For the case of $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-4}$, the MS-SBR2 algorithm outputs the resulting paraunitary matrix $\underline{\mathbf{H}}_{\text{tr}}(z)$ and the diagonal matrix $\underline{\mathbf{D}}_{\text{tr}}(z)$, and these matrices have the lowest orders among all the truncation cases. The stem plots of $\underline{\mathbf{D}}_{\text{tr}}(z)$ and $\underline{\mathbf{H}}_{\text{tr}}(z)$ are shown in Figures 4.20 and 4.21, respectively.

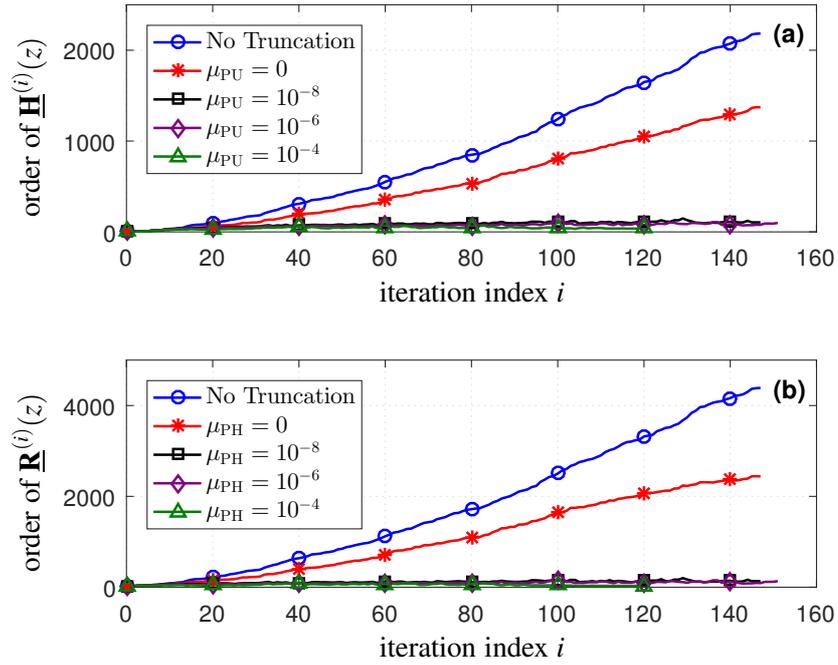


Figure 4.19 The polynomial orders of: **(a)** the paraunitary matrix $\underline{\mathbf{H}}^{(i)}(z)$ and **(b)** the transformed para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$ at the end of each iteration i of the MS-SBR2 algorithm for the cases when no truncation method is used and when the lag based truncation method is applied with a chosen set of values of μ_{PU} and μ_{PH} .

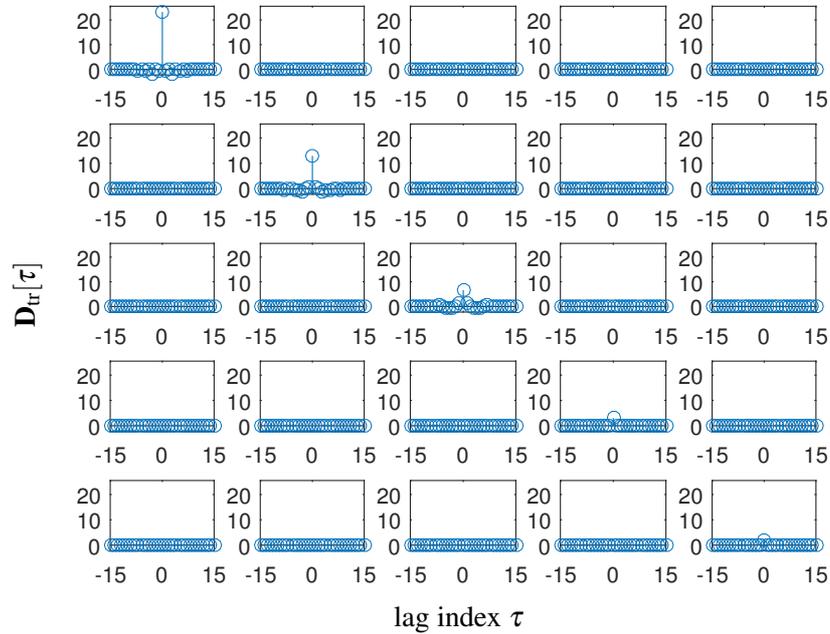


Figure 4.20 The truncated diagonal matrix $\underline{\mathbf{D}}_{tr}(z)$ produced by applying the MS-SBR2 algorithm to the CSD matrix example $\hat{\underline{\mathbf{R}}}_{xx}(z)$ shown in Figure 4.13, implementing the energy based para-Hermitian truncation method with $\mu_{PH} = 10^{-4}$.

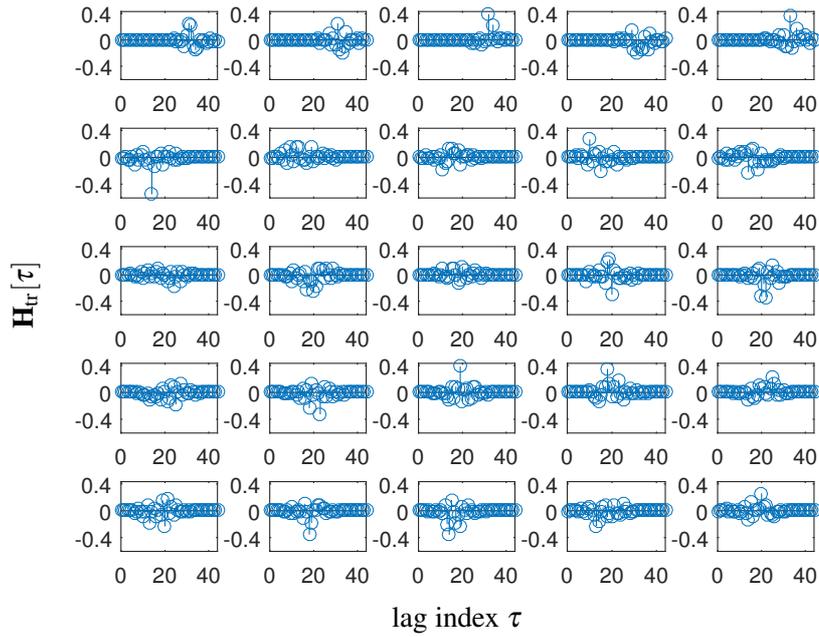


Figure 4.21 The truncated paraunitary matrix $\underline{\mathbf{H}}_{\text{tr}}(z)$ produced by applying the MS-SBR2 algorithm to the CSD matrix example $\hat{\underline{\mathbf{R}}}_{\text{xx}}(z)$ shown in Figure 4.13, implementing the lag based paraunitary truncation method with $\mu_{\text{PU}} = 10^{-4}$.

4.8 Chapter Summary

We have proposed the MS-SBR2 algorithm for calculating the PEVD of para-Hermitian matrices. Compared to the original SBR2 algorithm, MS-SBR2 can achieve the same level of diagonalisation with fewer iterations by annihilating multiple off-diagonal elements at each iteration. Also, it has been found that MS-SBR2 outperforms SBR2 in terms of the convergence speed in the studied examples. Furthermore, by using the direction-fixed shift approach, the OCMS-SBR2 algorithm has the advantage of controlling the unnecessary growth in the polynomial order and so the requirements of computational load and memory storage have been reduced, which therefore increases the computational speed of the algorithm.

Furthermore, we have investigated how differently conditioned para-Hermitian matrices, generated using the source model as discussed in Section 4.6, can affect the performance of the SBR2 and MS-SBR2 algorithms. Two types of conditioning, including the dynamic range and relations between the PSDs of the sources, have been

used to control the properties of para-Hermitian matrices. Simulations results suggest that for a lower dynamic range, the PEVD algorithms tend to converge faster in terms of reducing the off-diagonal energy. In particular, the majorised source model can produce a better diagonalisation measure than the unmajorised version, whilst maintaining a lower paraunitary order.

Finally, the proposed MS-SBR2 algorithm has been demonstrated in the application of strong decorrelation for broadband signals. In the studied example, the MS-SBR2 algorithm provides better performance than that of the SBR2 algorithm in terms of the diagonalisation measure, the order of the resulting matrices, and the computational speed if no truncation method is used. Furthermore, we have demonstrated how the truncation methods can be used to reduce the order of the polynomial matrices $\underline{\mathbf{H}}^{(i)}(z)$ and $\underline{\mathbf{R}}^{(i)}(z)$ obtained from the MS-SBR2 algorithm. Simulation results have suggested that suitable choices of the truncation parameters can be made to significantly reduce the order of the polynomial matrices whilst still maintaining the decomposition at an accurate level.

Chapter 5

Comparative Analysis of the PEVD

Algorithms

5.1 Introduction

This chapter will focus on analysing different PEVD algorithms from two main perspectives, i.e., computational complexity and algorithm performance. First, we will analyse the computational complexity of all PEVD algorithms discussed in Chapters 3 and 4. Computer simulations have been set up to examine how the proposed MS-SBR2 algorithm behaves in terms of the computational time when diagonalising an ensemble of randomised para-Hermitian matrices with different sizes. Results have been compared with those from other PEVD algorithms, including SBR2, SMD, ME-SMD, and MSME-SMD. Following on that, we will assess the different PEVD algorithms from the perspective of various performance metrics, including diagonalisation measure, convergence speed, and polynomial order. Some of these factors have already been examined and compared for the SBR2 family in Chapter 4, but not for the SMD family. Accordingly, this section will provide more thorough analysis of these factors.

5.2 Analysis of Computational Complexity

To indicate the computational complexity of the different PEVD algorithms, the “big-oh” notation $\mathcal{O}(\cdot)$, introduced by Bachmann in 1894 [94], was chosen to represent the number of multiply-accumulate (MAC) operations needed to reach the dominant order in the size parameters M and L ¹. For example, if an algorithm takes $\mathcal{O}(M^2L)$ time, it means that the algorithm performs fewer than KM^2L MAC operations for some constant K as M and $L \rightarrow \infty$. Note that in the context of polynomial matrices, the number of MAC operations is not just determined by the matrix dimension M but also by the number of lags L of the input para-Hermitian matrix $\underline{\mathbf{R}}(z) \in \mathbb{C}^{M \times M \times L}$. It is also important to note that the value of L increases as iterations continue [7, 35] and the extent of the growth varies according to the input para-Hermitian matrix $\underline{\mathbf{R}}(z)$ and aspects of the algorithm itself.

As described in Section 3.2.2, each iteration in an iterative PEVD algorithm comprises three main steps: (i) search for the dominant off-diagonal energy, (ii) time-shift the corresponding row and column onto the zero-lag plane using a paraunitary shift matrix, and (iii) annihilate the off-diagonal elements at zero-lag using a unitary rotation matrix and apply it to all other lags. As the implementation of the time-shift step mainly involves pre-allocation of matrices, few arithmetic operations are involved. However, it should be noted that the growth in polynomial order of $\underline{\mathbf{R}}(z)$ caused by the time-shift at each iteration will lead to an increase in the computational load of the key operations that account for a large proportion of an algorithm’s running time. These operations include norm calculation, comparisons for finding the maximum off-diagonal element (or the column norm in the case of SMD), and the diagonalisation step, which is also different between the SBR2 and SMD algorithm families. The computational load with respect to these operations will now be discussed.

¹An $\mathcal{O}(n)$ operation represents n -MAC operations that involve n multiplications and n additions.

5.2.1 Computational Complexity of the SBR2 Family

The SBR2 Algorithm

For the SBR2 algorithm, since the maximum off-diagonal element can be simply found by the L_∞ norm as shown in (3.13), the vector norm does not need to be calculated. The search for the maximum off-diagonal element usually requires an inspection of all the off-diagonal elements in $\underline{\mathbf{R}}(z)$. However, because of the para-Hermitian property, the maximum search can be restricted to the lower or upper triangular area across all lags of $\underline{\mathbf{R}}(z)$. Thus, the search step has order of $(M^2L - ML)/2 \approx M^2L$ time complexity assuming that one comparison for the maximum search is about as expensive as one MAC operation. Furthermore, the diagonalisation step involves $\mathcal{O}(M)$ operations for transferring the energy of the pairwise off-diagonal elements onto the diagonal at the zero-lag, which is equivalent to the cost of a single Jacobi rotation in the classical (or cyclic) Jacobi method for EVD [8], plus $\mathcal{O}(M^3(L - 1)) \approx \mathcal{O}(M^3L)$ MAC operations for applying the rotation matrix to all other lags. This results in a total diagonalisation complexity of $\mathcal{O}(M^3L)$. Therefore, the total computational complexity for a single iteration in SBR2 is dominated by the diagonalisation operation with the highest time complexity of $\mathcal{O}(M^3L)$, as shown in Table 5.1.

The MS-SBR2 Algorithm

The MS-SBR2 algorithm uses the same method to find the maxima as the SBR2 algorithm but with continuously reduced search spaces such that previously shifted maxima are not affected by later shifts [28]. This means that the cost of norm calculation is also $\mathcal{O}(0)$. However, MS-SBR2 performs a maximum of $\lfloor M/2 \rfloor$ column shifts at each iteration instead of only one column (or row) shift. Thus, the worst case search operation in each iteration is undertaken over a total of

$$\underbrace{(M^2 - M)L/2}_{1^{st} \text{ search}} + \underbrace{(M^2 - M - (4M - 6))L/2}_{2^{nd} \text{ search}} + \cdots + \underbrace{(M^2 - M - \cdots)L/2}_{\lfloor M/2 \rfloor^{th} \text{ search}} \quad (5.1)$$

elements. Again, for the purpose of asymptotic analysis, only the highest order operation is considered here, which means the time complexity of the search step in MS-SBR2 is increased by a factor of $\lfloor M/2 \rfloor$ compared to that of SBR2. Without loss of generality, we use a coefficient α_1 to represent the additional cost that arises in the “multiple shift” effect, so the overall search has $\mathcal{O}(\alpha_1 M^2 L)$ complexity, where $1 \leq \alpha_1 \leq \lfloor M/2 \rfloor$. Similarly, the diagonalisation step is $\mathcal{O}(\alpha_1 M^3 L)$ with consideration of a total of α_1 independent Jacobi rotations. Therefore, by taking the dominant operations the overall complexity for a single iteration in MS-SBR2 is $\mathcal{O}(\alpha_1 M^3 L)$.

5.2.2 Computational Complexity of the SMD Family

The SMD Algorithm

Unlike the SBR2 family, the SMD algorithm [35] aims to find the maximum L_2 norm $\|\hat{\mathbf{s}}_k^{(i-1)}[\boldsymbol{\tau}]\|_2, \forall \boldsymbol{\tau}$ and $k = 1, \dots, M$ in (3.40) at each iteration, so a total of ML column norms of the para-Hermitian matrix have to be calculated. This implies that the maximum column norm is found by a search over $\mathcal{O}(ML)$. Each L_2 norm involves a sum of squares of elements and a square root operation, but for the purpose of comparison, the square root operation can usually be omitted. Thus, with a total of M elements in each column, the norm calculation is $\mathcal{O}(M^2 L)$.

For the diagonalisation step in SMD, a full EVD is applied to the zero-lag of $\underline{\mathbf{S}}^{(i)'}(z)$, followed by the multiplications of the generated modal matrix with the remaining lags $\mathbf{S}^{(i)'}[\boldsymbol{\tau}], \boldsymbol{\tau} \neq 0$ at each iteration, as mentioned in Section 3.5.1. Here the EVD is calculated using the MATLAB `eig` function, which calls LAPACK driver routines `DSYEV` for the case of real symmetric matrices, and `ZHEEV` for Hermitian matrices (for inputs of type double) [58]. These LAPACK routines firstly reduce the zero-lag matrix $\mathbf{S}^{(i)'}[0]$ to tridiagonal form by Householder transformations which involve about $\frac{4}{3}M^3$ MAC operations [95, 96], and then find the EVD of the tridiagonal matrix, followed by the back-transformation to get the modal matrix of $\mathbf{S}^{(i)'}[0]$, which needs $2M^3$ MAC opera-

tions. The cost of solving the tridiagonal EVD varies according to the method used and the numerical values in the matrix. However, most existing software such as LAPACK takes $K_1 M^3$ operations in the worst case, where K_1 is a modest number usually varying from 4 to 12 [96]. Therefore, the number of MAC operations for the diagonalisation step is summed up as

$$\begin{aligned} \frac{4}{3}M^3 + K_1 M^3 + 2M^3 + M^3(L-1) &= \left(\frac{7}{3} + K_1\right)M^3 + M^3 L \\ &= K_2 M^3 + M^3 L, \end{aligned} \quad (5.2)$$

where $K_2 = \frac{7}{3} + K_1$ is a modest number related to K_1 . This indicates that the total cost for a single iteration in SMD is dominated by the diagonalisation step with $K_2 M^3 + M^3 L$ MAC operations. Clearly, for a significantly large value of L , the complexity is not dominated by the EVD calculation but rather by the application of the modal matrix to the rest of the lags, which is determined by the term $M^3 L$. Therefore, the diagonalisation step in SMD has $\mathcal{O}(\beta_1 M^3 L)$ time complexity in which $\beta_1 = \frac{K_2}{L} + 1 = 1$ for $L \rightarrow \infty$.

The ME-SMD Algorithm

The ME-SMD algorithm [35] replaces the L_2 norm as in the case of SMD with L_∞ to identify the maximum off-diagonal element at each iteration. By doing this, it avoids any norm calculation, but an enlarged maximum search over a set of $\mathcal{O}(M^2 L)$ elements is required. Although ME-SMD also calculates a full EVD of $\mathbf{S}^{(i)'}[0]$ at every iteration, the energy transferred onto the diagonal by ME-SMD is always smaller or equal to that eliminated by the original SMD algorithm [97]. Similar to SMD, the diagonalisation in ME-SMD requires $\mathcal{O}(\beta_2 M^3 L)$ MAC operations. Note that a different coefficient β_2 is used here in order to distinguish the complexity from that of the SMD algorithm, although they might be the same in most cases. Overall, the ME-SMD algorithm has a very similar behaviour as the original SMD algorithm in terms of complexity, which is dominated by the EVD calculation and modal matrix multiplications.

The MSME-SMD Algorithm

At the beginning of each iteration the MSME-SMD algorithm [30] is similar to ME-SMD in terms of scanning the maximum off-diagonal element based on L_∞ norm, and therefore it does not need any norm computation. However, the MSME-SMD algorithm does not just shift one column to the zero-lag at each iteration; instead, a total of $M - 1$ column shifts will be performed in order to maximise the energy transfer. For this reason, the cost of energy comparisons for MSME-SMD involves a total of $M - 1$ times searching over a set of M^2L elements, which results in $\mathcal{O}(M^3L)$. Again, $M - 1$ has been simplified to M here for the purpose of asymptotic analysis. The diagonalisation complexity of MSME-SMD is maintained the same as the other two versions of SMD, but it should be noted that the value of L varies with algorithms and grows with each iteration [97]. Similarly, a modest number β_3 is used to emphasise the difference of the complexity analysis.

Note that the computational complexity of all PEVD algorithms discussed here does not include the computation of the paraunitary matrix $\underline{\mathbf{H}}(z)$ and any order truncation process. The overall computational complexity of the different PEVD algorithms is summarised in Table 5.1.

Table 5.1 Computational complexity of the different PEVD algorithms.

Algorithm	Computational complexity of			
	norm calculation	comparisons	diagonalisation	single iteration
SBR2	$\mathcal{O}(0)$	$\mathcal{O}(M^2L)$	$\mathcal{O}(M^3L)$	$\mathcal{O}(M^3L)$
MS-SBR2	$\mathcal{O}(0)$	$\mathcal{O}(\alpha_1 M^2L)$	$\mathcal{O}(\alpha_1 M^3L)$	$\mathcal{O}(\alpha_1 M^3L)$
SMD	$\mathcal{O}(M^2L)$	$\mathcal{O}(ML)$	$\mathcal{O}(\beta_1 M^3L)$	$\mathcal{O}(\beta_1 M^3L)$
ME-SMD	$\mathcal{O}(0)$	$\mathcal{O}(M^2L)$	$\mathcal{O}(\beta_2 M^3L)$	$\mathcal{O}(\beta_2 M^3L)$
MSME-SMD	$\mathcal{O}(0)$	$\mathcal{O}(M^3L)$	$\mathcal{O}(\beta_3 M^3L)$	$\mathcal{O}(\beta_3 M^3L)$

5.3 Experimental Results and Analysis

This section will focus on assessing different PEVD algorithms in terms of complexity, convergence behaviour, and polynomial order growth. In addition, two different paraunitary matrix truncation approaches discussed in Section 3.6.3 are also compared among the PEVD algorithms by means of a set of numerical simulations.

5.3.1 Simulation Scenario and Performance Metrics

As the computational cost $\mathcal{O}(\cdot)$ of each PEVD algorithm depends on the matrix dimension M and the lag dimension L of the input para-Hermitian matrix $\mathbf{R}(z)$, we choose different values of M and L for the test matrix $\mathbf{R}(z)$ in order to see how the dimension parameters impact the efficiency of the PEVD algorithms. Specifically, an ensemble of 1000 random para-Hermitian matrices $\mathbf{R}(z) \in \mathbb{C}^{M \times M}$ with lag dimension $2L - 1$ are considered for $M = 3, 6, \dots, 30$ and $L = 50, 100, \dots, 500$. Each instance of $\mathbf{R}(z)$ is uniquely generated by $\mathbf{R}(z) = \mathbf{A}(z)\tilde{\mathbf{A}}(z)$, where $\mathbf{A}(z) \in \mathbb{C}^{M \times M \times L}$ is a random polynomial matrix whose entries are chosen to be independent and identically distributed zero mean and unit variance complex Gaussian values, i.e. $a_{jk}[\tau] \sim \mathcal{N}(0, 1), \forall \tau$ and $j, k = 1, 2, \dots, M$.

In this experiment, mean execution time $E\{t\}$ of a single iteration over an ensemble of 1000 realisations is used to examine the computational cost of each algorithm, and the simulation was implemented in MATLAB R2016a with the following desktop computer specifications: Linux Mint 17.1 with Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz \times 4 cores and 16 GB RAM. Apart from execution time, normalised off-diagonal energy $\eta^{(i)}$, as described in (4.19), is also recorded after i iterations in order to measure the level of diagonalisation by these algorithms. For the detailed definition of $\eta^{(i)}$, refer to Section 4.5.1.

5.3.2 Computational Run-Time Evaluation

The asymptotic analysis based on $\mathcal{O}(\cdot)$ gives an idea of how rapidly the running time of the PEVD algorithm grows as the dimension of $\mathbf{R}(z)$ increases. It also concisely captures the important differences in the asymptotic growth rates of different algorithms.

The real-time complexity of the SBR2 algorithm is illustrated in Figure 5.1(a): with a fixed matrix dimension M , the mean execution time $E\{t\}$ grows linearly with the increase of the lag dimension L . Similarly, with L unchanged, the SBR2 algorithm takes cubic time, which is consistent with the complexity analysis of $\mathcal{O}(M^3L)$. Figure 5.1(b) shows the real-time performance of the MS-SBR2 algorithm versus different sizes of the input para-Hermitian matrix. It can be seen clearly that MS-SBR2 is significantly slower for large size matrices compared to SBR2, but it is important to note that more off-diagonal energy is eliminated at each iteration when applying MS-SBR2.

Although the ME-SMD algorithm (based on the maximum element search) as shown in Figure 5.1(d) is designed to be less costly than the SMD algorithm (based on the maximum column norm search) as in Figure 5.1(c), they both present very similar asymptotic behaviour of execution time for different matrix sizes. This is because both algorithms operate the same EVD routine at the diagonalisation step and that step dominates the computational cost at each iteration. Figure 5.1(e) shows the real-time performance of the MSME-SMD algorithm. The steep polynomial increase with the matrix dimension M and the linear growth with the number of lags L clearly reflect the complexity analysis of $\mathcal{O}(M^3L)$. Compared to the other algorithms, the MSME-SMD algorithm requires the most execution time to complete a single iteration, especially for large matrices. This is because MSME-SMD has a total of $(M - 1)$ shifts at each iteration, which can make the polynomial order significantly large therefore dramatically increasing the computational load for the subsequent operations, including a full EVD to diagonalise the zero-lag matrix and the multiplications of the modal matrix to the other lags.

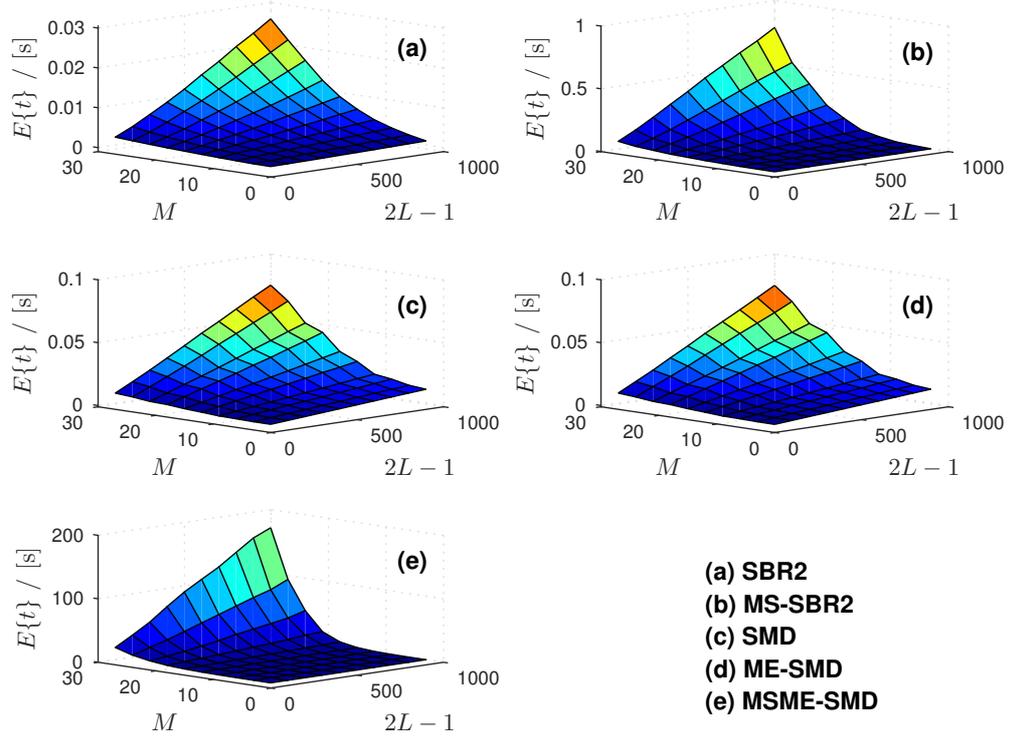


Figure 5.1 Time complexity of different PEVD algorithms for diagonalising randomly generated para-Hermitian matrices, showing mean execution time $E\{t\}$ of a single iteration for varying matrix dimensions $M = 3, 6, \dots, 30$ and lag dimensions $L = 50, 100, \dots, 500$ over an ensemble of 1000 realisations.

5.3.3 Convergence Analysis

While the previous section has shown how different matrix sizes can impact the computational complexity of the PEVD algorithms, this section will now analyse the convergence behaviour of these algorithms in terms of the computational time over 100 iterations. The para-Hermitian matrix examples used here are the same as those in Section 5.3.2, but with the matrix dimension M restricted to 6, 12, and 24 and lag length $L = 5$ only. Thus, three sets of experimental results will be presented here in accordance with the choice of test matrices.

Figure 5.2 illustrates the average remaining off-diagonal energy versus mean execution time when applying the PEVD algorithms to an ensemble of randomised para-Hermitian matrices over 100 iterations, and Table 5.2 shows the average final length of the transformed para-Hermitian matrix $\underline{\mathbf{R}}^{(100)}(z)$ and the paraunitary matrix $\underline{\mathbf{H}}^{(100)}(z)$

5.3 Experimental Results and Analysis

for when $M = 6, 12,$ and $24,$ respectively. Note that no truncation method is used when conducting this experiment. With the same amount of off-diagonal energy reduction, it can be clearly seen that the MS-SBR2 algorithm converges faster than all the other PEVD algorithms for all three cases, and the SBR2 algorithm shows a marginal fall back but still outperforms the SMD algorithm family. Despite the highest computational complexity and fastest growth in $L,$ the MSME-SMD algorithm outperforms the other two versions of SMD in both convergence speed and diagonalisation measure aspects. Although the ME-SMD algorithm is designed with a lower complexity search strategy, it does not seem to be advantageous over the conventional SMD algorithm in terms of the convergence speed, as most of the cost is dominated by the application of the EVD step in (3.39).

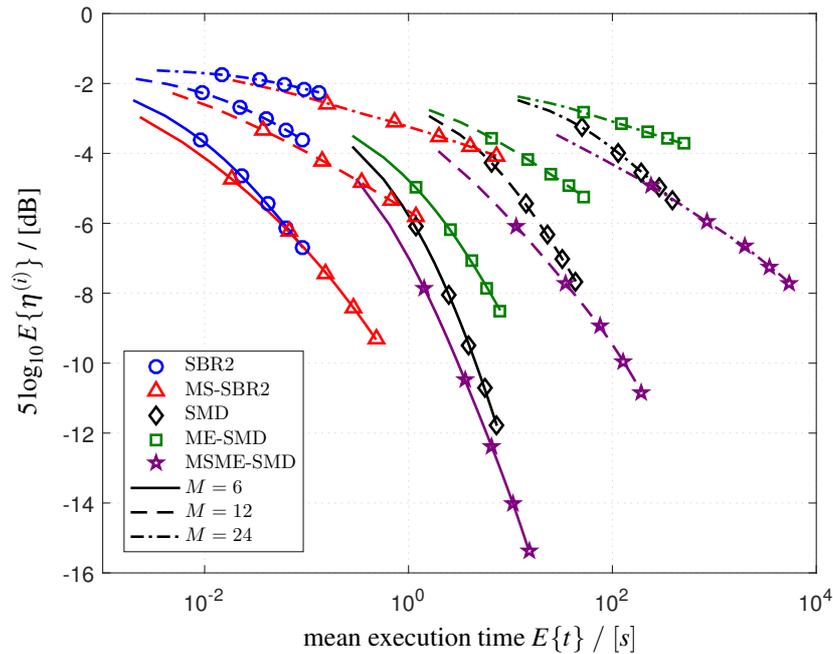


Figure 5.2 Convergence comparison of the PEVD algorithms showing the reduction of off-diagonal energy vs. mean execution time over 100 iterations for an ensemble of randomised para-Hermitian matrices with sizes $M = 6, 12,$ and $24.$

Overall, if only considering the algorithm efficiency, MS-SBR2 appears to be the best choice among these algorithms. Otherwise, MSME-SMD provides the best diagonalisation and decent convergence speed at the cost of high computational load. Upon

inspection of the average final length for both $\underline{\mathbf{R}}^{(100)}(z)$ and $\underline{\mathbf{H}}^{(100)}(z)$, it can be seen that the polynomial orders can become extremely large, especially for the MSME-SMD algorithm. Since no truncation method is used, the paraunitary matrix $\underline{\mathbf{H}}^{(i)}(z)$ always has half the amount of order growth as the para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$.

Table 5.2 Average length of the resulting polynomial matrices obtained from applying various PEVD algorithms to an ensemble of randomised para-Hermitian matrices with sizes $M = 6, 12,$ and 24 for 100 iterations, without using any truncation methods.

Algorithm	Average length of $\underline{\mathbf{R}}^{(100)}(z)$			Average length of $\underline{\mathbf{H}}^{(100)}(z)$		
	$M = 6$	$M = 12$	$M = 24$	$M = 6$	$M = 12$	$M = 24$
SBR2	809	541	406	401	267	200
MS-SBR2	2166	2697	3421	1080	1345	1707
SMD	1092	777	585	543	385	289
ME-SMD	1538	1456	1250	766	725	622
MSME-SMD	13603	39311	112536	6798	19652	56265

5.3.4 Polynomial Order Truncation

As discussed in Section 3.6, the order of the para-Hermitian and paraunitary matrices can grow unnecessarily large with increasing iterations when performing the PEVD. This can cause significant increasing of the computational complexity of the algorithm, which can subsequently slow down the computational speed. In addition, the cost of applying the finally extracted paraunitary matrix to broadband subspace based applications can be very expensive. As illustrated in Table 5.2, without order truncation involved, the final length of $\underline{\mathbf{R}}^{(100)}(z)$ is much larger than the length of the input matrix $\underline{\mathbf{R}}(z)$ (of order 8).

This section will focus on examining how the order truncation methods perform on various PEVD algorithms. The para-Hermitian matrix examples used here are the same as those in Section 5.3.1, but with dimension parameters restricted to only $M = 6$ and $L = 5$. As to the simulation, the energy based truncation method, as mentioned in Section 3.6.2, is used to shorten the para-Hermitian matrix, whilst two different

paraunitary order truncation methods, including the lag based truncation [87] and row-shift corrected truncation [83], are employed and compared for all PEVD algorithms.

It has been demonstrated in [83, 85] that the row-shift corrected truncation for paraunitary matrices has the potential benefit of achieving more aggressive order reduction than that of the lag-based method [87], whilst keeping the paraunitary reconstruction error at a similar level. The test para-Hermitian matrices $\underline{\mathbf{R}}(z)$ used in those papers are generated from the source model outlined in [35], where the ground truth PEVD comprises arbitrary paraunitary matrix factors and a spectrally majorised diagonal matrix. However, in this section, these two different paraunitary truncation methods are assessed and compared by means of the randomly generated para-Hermitian matrices, i.e., $\underline{\mathbf{R}}(z) = \underline{\mathbf{A}}(z)\tilde{\underline{\mathbf{A}}}(z)$, as mentioned in Section 5.3.1.

Truncated Para-Hermitian Order and Reconstruction Error

The truncation of the transformed para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$ at the i -th iteration of the PEVD was implemented for three specific values of μ_{PH} : (i) $\mu_{\text{PH}} = 10^{-5}$, (ii) $\mu_{\text{PH}} = 10^{-4}$, and (iii) $\mu_{\text{PH}} = 10^{-3}$. In each case, each of the PEVD algorithms is applied to an ensemble of 1000 randomly generated para-Hermitian matrices for 100 iterations, and at the i -th iteration the truncation method is applied to the untrimmed para-Hermitian matrix $\underline{\mathbf{R}}^{(i)}(z)$, resulting in the truncated version represented by $\underline{\mathbf{R}}_{\text{tr}}^{(i)}(z)$. This procedure is very similar to the simulation scenario described in Section 4.5.4. This means that there will be approximately the same amount of energy being removed from the input matrix $\underline{\mathbf{R}}(z)$ for every truncation step, i.e., $\|\underline{\mathbf{R}}_{\text{tr}}^{(i+1)}\|_{\text{F}}^2 \approx \|\underline{\mathbf{R}}_{\text{tr}}^{(i)}\|_{\text{F}}^2$, $i = 1, \dots, 99$. In other words, the order truncation is not carried out as part of the iterative process in the PEVD algorithms.

The final average order of $\underline{\mathbf{R}}_{\text{tr}}^{(100)}(z)$ for the three cases is shown in Table 5.3(a). Accordingly, the para-Hermitian matrix reconstruction error $\xi_{\text{PH}}^{(100)}$ can be calculated using (4.20), and the ensemble average results are presented in Table 5.3(b). The results presented here show the great benefit of truncating the para-Hermitian matrix.

5.3 Experimental Results and Analysis

Compared to the untrimmed results presented in Table 5.2, the order of the transformed para-Hermitian matrix has been dramatically reduced with little compromise to the accuracy of the decomposition. For example, in the case of $\mu_{\text{PH}} = 10^{-5}$, the MSME-SMD algorithm has the largest order reduction, from 13603 to 128, amounting to $1 - 128/13603 \approx 99.1\%$. The MS-SBR2 algorithm has the second largest order reduction, amounting to $1 - 84/2166 \approx 96.1\%$. When μ_{PH} is increased, more outer coefficient matrices in $\underline{\mathbf{R}}^{(100)}(z)$ will be truncated, which leads to an increase of the reconstruction error $\xi_{\text{PH}}^{(100)}$. It is interesting to see that the MSME-SMD algorithm has the most significant order reduction when μ_{PH} increased to 10^{-3} . This means that the coefficients in $\underline{\mathbf{R}}^{(100)}(z)$ are distributed more sparsely due to the multiple-shift operation in comparison with other algorithms.

Table 5.3 Average order of the truncated para-Hermitian matrix and the corresponding reconstruction error obtained from the PEVD algorithms after 100 iterations, implementing the energy based truncation method with different values of μ_{PH} .

(a) Average order of $\underline{\mathbf{R}}_{\text{tr}}^{(100)}(z)$			
Algorithm	$\mu_{\text{PH}} = 10^{-5}$	$\mu_{\text{PH}} = 10^{-4}$	$\mu_{\text{PH}} = 10^{-3}$
SBR2	59	51	42
MS-SBR2	84	70	52
SMD	73	59	36
ME-SMD	116	94	70
MSME-SMD	128	87	21

(b) Average para-Hermitian matrix reconstruction error $E\{\xi_{\text{PH}}^{(100)}\}$			
Algorithm	$\mu_{\text{PH}} = 10^{-5}$	$\mu_{\text{PH}} = 10^{-4}$	$\mu_{\text{PH}} = 10^{-3}$
SBR2	1.084×10^{-8}	9.904×10^{-7}	8.771×10^{-5}
MS-SBR2	1.512×10^{-8}	1.190×10^{-6}	7.961×10^{-5}
SMD	1.808×10^{-8}	1.537×10^{-6}	1.104×10^{-4}
ME-SMD	1.462×10^{-8}	1.057×10^{-6}	7.588×10^{-5}
MSME-SMD	5.032×10^{-8}	5.243×10^{-6}	1.486×10^{-4}

Truncated Paraunitary Order and Diagonalisation Measure

To compare these two paraunitary truncation approaches, the lag based paraunitary truncation parameter μ_{PU} is chosen from three different values, and set to be the same as the row-shift truncation parameter μ'_{PU} , i.e., $\mu_{\text{PU}} = \mu'_{\text{PU}} = 10^{-5}$, 10^{-4} , and 10^{-3} . Figure 5.3 shows the average paraunitary order and diagonalisation measures for the different PEVD algorithms and truncation methods over 100 iterations. Here, results are recorded after each iteration. As shown in Figures 5.3 (a), (c), and (e), the paraunitary order truncated using row-shift corrected methods is lower than that using the lag-based method for all PEVD algorithms. Especially for MSME-SMD, the benefit of using the row-shift truncation is much more obvious than others as iterations continue. In addition, the SBR2 family appears to have slightly better order reduction than both the SMD and ME-SMD algorithms. This is because there tend to be fewer outer lags which need to be corrected by the row-shift truncation in the SMD and ME-SMD algorithms for the studied examples. Figures 5.3 (b), (d), and (f) illustrate that for the same level of diagonalisation, SMD with the row-shift truncation produces the lowest paraunitary order for this specific example.

Table 5.4 presents the average order of the truncated paraunitary matrix $\mathbf{H}_{\text{tr}}^{(100)}(z)$ obtained from the two different truncation methods with three specific settings of the truncation parameters. It can be seen clearly that the truncated paraunitary order from the row-shift method is always smaller than that from the lag based truncation method for all PEVD algorithms.

Paraunitary Matrix Reconstruction Error

In addition to the order truncation measure, the paraunitary matrix reconstruction errors for the different truncation methods and PEVD algorithms are also recorded. With the same truncation parameter value ($\mu_{\text{PU}} = \mu'_{\text{PU}}$) for both truncation methods, each PEVD algorithm presents very similar reconstruction errors, as shown in Figure 5.4. The reconstruction error $\xi_{\text{PU}}^{(i)}$ starts at a very low value but quickly increases as the

5.3 Experimental Results and Analysis

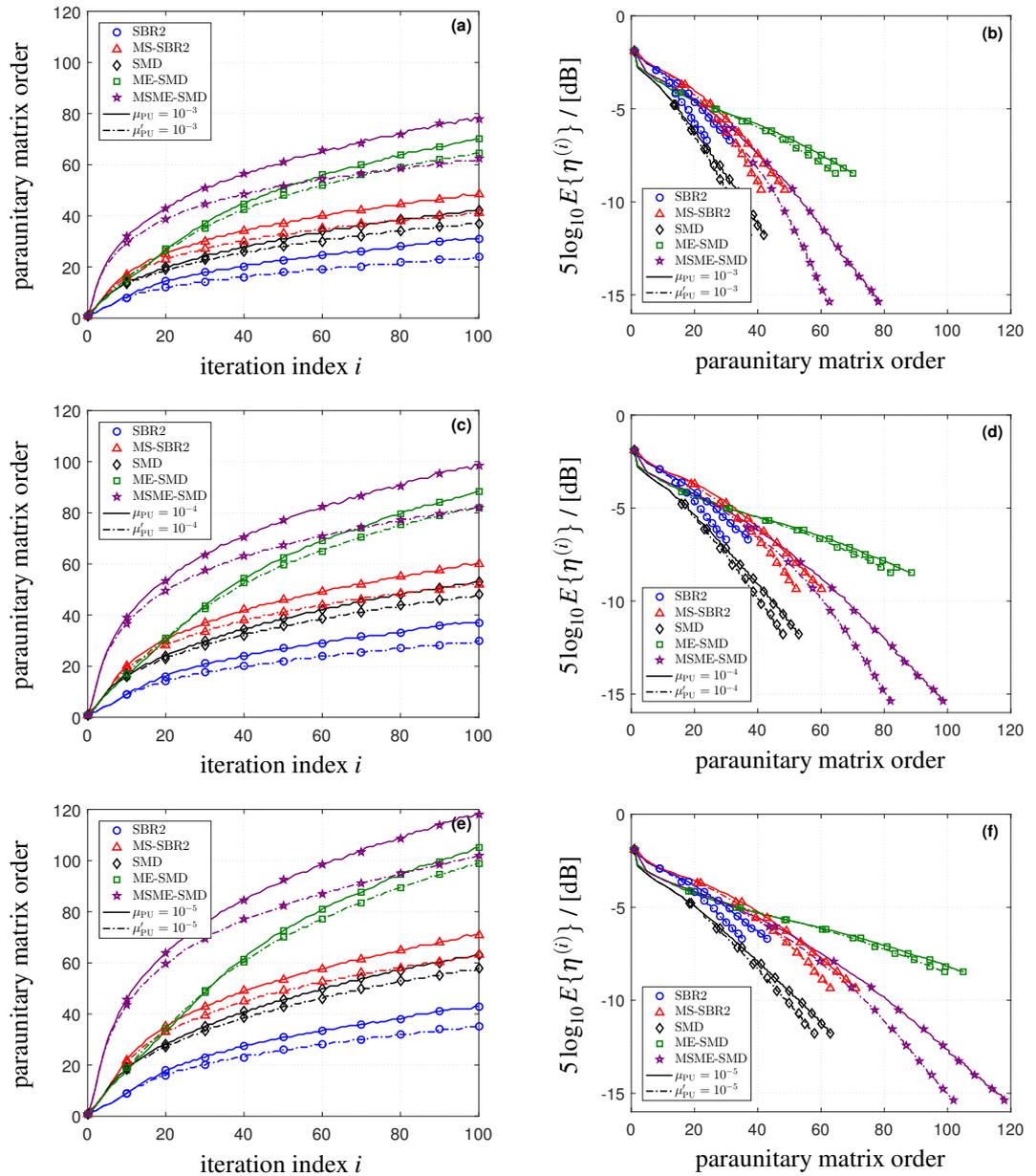


Figure 5.3 Comparison of paraunitary order truncation methods when applied to the different PEVD algorithms, showing (a), (c), and (e) average truncated paraunitary order vs. iterations, and (b), (d), and (f) diagonalisation measure vs. average truncated paraunitary order, over 1000 realisations when $\mu_{\text{PU}} = \mu'_{\text{PU}} = 10^{-3}$, 10^{-4} , and 10^{-5} , respectively.

truncation algorithms begin to remove the outer lags that contain the specified proportion of energy of the paraunitary matrix by μ_{PU} or μ'_{PU} . In addition, the growth rate of $\xi_{\text{PU}}^{(i)}$ dramatically slows after a certain number of iterations.

5.3 Experimental Results and Analysis

Table 5.4 Average order of the truncated paraunitary matrix obtained from the PEVD algorithms after 100 iterations, implementing the lag based and row-shift corrected truncation methods, respectively.

Algorithm	Average order of $\mathbf{H}_{\text{tr}}^{(100)}$ for lag based truncation $\mu_{\text{PU}} =$			Average order of $\mathbf{H}_{\text{tr}}^{(100)}$ for row-shift truncation $\mu'_{\text{PU}} =$		
	10^{-5}	10^{-4}	10^{-3}	10^{-5}	10^{-4}	10^{-3}
SBR2	43	37	31	35	30	24
MS-SBR2	71	60	48	63	52	41
SMD	63	53	42	58	48	37
ME-SMD	105	88	70	99	82	64
MSME-SMD	118	98	78	102	82	62

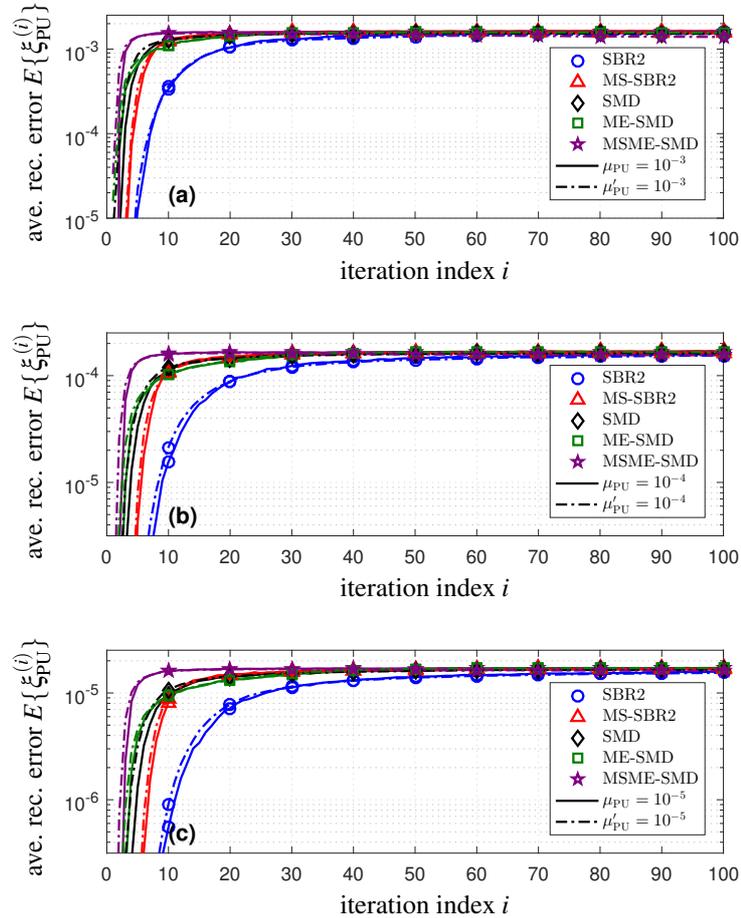


Figure 5.4 Average paraunitary matrix reconstruction error $E\{\xi_{\text{PU}}^{(i)}\}$ vs. iterations for the different truncation methods and PEVD algorithms, showing (a), (b), and (c) for different values of the truncation parameter.

5.4 Chapter Summary

The final average errors $E\{\xi_{\text{PU}}^{(100)}\}$ after 100 iterations for different truncation methods and PEVD algorithms are recorded in Table 5.5. Clearly, with the same truncation parameter settings $\mu_{\text{PU}} = \mu'_{\text{PU}}$, the row-shift corrected method has lower error for all PEVD algorithms except for the single case of ME-SMD with $\mu_{\text{PU}} = \mu'_{\text{PU}} = 10^{-5}$, in which the lag based truncation shows a slightly better error result than the row-shift method.

Table 5.5 Average paraunitary matrix reconstruction error obtained from the PEVD algorithms after 100 iterations, showing (a) implementing the lag based truncation with μ_{PU} and (b) the row-shift corrected truncation with μ'_{PU} .

(a) Lag based truncation			
Algorithm	Average paraunitary matrix reconstruction error $E\{\xi_{\text{PU}}^{(100)}\}$ for		
	$\mu_{\text{PU}} = 10^{-5}$	$\mu_{\text{PU}} = 10^{-4}$	$\mu_{\text{PU}} = 10^{-3}$
SBR2	1.576×10^{-5}	1.608×10^{-4}	1.595×10^{-3}
MS-SBR2	1.718×10^{-5}	1.703×10^{-4}	1.637×10^{-3}
SMD	1.679×10^{-5}	1.649×10^{-4}	1.576×10^{-3}
ME-SMD	1.698×10^{-5}	1.657×10^{-4}	1.580×10^{-3}
MSME-SMD	1.699×10^{-5}	1.665×10^{-4}	1.615×10^{-3}

(b) Row-shift corrected truncation			
Algorithm	Average paraunitary matrix reconstruction error $E\{\xi_{\text{PU}}^{(100)}\}$ for		
	$\mu'_{\text{PU}} = 10^{-5}$	$\mu'_{\text{PU}} = 10^{-4}$	$\mu'_{\text{PU}} = 10^{-3}$
SBR2	1.545×10^{-5}	1.543×10^{-4}	1.513×10^{-3}
MS-SBR2	1.666×10^{-5}	1.625×10^{-4}	1.528×10^{-3}
SMD	1.643×10^{-5}	1.606×10^{-4}	1.515×10^{-3}
ME-SMD	1.710×10^{-5}	1.655×10^{-4}	1.560×10^{-3}
MSME-SMD	1.613×10^{-5}	1.551×10^{-4}	1.392×10^{-3}

5.4 Chapter Summary

In this chapter, the complexities of the different PEVD algorithms, including the SBR2 and SMD families, have been analysed and compared in detail. With the simple Jacobi

rotation, the SBR2 algorithm has the lowest complexity among all the PEVD algorithms; however, it can only transfer a small amount of off-diagonal energy onto the diagonal at each iteration. For the SMD family, as a full EVD is applied to the zero-lag matrix at each iteration, there is usually more off-diagonal energy transferred onto the diagonal compared to the SBR2 algorithm, but using a full EVD also significantly increases the complexity of the SMD family, especially for matrices with large sizes. When the more complex multiple-shift strategy is introduced, the MS-SBR2 algorithm can transfer more energy at each iteration, which makes the algorithm converge faster than SBR2 in terms of reducing the off-diagonal energy. Although the complexity of MS-SBR2 has been increased, the MS-SBR2 algorithm is the fastest in terms of computational time, as shown by the studied examples. Furthermore, the MSME-SMD algorithm can achieve the best diagonalisation measure; however, this comes at the cost of the highest complexity and extremely large polynomial order. Although the maximum element based ME-SMD algorithm is designed to have lower complexity than the column norm based SMD algorithm, it does not seem to be advantageous in terms of the computational speed. On the contrary, the SMD algorithm is computationally faster and can usually bring more off-diagonal energy onto the zero-lag plane according to the experimental results.

As the order of polynomial matrices will generally increase after each iteration of the PEVD, and the extent of the order growth is determined by both the algorithm and the input para-Hermitian matrix $\mathbf{R}(z)$, it is very difficult to analyse the exact complexities for each of the PEVD algorithms after running a number of iterations. It is also important to note that the truncation methods have not been taken into account when analysing the computational complexity. Nonetheless, the main complexity of the truncation methods lies in the calculation of norms, and the amount of norm calculations is determined by the specified upper bound proportion of the squared Frobenius norm of the polynomial matrix, i.e., the truncation parameter μ .

Simulations have demonstrated that the order of the polynomial matrices obtained from the PEVD algorithms can be drastically reduced using the truncation approaches whilst still maintaining a decent level of accuracy. The two different paraunitary truncation methods mentioned in Section 3.6.3 have also been compared across all PEVD algorithms by means of numerical examples. Experimental results have shown that the row-shift corrected truncation method outperforms the lag based method in terms of reducing the paraunitary order, and more interestingly, the paraunitary reconstruction error is even found to be slightly lower for the row-shift truncation method in most of the studied cases.

Chapter 6

Multichannel Spectral Factorisation using the Polynomial Eigenvalue Decomposition Method

6.1 Introduction

This chapter presents one potential application of polynomial eigenvalue decomposition (PEVD) to the multichannel spectral factorisation problem. Spectral factorisation arises in the analysis and design of linear systems, such as finding a causal system (or minimum-phase system) with a given spectral density function, which has attracted significant interest in digital signal processing and communications in recent years. Applications of spectral factorisation have been found in many areas, such as designing quadrature-mirror filter banks [98], optimum transmit-and-receive filter matrices for precoding and equalisation of multiple input multiple output (MIMO) communications [99], minimum-phase finite impulse response (FIR) precoders for multicasting MIMO frequency-selective channels [100], and quadrature amplitude modulation filter bank based multi-carrier (QAM-FBMC) systems [101].

There exist a number of algorithms for calculating spectral factorisation, such as a Newton-Raphson based method proposed by Wilson for the scalar case [102] and the polynomial matrix case [103], and the spectral factorisation algorithm developed by Janashia et al. [104]. A paper written by Kučera [105] illustrated some major parametric methods for calculating the spectral factorisation, including Toeplitz matrix decomposition and Newton-Raphson iterations. Furthermore, a survey of spectral factorisation methods was presented by Sayed and Kailath [106] that briefly reviewed different methods including the Bauer method, the Schur algorithm, the Levinson-Durbin algorithm, and techniques based on the Riccati equation, the Kalman filter, and so on. Most of these algorithms, with the exception of those of Wilson and Janashia, do not extend to the multichannel situation. Wilson's algorithm seems to provide a viable approach to the multichannel spectral factorisation problem in terms of stability and reliability but is reputed to run into problems when the number of channels grows too large.

In this chapter, we introduce a novel approach to compute the multichannel spectral factorisation. The proposed method employs a PEVD algorithm to break down the multichannel spectral factorisation problem into a set of independent single channel spectral factorisation problems for which suitable algorithms already exist. In effect, it transforms the multichannel spectral factorisation problem into one which is much easier to solve. The proposed method can be used to calculate the approximate spectral factor of any para-Hermitian polynomial matrix. Numerical examples are presented to demonstrate its ability to find the valid spectral factor and to demonstrate the level of accuracy which can be achieved. Furthermore, the fundamental indeterminacy of spectral factorisation has been exploited in order to minimise the order of the resulting spectral factor.

The rest of this chapter is structured as follows. We start by introducing the concept of one-dimensional spectral factorisation in Section 6.2, followed by the multichannel spectral factorisation and our proposed method in Section 6.3. Numerical examples are presented in Section 6.4 and conclusions are drawn in Section 6.5.

6.2 One-Dimensional Spectral Factorization

The one-dimensional spectral factorisation problem can be stated briefly as follows. Given a data sequence $g[n]$, derive an associated causal sequence $h[n]$ such that

$$g[n] = h[n] \otimes h^*[-n], \quad (6.1)$$

or equivalently in z -transform

$$G(z) = H(z)H^*(1/z) = G^+(z)G^-(z). \quad (6.2)$$

Both $g[n]$ and $h[n]$ represent discrete digital sequences. Note that the sequence $g[n]$ constitutes the autocorrelation of $h[n]$, and only when $g[n]$ is a symmetric sequence which satisfies $g[n] = g^*[-n]$ can it be factored as in (6.1). Equation (6.2) can be seen as the product of an *outer* spectral factor $G^+(z)$ and an *inner* spectral factor $G^-(z)$ [104]. Finding the spectral factor of $g[n]$ corresponds to evaluating all the roots of $G(z)$ expressed as a proper polynomial in terms of z . For example, considering an FIR filter of length L with its z -transform given by

$$G(z) = \sum_{n=0}^{L-1} g[n]z^{-n} = g[0] + g[1]z^{-1} + \dots + g[L-1]z^{-(L-1)}, \quad (6.3)$$

we can rewrite this function by multiplying both sides with z^{L-1} , such that

$$G(z) = \frac{g[0]z^{L-1} + g[1]z^{L-2} + \dots + g[L-1]}{z^{L-1}}. \quad (6.4)$$

This then constitutes a proper polynomial as required, and hence the zeros of this polynomial are the roots of the numerator in (6.4).

In general, there is an infinite number of sequences that share the same autocorrelation as in (6.1). This leaves us with a large ambiguity. One way to resolve this ambiguity is to insist on a causal function, which is also known as the minimum-phase solution.

Since there is only one possible minimum-phase function with a given autocorrelation, the minimum-phase solution determines the uniqueness of one-dimensional spectral factorisation. Therefore, to find the minimum-phase solution, corresponding to a stable filter, only the roots inside the unit circle, $|z| < 1$, and half of those roots on the unit circle, $|z| = 1$, can be chosen [107]. Then the problem remains to find $h[n]$ from the selected roots, which can be easily solved using, for example, Wilson's algorithm [102].

6.3 Multichannel Spectral Factorisation

One-dimensional spectral factorisation is only suitable for single input and single output (SISO) systems. When it comes to MIMO systems, multichannel spectral factorisation (or matrix spectral factorisation) is needed. The original approach for solving multichannel spectral factorisation was proposed in [108] by Wiener. Since then, tens of different algorithms have appeared in the literature (see the survey papers [105, 106]). All existing methods have the same necessary and sufficient conditions for the existence of spectral factorisation, which states that if a para-Hermitian polynomial matrix $\underline{\mathbf{R}}(z) \in \mathbb{C}^{M \times M}$ is positive definite on the unit circle, that is, $\underline{\mathbf{R}}(z) > 0, \forall |z| = 1$, and if $\det\{\underline{\mathbf{R}}(e^{j\theta})\}$ for $z = e^{j\theta}$ satisfies the Paley-Wiener condition [104, 108–110]

$$\int_{-\pi}^{\pi} \ln \det\{\underline{\mathbf{R}}(e^{j\theta})\} d\theta < \infty, \quad (6.5)$$

then $\underline{\mathbf{R}}(z)$ has a spectral factorisation such that

$$\underline{\mathbf{R}}(z) = \underline{\mathbf{R}}^+(z)\underline{\mathbf{R}}^-(z) = \underline{\mathbf{R}}^+(z)\tilde{\underline{\mathbf{R}}}^+(z), \quad (6.6)$$

where $\underline{\mathbf{R}}^+(z)$ and $\underline{\mathbf{R}}^-(z)$ are respectively defined as an *outer* and *inner* spectral factor [104], and $\underline{\mathbf{R}}^-(z)$ is the paraconjugate of $\underline{\mathbf{R}}^+(z)$, i.e., $\underline{\mathbf{R}}^-(z) = \tilde{\underline{\mathbf{R}}}^+(z)$.

6.3.1 Ambiguity of Multichannel Spectral Factorisation

The spectral factor $\underline{\mathbf{R}}^+(z)$ (or $\underline{\mathbf{R}}^-(z)$) in (6.6) is not unique due to the fundamental indeterminacy in spectral factorisation whereby if $\underline{\mathbf{R}}^+(z)$ is a valid *outer* spectral factor of $\underline{\mathbf{R}}(z)$, so also is $\bar{\underline{\mathbf{R}}}^+(z) = \underline{\mathbf{R}}^+(z)\underline{\mathbf{U}}(z)$ where $\underline{\mathbf{U}}(z)$ represents any paraunitary polynomial matrix which preserves the essential properties associated with an *outer* spectral factor. It follows that

$$\underline{\mathbf{R}}(z) = \bar{\underline{\mathbf{R}}}^+(z)\bar{\underline{\mathbf{R}}}^-(z) = \underline{\mathbf{R}}^+(z)\underline{\mathbf{U}}(z)\tilde{\underline{\mathbf{U}}}(z)\underline{\mathbf{R}}^-(z) = \underline{\mathbf{R}}^+(z)\underline{\mathbf{R}}^-(z). \quad (6.7)$$

This includes simple examples such as $\underline{\mathbf{U}}(z) = z^T \mathbf{I}$, $\underline{\mathbf{U}}(z) = \mathbf{S}$ where \mathbf{S} is a simple unitary matrix, or the case in which $\underline{\mathbf{U}}(z)$ takes the form of a diagonal matrix with each entry given by a power of z which need not be the same for all entries.

Similar to the scalar spectral factorisation, the unique solution of matrix spectral factorisation is also a key problem which has been frequently studied. Some of the latest papers, including [99], [104], and [111], suggest that the *outer* spectral factor $\underline{\mathbf{R}}^+(z)$ in (6.6) is unique up to a constant unitary factor \mathbf{C} , such that

$$\underline{\mathbf{R}}_c^+(z) = \underline{\mathbf{R}}^+(z)\mathbf{C}, \quad (6.8)$$

and the unique spectral factor $\underline{\mathbf{R}}_c^+(z)$ is positive definite at the origin, i.e., $\underline{\mathbf{R}}_c^+(0) > 0$. Furthermore, it admits the following conditions:

1. $\det\{\underline{\mathbf{R}}_c^+(z)\} \neq 0, \forall |z| < 1$;
2. the coefficient matrix $\underline{\mathbf{R}}_c^+[0]$ is lower triangular with unit diagonal entries.

However, the spectral factor found by our proposed method does not guarantee the unique structure as mentioned above because of the polynomial order growth in the PEVD algorithms. In addition, there is another ambiguity in the paraunitary matrix $\mathbf{H}(z)$ obtained from the PEVD algorithms, as discussed in Section 3.2.1, so the resulting spectral factor may vary across different PEVD algorithms. Having said that,

when the *outer* and *inner* spectral factors are multiplied together, the reconstructed para-Hermitian matrix $\hat{\mathbf{R}}(z) = \mathbf{R}^+(z)\mathbf{R}^-(z)$ is nonetheless accurate. This proves that our proposed method is valid and feasible in terms of solving the multichannel spectral factorisation. Details of the proposed multichannel spectral factorisation method will now be discussed.

6.3.2 Outline of the Proposed Algorithm

Given a para-Hermitian matrix $\mathbf{R}(z) \in \mathbb{C}^{M \times M}$, the proposed multichannel spectral factorisation method starts by diagonalising $\mathbf{R}(z)$ using a PEVD algorithm. After a sufficient number of iterations, the PEVD algorithm generates the diagonal matrix $\mathbf{D}(z)$ and the corresponding paraunitary matrix $\mathbf{H}(z)$. Each of the diagonal entries $\underline{d}_i(z)$, $i = 1, \dots, M$ can then be factored as the product of its *outer* and *inner* spectral factors, such that

$$\begin{aligned} \mathbf{D}(z) &= \text{diag}\{\underline{d}_1(z), \underline{d}_2(z), \dots, \underline{d}_M(z)\} \\ &= \text{diag}\{\underline{d}_1^+(z), \dots, \underline{d}_M^+(z)\} \text{diag}\{\underline{d}_1^-(z), \dots, \underline{d}_M^-(z)\} \\ &= \mathbf{D}^+(z)\mathbf{D}^-(z), \end{aligned} \quad (6.9)$$

where $\underline{d}_i^+(z)$ and $\underline{d}_i^-(z)$ for $i = 1, \dots, M$ are the *outer* and *inner* spectral factors of $\underline{d}_i(z)$, respectively.

Each polynomial element $\underline{d}_i(z)$ within $\mathbf{D}(z)$ defines a single-channel spectral factorisation problem. In effect, the PEVD transformation breaks the multichannel spectral factorisation problem down into a set of distinct single-channel spectral factorisation problems. In this thesis, the single-channel spectral factorisation of $\underline{d}_i(z)$ is calculated using the Newton-Raphson method, as adopted for the $\text{spf}(\cdot)$ function in the MATLAB PolyX toolbox [112].

The spectral factors of the diagonal matrix $\mathbf{D}(z)$ in (6.9) are then used to construct the spectral factor of the input para-Hermitian matrix $\mathbf{R}(z)$. By applying the inverse

decomposition to the PEVD equation in (2.26), we have

$$\underline{\mathbf{R}}(z) \approx \underline{\tilde{\mathbf{H}}}(z)\underline{\mathbf{D}}(z)\underline{\mathbf{H}}(z), \quad (6.10)$$

and on substituting (6.9) into (6.10), this equation can be rewritten as

$$\underline{\mathbf{R}}(z) = \underline{\mathbf{R}}^+(z)\underline{\mathbf{R}}^-(z) \approx \underline{\tilde{\mathbf{H}}}(z)\underline{\mathbf{D}}^+(z)\underline{\mathbf{D}}^-(z)\underline{\mathbf{H}}(z). \quad (6.11)$$

Note that the paraunitary matrix $\underline{\mathbf{H}}(z)$ satisfies $\det\{\underline{\mathbf{H}}(z)\} = az^{-\Delta}$, $|a| = 1$ [69], and the transformation required to generate $\underline{\mathbf{R}}^+(z)$ does not affect the *outer* spectral factor property of $\underline{\mathbf{D}}^+(z)$, such that

$$\begin{aligned} \det\{\underline{\mathbf{R}}^+(z)\} &= \det\{\underline{\tilde{\mathbf{H}}}(z)\underline{\mathbf{D}}^+(z)\} = az^{-\Delta}\det\{\underline{\mathbf{D}}^+(z)\} \\ &= az^{-\Delta}\prod_{i=1}^M d_i^+(z) \neq 0, \quad \forall |z| < 1. \end{aligned} \quad (6.12)$$

Thus,

$$\det\{\underline{\mathbf{D}}^+(z)\} \neq 0, \quad \forall |z| < 1, \quad (6.13)$$

which means $\underline{\mathbf{R}}(z)$ has to be full rank when applying the PEVD. Therefore, the resulting *outer* spectral factor $\underline{\mathbf{R}}^+(z)$ can be estimated as $\underline{\tilde{\mathbf{H}}}(z)\underline{\mathbf{D}}^+(z)$, and the *inner* spectral factor $\underline{\mathbf{R}}^-(z)$ can be estimated as $\underline{\mathbf{D}}^-(z)\underline{\mathbf{H}}(z)$, which is the paraconjugate of $\underline{\tilde{\mathbf{H}}}(z)\underline{\mathbf{D}}^+(z)$.

6.3.3 Order Shortening of the Spectral Factor

As the polynomial orders of $\underline{\mathbf{H}}(z)$ and $\underline{\mathbf{D}}(z)$ can grow unnecessarily large after the iterative paraunitary transformations using PEVD, the computed spectral factors in (6.11) can accumulate time delays which are unnecessarily large. Spectral factors with large polynomial orders are problematic, as they can be very costly to implement in minimum-phase filter based applications as mentioned previously.

To keep the order of the spectral factors as low as possible, the truncation methods mentioned in Section 3.6 are employed to shorten the orders of $\underline{\mathbf{H}}(z)$ and $\underline{\mathbf{D}}(z)$. For the results presented in the next section, the order truncation will only be implemented once after the MS-SBR2 algorithm converges. In particular, lag bound fixed truncation [67] for the diagonal matrix $\underline{\mathbf{D}}(z)$ is used, whilst both paraunitary matrix truncation methods, including the lag based truncation [87] and the row-shift corrected truncation [83], are used to truncate the resulting paraunitary matrix $\underline{\mathbf{H}}(z)$. The performance of each paraunitary truncation method will be examined based on the order and the accuracy of the resulting spectral factor.

6.4 Numerical Examples

This section focuses on demonstrating the concept and feasibility of the MS-SBR2 algorithm based spectral factorisation method by means of two worked examples.

6.4.1 Example 1

A simple 2×2 para-Hermitian matrix used by Janashia et al. [104] was chosen as the first test example, where the matrix is given by

$$\underline{\mathbf{R}}_1(z) = \begin{bmatrix} 2z^{-1} + 6 + 2z & 7z^{-1} + 22 + 11z \\ 11z^{-1} + 22 + 7z & 38z^{-1} + 84 + 38z \end{bmatrix}. \quad (6.14)$$

With the stopping condition, mentioned in (4.27), set as $10^{-4} \times \sqrt{\frac{1}{2} \sum_{m=1}^2 |r_{mm}[0]|^2} \approx 0.006$, the MS-SBR2 algorithm takes 153 iterations to diagonalise this matrix, and the resulting diagonal matrix $\underline{\mathbf{D}}_1(z)$ after truncation is given by

$$\underline{\mathbf{D}}_1(z) \approx \begin{bmatrix} 40z^{-1} + 90 + 40z & 0 \\ 0 & -0.01z^{-1} + 0.03 - 0.01z \end{bmatrix}. \quad (6.15)$$

6.4 Numerical Examples

Note that the lag bound fixed truncation method in Section 3.6.2 was employed to truncate any small coefficients in the diagonalised para-Hermitian matrix, and the order of $\underline{\mathbf{D}}_1(z)$ has been shortened to the same length as the input para-Hermitian matrix $\underline{\mathbf{R}}_1(z)$ (order of 2). Figure 6.1(a) and (b) respectively show the stem plot of the diagonal matrix $\underline{\mathbf{D}}_1(z)$ and the truncated paraunitary matrix $\underline{\mathbf{H}}_1(z)$ to which the lag based truncation was applied with $\mu_{\text{PU}} = 10^{-4}$. Figure 6.2 illustrates the power spectral density of the on-diagonal polynomials in $\underline{\mathbf{D}}_1(z)$. Upon inspection of these plots, the para-Hermitian matrix $\underline{\mathbf{R}}_1(z)$ has been fully diagonalised whilst $\underline{\mathbf{D}}_1(z)$ has achieved spectral majorisation. Also, $\underline{\mathbf{R}}_1(z)$ is almost generically rank deficient but not quite. Accordingly

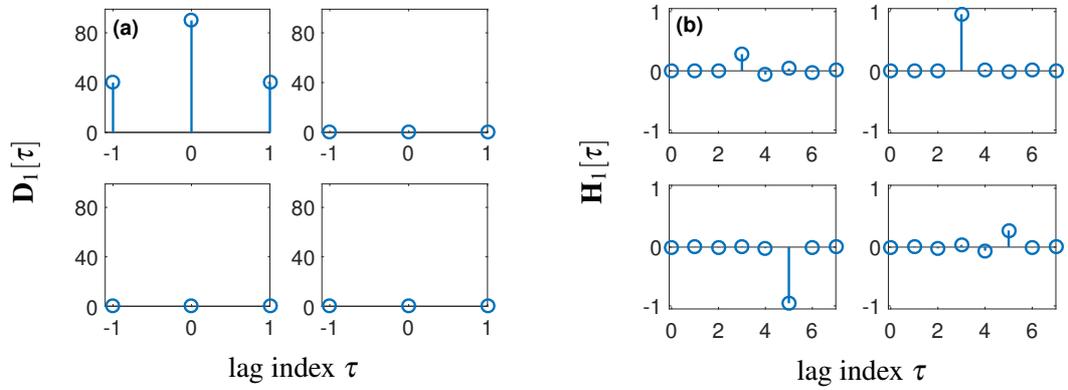


Figure 6.1 The resulting matrices obtained from applying the MS-SBR2 algorithm to Example (6.14), showing (a) the diagonalised matrix with lag bound fixed truncation and (b) the truncated paraunitary matrix resulting from lag based truncation.

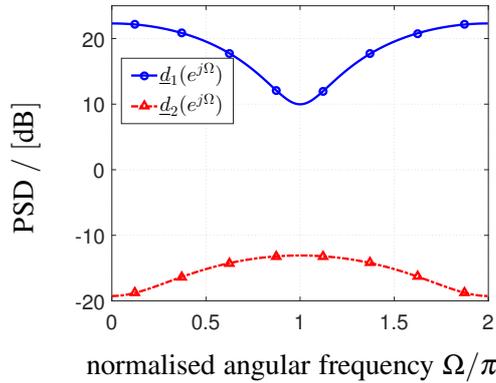


Figure 6.2 Power spectral density for the on-diagonal polynomials in $\underline{\mathbf{D}}_1(z)$ obtained from MS-SBR2.

the *outer* spectral factor $\underline{\mathbf{D}}_1^+(z)$ can be obtained by independently applying the one-

6.4 Numerical Examples

dimensional spectral factorisation to each of the two diagonal entries in $\underline{\mathbf{D}}_1(z)$, which is expressed as

$$\underline{\mathbf{D}}_1^+(z) = \text{diag}\{\underline{d}_1^+(z), \underline{d}_2^+(z)\} \approx \begin{bmatrix} 8.096 + 4.942z & 0 \\ 0 & 0.165 - 0.056z \end{bmatrix}. \quad (6.16)$$

Note that the spectral factors $\underline{d}_1^+(z)$ and $\underline{d}_2^+(z)$ obtained from the $\text{spf}(\cdot)$ function are the minimum-phase solutions, and these results are all quoted to the standard accuracy provided by the MATLAB PolyX toolbox [112]. By forming the product $\tilde{\mathbf{H}}_1(z)\underline{\mathbf{D}}_1^+(z)$, the final *outer* spectral factor $\underline{\mathbf{R}}_1^+(z)$ is shown in Figure 6.3(a), alongside its paraconjugate, i.e., the *inner* spectral factor $\underline{\mathbf{R}}_1^-(z)$ in Figure 6.3(b).

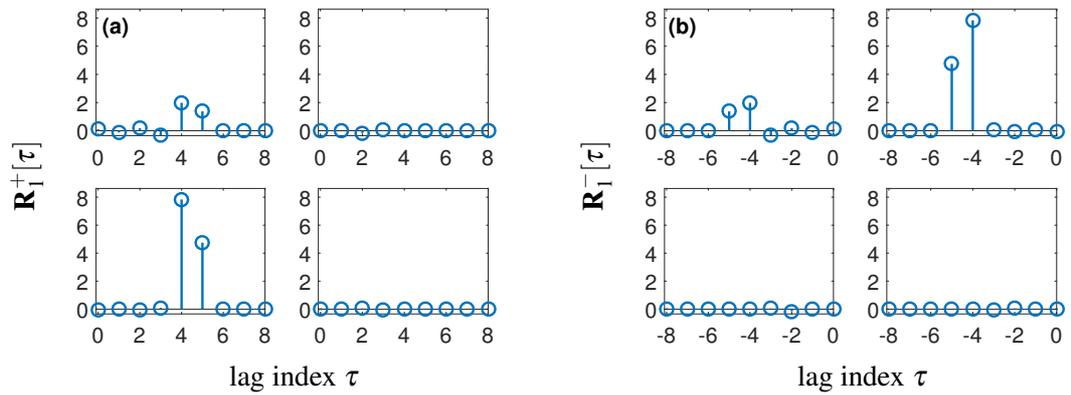


Figure 6.3 The resulting spectral factors of Example (6.14), showing (a) the *outer* spectral factor and (b) the *inner* spectral factor.

As can be seen from the stem plot, the order of the spectral factor was increased due to the paraunitary transformation in the MS-SBR2 algorithm. This does not seem to follow the standard rule that if $\underline{\mathbf{R}}(z)$ has order of $2T$, its spectral factor $\underline{\mathbf{R}}_1^+(z)$ (or $\underline{\mathbf{R}}_1^-(z)$) should theoretically have order of T . However, most of the coefficients in the resulting spectral factors are very small values which are effectively zero. The dominant coefficients in $\underline{\mathbf{R}}^+(z)$ are only seen in two lags, which are given by

$$\dots + \begin{bmatrix} 1.9806 & 0.0032 \\ 7.8366 & 0.0097 \end{bmatrix} z^4 + \begin{bmatrix} 1.3753 & -0.0020 \\ 4.7509 & -0.0059 \end{bmatrix} z^5 + \dots \quad (6.17)$$

Compared to the spectral factorisation result from Janashia's paper [104], the result obtained using the PEVD based method looks quite different. To assess the relative error of our method, we examine the energy difference between the input para-Hermitian matrix $\underline{\mathbf{R}}(z)$ and the reconstructed matrix $\hat{\underline{\mathbf{R}}}(z) = \underline{\mathbf{R}}^+(z)\underline{\mathbf{R}}^-(z)$, which is defined as

$$v = \frac{\|\underline{\mathbf{R}}(z) - \hat{\underline{\mathbf{R}}}(z)\|_{\mathbb{F}}^2}{\|\underline{\mathbf{R}}(z)\|_{\mathbb{F}}^2} = \frac{\|\underline{\mathbf{R}}(z) - \underline{\mathbf{R}}^+(z)\underline{\mathbf{R}}^-(z)\|_{\mathbb{F}}^2}{\|\underline{\mathbf{R}}(z)\|_{\mathbb{F}}^2}. \quad (6.18)$$

In this example, the error is calculated as $v_1 \approx 1.1498 \times 10^{-8}$. When the para-unitary matrix is truncated using the row-shift method, the order of $\underline{\mathbf{H}}_1(z)$ is further reduced to 5, while the error is increased to $v_1 \approx 2.0919 \times 10^{-5}$. Therefore, truncating the very small coefficients in $\underline{\mathbf{D}}_1(z)$ and $\underline{\mathbf{H}}_1(z)$ was found to have very little impact on the accuracy of the reconstructed para-Hermitian matrix (almost identical to $\hat{\underline{\mathbf{R}}}_1(z)$).

6.4.2 Example 2

The algorithm has been tested further by means of another, more realistic example. In fact, this example is the same as the one which we used to conduct the strong decorrelation simulation in Section 4.7, where the para-Hermitian matrix $\underline{\mathbf{R}}_2(z) \in \mathbb{C}^{5 \times 5}$ of order 20 was obtained from a convolutive mixing model with four sources and five sensors, and its stem plot is shown in Figure 4.13. The MS-SBR2 algorithm, when applied to this matrix, took 1507 iterations to converge to a point where the maximum off-diagonal element in $\underline{\mathbf{D}}_2(z)$ is less than $10^{-4} \times \sqrt{\frac{1}{2} \sum_{m=1}^5 |r_{mm}[0]|^2} \approx 9.6186 \times 10^{-4}$. The resulting order of the matrices $\underline{\mathbf{D}}_2(z)$ and $\underline{\mathbf{H}}_2(z)$ are 132,876 and 66,428, respectively.

The matrix $\underline{\mathbf{D}}_2(z)$ was then truncated to the same order as $\underline{\mathbf{R}}_2(z)$ using the lag bound fixed truncation method, and its stem plot is shown in Figure 6.4. As to the truncation of $\underline{\mathbf{H}}_2(z)$, the lag based truncation was applied using $\mu_{\text{PU}} = 10^{-2}$, and the truncated matrix $\underline{\mathbf{H}}_2(z)$ is of order 345. However, when the row-shift corrected truncation was

used with the same truncation value $\mu'_{\text{PU}} = 10^{-2}$, the matrix $\underline{\mathbf{H}}'_2(z)$ only has order of 41.

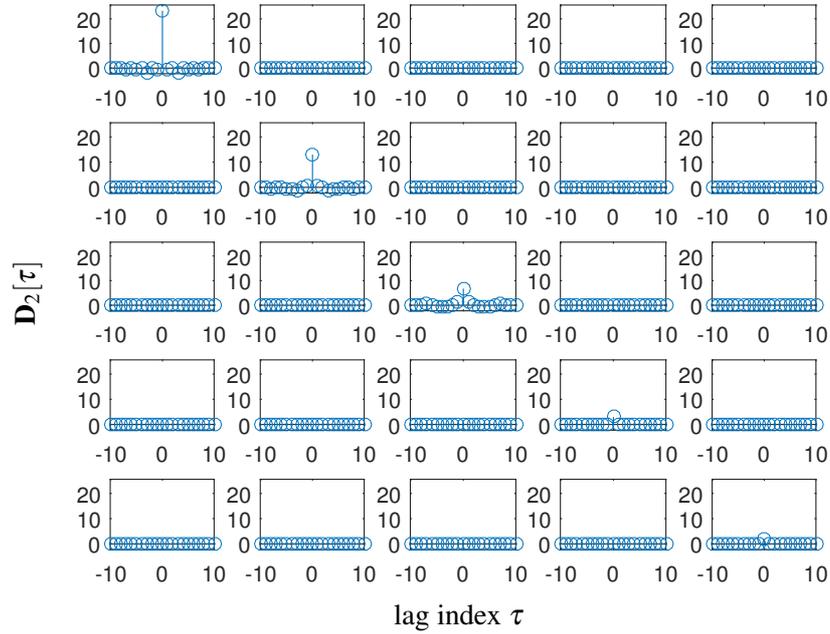


Figure 6.4 The diagonalised matrix $\underline{\mathbf{D}}_2(z)$ obtained from applying the MS-SBR2 algorithm to the CSD matrix example in Figure 4.13 and truncated to the same order as the CSD matrix.

Accordingly, each of the diagonal elements in $\underline{\mathbf{D}}_2(z)$ can now be considered as a scalar spectral factorisation problem and addressed by Wilson's method. The *outer* spectral factor $\underline{\mathbf{R}}_2^+(z)$ generated by forming the product $\tilde{\underline{\mathbf{H}}}_2(z)\underline{\mathbf{D}}_2^+(z)$ is shown in Figure 6.5. Similarly, the *outer* spectral factor $\underline{\mathbf{R}}_2'^+(z)$, with the row-shift truncation for $\underline{\mathbf{H}}_2(z)$, is plotted in Figure 6.6. Clearly, by using the row-shift truncation, the order of the spectral factor $\underline{\mathbf{R}}_2'^+(z)$ is much less than that of $\underline{\mathbf{R}}_2^+(z)$.

By analogy to the previous example, the accuracy of these two spectral factors for each of the paraunitary matrix truncation cases is examined according to (6.18), and the relative errors are given by $v_2 \approx 2.7630 \times 10^{-3}$ for the lag based truncation case and $v_2' \approx 6.5403 \times 10^{-3}$ for the row-shift truncation case, respectively. Clearly, without losing too much accuracy, a significant order reduction was seen in the resulting spectral factor obtained using the row-shift truncation. In general, these errors can be reduced by assigning smaller values to the truncation parameters μ_{PU} and μ'_{PU} , but

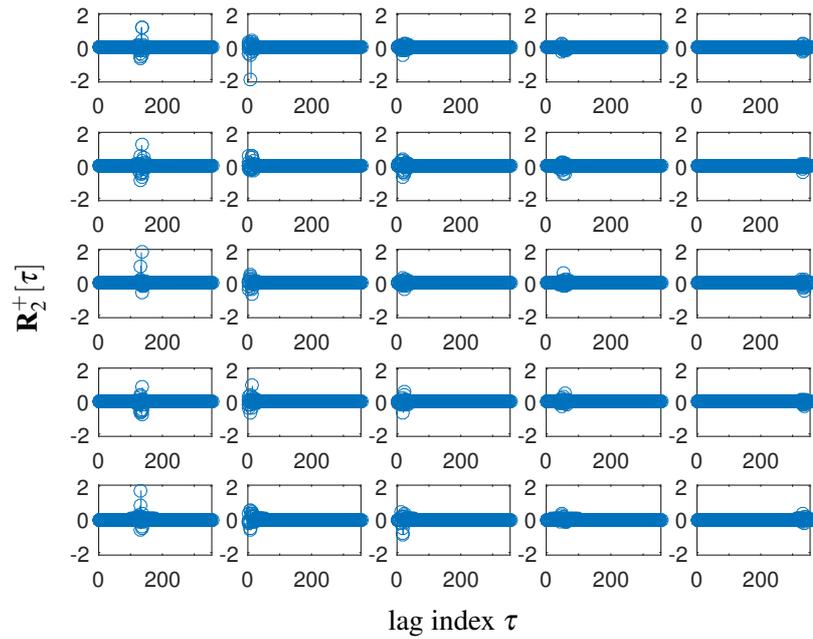


Figure 6.5 The *outer* spectral factor of the CSD matrix example in Figure 4.13, implementing the lag based truncation with $\mu_{\text{PU}} = 10^{-2}$.

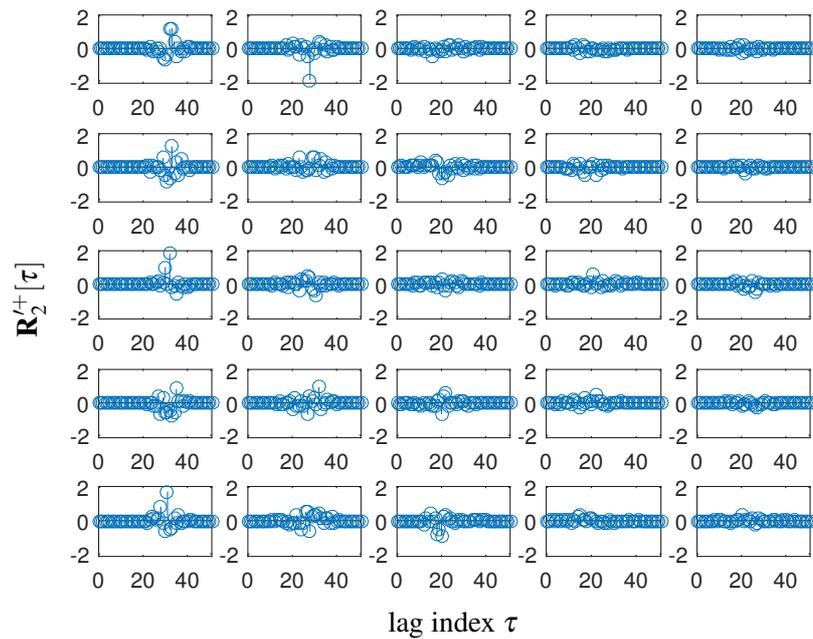


Figure 6.6 The *outer* spectral factor of the CSD matrix example in Figure 4.13, implementing the row-shift corrected truncation with $\mu'_{\text{PU}} = 10^{-2}$.

the order of the spectral factor will be increased as less energy is truncated. It is difficult to know in advance what value to choose for the truncation parameter in order to obtain the spectral factor with a specified order. It would not be sensible to apply

the lag bound fixed truncation to the spectral factor, either, as the spectral factor is not necessarily symmetric. Empirically, however, we can try a set of different truncation parameters to determine whether there exists one case for which the accuracy and the order of the spectral factor could be balanced.

6.5 Chapter Summary

In conclusion, the proposed spectral factorisation method based on the iterative PEVD algorithm provides an alternative way of solving multichannel spectral factorisation problems, and it is seen to offer a significant advantage in that the multichannel spectral factorisation problem is reduced to a number of independent single-channel problems for which suitable algorithms already exist. Although the definition of the unique spectral factor provided in some of the literature does not apply to our situation due to the fact that the polynomial order increases with increasing number of iterations, the validity of the spectral factor found by our method has been proven by means of worked examples. In addition, the order of the spectral factor can be kept as low as possible by exploiting the ambiguity of matrix spectral factorisation, which can potentially help to reduce the cost in filter bank based applications.

Chapter 7

Decoupling of Broadband MIMO Systems using the Polynomial Singular Value Decomposition by Polynomial Eigenvalue Decomposition Method

7.1 Introduction

This chapter demonstrates a potential application of polynomial singular value decomposition (PSVD) to design the precoding and equalisation for broadband multiple input multiple output (MIMO) communications. The PSVD, which can be seen as an extension of the singular value decomposition (SVD) for polynomial matrices, plays a very important role in broadband MIMO systems. One of its applications lies in the decoupling of a convolutive MIMO channel into a set of independent single input single output (SISO) channels, where standard equalisation techniques, such as zero-forcing equalisation, can then be applied to each SISO channel to obtain an estimate of one of the source signals.

The main contributions of this chapter are the exploitation of the proposed PSVD by MS-SBR2 method in terms of solving the broadband MIMO decoupling problem, comparisons against the PSVD by SBR2 method, and discussions of the accuracy of the PSVD method. Two worked examples are presented to demonstrate our proposed method, including a simulated MIMO channel and a measured 2×2 optical MIMO channel. In particular, a 2×2 optical MIMO channel comprising a 1.4 km multi-mode fibre (MMF) and optical couplers at both ends is designed to examine the bit error rate (BER) performance of an optical MIMO system in which the channel impulse responses are measured for the operating wavelength of 1576 nm [16]. Furthermore, different transmission and power allocation (PA) schemes are investigated to further optimise the BER performance.

7.2 The State of the Art

Explosive developments in MIMO technology have been witnessed in wireless communication systems over the last decade. Compared to SISO systems, MIMO systems are capable of achieving higher data rates and transmission reliabilities by using the techniques of spatial multiplexing and transmit diversity. Aiming to increase the fibre capacity, the concept of MIMO in optical transmission systems has also attracted intensive research interests [113, 114].

Due to the multipath effect in broadband MIMO systems, the channel is characterized by frequency-selective fading, so apart from the co-channel interference (CCI) caused by the MIMO components, there also exists inter-symbol interference (ISI) between the transmit symbols. Provided the approximate channel length is known to the transmitter, a standard approach of combating the ISI is to use multi-carrier modulation techniques, such as orthogonal frequency division multiplexing (OFDM) with cyclic prefix, which can divide the spectrum into a number of narrowband channels. In other words, the frequency-selective or broadband MIMO channel is turned into a

set of parallel frequency-flat or narrowband MIMO channels where the ISI no longer exists, and each narrowband channel can be independently addressed using existing narrowband optimal techniques. This type of MIMO-OFDM system was well implemented by combining spatio-temporal vector coding (STVC) [66, 115] with the SVD based equalisation technique [61]. A second alternative is based on the optimal filter bank transceiver techniques [116], which involve block processing and guard intervals to eliminate inter-block interference (IBI). These two traditional techniques are in common in the sense of eliminating the polynomial nature of the channel corresponding to the IBI due to the time-dispersive nature of the channel, and essentially they are all designed to convert the broadband problem into narrowband problems.

With the development of the PSVD techniques over the last decade, a PSVD based precoding and equalisation method [14, 16, 34, 117] has been employed in broadband MIMO systems. The advantage of the PSVD based method over other existing approaches lies in the natural ability of broadband decomposition algorithms which preserve and exploit the coherence of signals. This method generally consists of two steps. The first step is to use the PSVD to decouple the frequency-selective MIMO channel into a number of independent frequency-selective SISO channels so that the CCI can be removed. The second step involves removing the remaining ISI for each SISO channel, where it can be solved by standard equalisation techniques, such as zero-forcing (ZF) equalisation, maximum-likelihood sequence estimation, or decision-feedback equalisation. The following sections present the idea of this method and demonstrate how the proposed MS-SBR2 algorithm performs in terms of formulating the PSVD.

7.3 Decoupling of MIMO Channel using PSVD

Given a frequency selective MIMO link with n_T inputs and n_R outputs, the convolutive mixing channel can be modelled as a polynomial matrix

$$\underline{\mathbf{C}}(z) = \sum_{\tau=0}^T \mathbf{C}[\tau]z^{-\tau} = \begin{bmatrix} \underline{c}_{11}(z) & \cdots & \underline{c}_{1n_T}(z) \\ \vdots & \ddots & \vdots \\ \underline{c}_{n_R1}(z) & \cdots & \underline{c}_{n_Rn_T}(z) \end{bmatrix}, \quad (7.1)$$

where $\tau, T \in \mathbb{Z}$, and $\mathbf{C}[\tau] \in \mathbb{C}^{n_R \times n_T}$ denotes the polynomial coefficient matrix at time lag τ . $\underline{c}_{v\mu}(z)$ for $v = 1, \dots, n_R$ and $\mu = 1, \dots, n_T$ are the polynomial matrix entries which represent the channel impulse responses between the μ -th input and the v -th output, i.e.

$$\underline{c}_{v\mu}(z) = \sum_{\tau=0}^T c_{v\mu}[\tau]z^{-\tau}, \quad (7.2)$$

where $c_{v\mu}[\tau]$ denotes a non-zero element of the symbol rate sampled channel impulse response at the τ -th lag. In this case, there are $T + 1$ time lags in total for each SISO channel.

Assuming the transmitted signals are denoted as $\underline{\mathbf{s}}'(z) \in \mathbb{C}^{n_T \times 1}$, they are then emitted through the convolutive mixing channel $\underline{\mathbf{C}}(z)$, which results in the received signals $\underline{\mathbf{x}}'(z) \in \mathbb{C}^{n_R \times 1}$ represented by

$$\underline{\mathbf{x}}'(z) = \underline{\mathbf{C}}(z)\underline{\mathbf{s}}'(z) + \underline{\mathbf{n}}(z), \quad (7.3)$$

where $\underline{\mathbf{n}}(z) \in \mathbb{C}^{n_R \times 1}$ denotes a multivariate additive Gaussian noise process with covariance of $\sigma^2 \mathbf{I}_{n_R}$. Note that for this application, the signals $\underline{\mathbf{s}}'(z)$ represents the filtered source signals which are different from the source signals $\underline{\mathbf{s}}(z)$ shown in the convolutive model in (2.23).

If the channel matrix in (7.1) is known, PSVD can be used to convert the MIMO channel equalisation problem into a set of independent SISO problems. In other words,

the CCI can be removed by performing PSVD on the channel matrix $\underline{\mathbf{C}}(z)$ [81], i.e.,

$$\underline{\mathbf{C}}(z) = \tilde{\underline{\mathbf{U}}}(z)\underline{\Sigma}(z)\underline{\mathbf{V}}(z) = \tilde{\underline{\mathbf{U}}}(z) \begin{bmatrix} \underline{\Gamma}(z) \\ \mathbf{0} \end{bmatrix} \underline{\mathbf{V}}(z), \quad (7.4)$$

where we assume that there are at least as many sensors as sources, i.e., $n_R \geq n_T$. $\underline{\Gamma}(z)$ is a diagonal matrix with $n = n_T$ diagonal elements, i.e.,

$$\underline{\Gamma}(z) = \text{diag} \left\{ \underline{\gamma}_{11}(z), \underline{\gamma}_{22}(z), \dots, \underline{\gamma}_{nn}(z) \right\}. \quad (7.5)$$

Both $\tilde{\underline{\mathbf{U}}}(z) \in \mathbb{C}^{n_R \times n_R}$ and $\underline{\mathbf{V}}(z) \in \mathbb{C}^{n_T \times n_T}$ are paraunitary matrices, which satisfy

$$\begin{aligned} \tilde{\underline{\mathbf{U}}}(z)\underline{\mathbf{U}}(z) &= \underline{\mathbf{U}}(z)\tilde{\underline{\mathbf{U}}}(z) = \mathbf{I}_{n_R} \\ \tilde{\underline{\mathbf{V}}}(z)\underline{\mathbf{V}}(z) &= \underline{\mathbf{V}}(z)\tilde{\underline{\mathbf{V}}}(z) = \mathbf{I}_{n_T}. \end{aligned} \quad (7.6)$$

For this reason, $\tilde{\underline{\mathbf{U}}}(z)$ and $\underline{\mathbf{V}}(z)$ can be seen as the multichannel all-pass filters which preserve the total power of the signals at every frequency [69].

There are different ways of calculating the PSVD in (7.4), such as using the PQRD to formulate the PSVD [84], the PSVD based on generalised Kogbetliantz transformations [118], and the PSVD by PEVD method [81], which is analogous to how the EVD can be used to generate the SVD of a matrix. In terms of the PSVD by PEVD method, the SBR2 algorithm [7] has been adopted in all existing literatures [10, 14, 16, 34, 117]. In this thesis, we use the proposed MS-SBR2 algorithm to formulate the PSVD in order to see how it performs in comparison to the PSVD by SBR2 algorithm.

To formulate the PSVD using the PEVD, both the paraunitary matrices $\underline{\mathbf{U}}(z)$ and $\tilde{\underline{\mathbf{V}}}(z)$ need to be found. This is implemented by calculating the PEVD of two para-Hermitian matrices $\underline{\mathbf{C}}(z)\tilde{\underline{\mathbf{C}}}(z)$ and $\tilde{\underline{\mathbf{C}}}(z)\underline{\mathbf{C}}(z)$, which take the form of

$$[\underline{\mathbf{C}}(z)\tilde{\underline{\mathbf{C}}}(z)]_{n_R \times n_R} = \tilde{\underline{\mathbf{U}}}(z)\underline{\Sigma}(z)\tilde{\underline{\Sigma}}(z)\underline{\mathbf{U}}(z), \quad (7.7)$$

and

$$[\tilde{\mathbf{C}}(z)\mathbf{C}(z)]_{n_T \times n_T} = \tilde{\mathbf{V}}(z)\tilde{\mathbf{\Sigma}}(z)\mathbf{\Sigma}(z)\mathbf{V}(z). \quad (7.8)$$

After obtaining the paraunitary matrices $\mathbf{U}(z)$ and $\tilde{\mathbf{V}}(z)$, the source signals $\mathbf{s}(z) \in \mathbb{C}^{n_T \times 1}$, which are generally drawn from a finite constellation such as binary phase shift keying (BPSK) or quadrature amplitude modulation (QAM), are passed through a transmit filter bank represented by the paraunitary matrix $\tilde{\mathbf{V}}(z)$. Therefore, the transmitted signals in (7.3) can be expressed as

$$\mathbf{s}'(z) = \tilde{\mathbf{V}}(z)\mathbf{s}(z), \quad (7.9)$$

and the received signals $\mathbf{x}'(z)$ in (7.3) are then filtered by the paraunitary matrix $\mathbf{U}(z)$ at the receiver, i.e., $\mathbf{x}(z) = \mathbf{U}(z)\mathbf{x}'(z)$. By combining the pre- and post-processing, the filtered received signals can be expressed as

$$\begin{aligned} \mathbf{x}(z) &= \mathbf{U}(z)\mathbf{C}(z)\tilde{\mathbf{V}}(z)\mathbf{s}(z) + \mathbf{U}(z)\mathbf{n}(z) \\ &= \mathbf{\Sigma}(z)\mathbf{s}(z) + \mathbf{n}'(z), \end{aligned} \quad (7.10)$$

where $\mathbf{n}'(z) = \mathbf{U}(z)\mathbf{n}(z)$. Furthermore, as both the transmit filter bank $\tilde{\mathbf{V}}(z)$ and the receive filter bank $\mathbf{U}(z)$ are paraunitary, the transmit signal power is not increased, nor is the channel noise enhanced throughout this process. A block diagram of the proposed communication system is depicted in Figure 7.1.

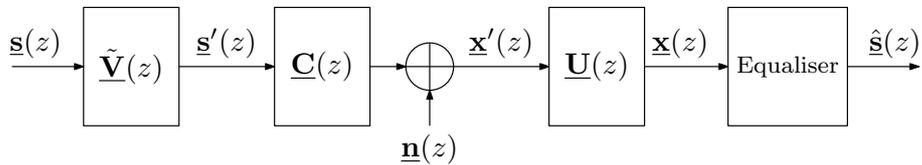


Figure 7.1 A block diagram of the proposed MIMO communication system using the PSVD based equalisation scheme.

Unlike the conventional SVD based method, each diagonal element (also called layer) $\underline{\sigma}_\ell(z)$ in $\mathbf{\Sigma}(z)$ represents a frequency-selective SISO channel and hence, ISI oc-

curs. In order to remove the ISI, the layer-specific ZF equalisation scheme [16] is employed with its block diagram as depicted in Figure 7.2. The ZF equaliser $\underline{f}_\ell(z) =$

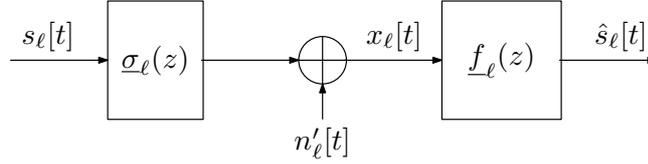


Figure 7.2 A block diagram of the layer-specific ZF equalisation for each SISO channel obtained using the PSVD.

$\sum_t f_\ell[t]z^{-t}$ of the ℓ -th layer is chosen as an FIR filter which aims to neutralise the frequency response imposed by the SISO channel transfer function $\underline{\sigma}_\ell(z) = \sum_t \sigma_\ell[t]z^{-t}$, such that

$$\begin{aligned} \sigma_\ell[t] \otimes f_\ell[t] &= h_\ell[t] \\ &= [0, \dots, 0, 1, 0, \dots, 0], \quad \ell = 1, \dots, L, \end{aligned} \quad (7.11)$$

where $L = \min\{n_T, n_R\}$. Note that there is no restriction imposed on the position of the coefficient ‘1’ in $h_\ell[t]$. Thus, the equivalent ISI free channel model can be depicted in Figure 7.3, and the equalised signal is given by

$$\hat{s}_\ell[t] = s_\ell[t] + n'_\ell[t] \otimes f_\ell[t], \quad \ell = 1, \dots, L, \quad (7.12)$$

Note that the power of the noise $n'_\ell[t]$ is now weighted by the ZF equaliser.

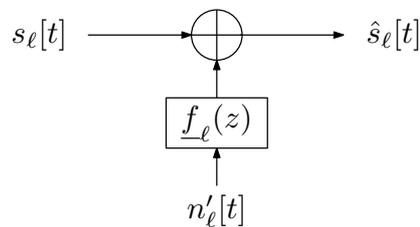


Figure 7.3 The equivalent SISO channel model with ISI removed using the layer-specific ZF equalisation.

7.3.1 Accuracy of the Decomposition

There are several factors which can affect the accuracy of the decomposition. Firstly, because the decomposition is performed upon the two para-Hermitian matrices $\underline{\mathbf{C}}(z)\tilde{\underline{\mathbf{C}}}(z)$ and $\tilde{\underline{\mathbf{C}}}(z)\underline{\mathbf{C}}(z)$ as shown in (7.7) and (7.8), the resulting diagonal matrix $\underline{\Sigma}(z)$ might be less accurate than that found by applying the decomposition directly to the channel matrix $\underline{\mathbf{C}}(z)$, such as the PSVD using generalised Kogbetliantz transformations [118]. However, the investigation of this method is beyond the scope of this thesis.

Secondly, in the sense of broadband MIMO decoupling, a strictly diagonalised channel matrix is required. However, the proposed PSVD method can only generate an approximately diagonal matrix subject to the pre-specified stop condition of the algorithm, so there will be errors when assuming all the off-diagonal elements of $\underline{\Sigma}(z)$ are equal to zero.

Furthermore, because the order of the polynomial matrices within the algorithm increases as the iterations increase, the equalisation for each SISO channel becomes very difficult when the resulting order of $\underline{\Sigma}(z)$ is too large. Therefore, polynomial order truncation is very necessary in order to keep the polynomial order as low as possible and reduce the computational cost of the algorithm. However, truncating polynomial matrices can also cause a very small proportion of the total Frobenius norm of the polynomial matrix to be removed, which affects the accuracy of the decomposition. To assess how well the proposed PSVD method performs, the error metric of the PSVD is defined as

$$\mathcal{E} = \frac{\|\underline{\mathbf{C}}(z) - \hat{\underline{\mathbf{C}}}(z)\|_{\text{F}}^2}{\|\underline{\mathbf{C}}(z)\|_{\text{F}}^2} = \frac{\|\underline{\mathbf{C}}(z) - \tilde{\underline{\mathbf{U}}}(z)\hat{\underline{\Sigma}}(z)\underline{\mathbf{V}}(z)\|_{\text{F}}^2}{\|\underline{\mathbf{C}}(z)\|_{\text{F}}^2}, \quad (7.13)$$

where $\hat{\underline{\mathbf{C}}}(z)$ denotes the reconstructed matrix obtained from calculating the inverse PSVD. $\hat{\underline{\Sigma}}(z)$ is equal to $\underline{\Sigma}(z)$, but with all the off-diagonal elements set to zero. Note that the polynomial coefficients of $\hat{\underline{\mathbf{C}}}(z)$ must be properly aligned with the coefficients of $\underline{\mathbf{C}}(z)$ when calculating this error metric. As the polynomial matrix $\hat{\underline{\mathbf{C}}}(z)$ is usually

not para-Hermitian, it is more difficult to obtain an accurate value of \mathcal{E} compared to the way of calculating the para-Hermitian reconstruction error mentioned in Section 4.20.

7.3.2 Transmission Quality

The quality of the data transmission is in general assessed using the signal-to-noise (SNR) ratio at the detector's input, which can be defined as [16]

$$\rho = \frac{(\text{half vertical eye opening})^2}{\text{noise power}} = \frac{U_A^2}{P_R}, \quad (7.14)$$

where U_A and P_R correspond to one quadrature component in M -ary QAM. Let $\rho^{(\ell)}$ represent the SNR of the ℓ -th layer of a MIMO system; the bit error probability for the constellation size of M_ℓ can then be expressed as [119]

$$P_{\text{BER}}^{(\ell)} = \frac{2}{\log_2(M_\ell)} \left(1 - \frac{1}{\sqrt{M_\ell}} \right) \text{erfc} \left(\sqrt{\frac{\rho^{(\ell)}}{2}} \right). \quad (7.15)$$

Considering all the transformed SISO channels (or layers) obtained from the PSVD with ZF equalisation, the average BER is calculated as

$$P_{\text{BER}} = \frac{1}{\sum_{\ell=1}^L \log_2(M_\ell)} \sum_{\ell=1}^L \log_2(M_\ell) P_{\text{BER}}^{(\ell)}, \quad (7.16)$$

For QAM constellations, the average symbol power per layer is given by [120, 121]

$$P_{s,\ell} = \frac{2}{3} U_{s,\ell}^2 (M_\ell - 1), \quad (7.17)$$

where $U_{s,\ell}$ denotes the half amplitude of the transmitted symbol, and because the ISI is removed by the ZF equaliser, $U_{s,\ell}$ is equal to the half vertical eye opening of the received symbol, i.e., $U_{s,\ell} = U_{A,\ell}$. Assuming that total transmit power P_s is uniformly allocated to all MIMO layers, the transmit power of each SISO channel is given by $P_{s,\ell} = P_s/L$. Hence, by rearranging equation (7.17), the half amplitude of the transmit-

ted symbol per layer is given by

$$U_{s,\ell} = \sqrt{\frac{3P_s}{2L(M_\ell - 1)}}. \quad (7.18)$$

As the noise is affected by the ZF equaliser as demonstrated in Figure 7.3, the noise power P_R will be weighted by the equaliser coefficients such that

$$P_{R,\ell} = \theta_\ell P_R \quad \text{with} \quad \theta_\ell = \sum_{\forall t} |f_\ell[t]|. \quad (7.19)$$

Thus, using (7.18) and (7.19), the SNR $\rho^{(\ell)}$ can be calculated as

$$\rho^{(\ell)} = \frac{U_{A,\ell}^2}{P_{R,\ell}} = \frac{U_{s,\ell}^2}{\theta_\ell P_R} = \frac{3P_s}{2L(M_\ell - 1)\theta_\ell P_R} = \frac{3}{\theta_\ell L(M_\ell - 1)} \frac{E_s}{N_0} \quad (7.20)$$

with $P_s/P_R = 2E_s/N_0$, where E_s and N_0 denote the average symbol energy and the power spectral density of white Gaussian noise.

7.4 Numerical Examples

Two different channel matrix examples are used to demonstrate the proposed MS-SBR2 algorithm for calculating the PSVD. The first example was generated based on a 3×4 MIMO propagation channel, and the second one was obtained by measuring a 2×2 optical MIMO channel impulse response, as mentioned earlier.

7.4.1 Example 1

For this example, the channel matrix $\underline{\mathbf{C}}_1(z) \in \mathbb{C}^{4 \times 3}$ was generated to model the propagation of three source signals onto four sensors. Each of the polynomial entries in $\underline{\mathbf{C}}_1(z)$ was chosen to be a fourth order FIR filter, where both the real and imaginary parts of the polynomial coefficients $c_{mn}[\tau], \forall \tau, m = 1, \dots, 4$ and $n = 1, \dots, 3$ are

drawn randomly from a uniform distribution in the range $[-1, 1]$. Figure 7.4 shows the magnitudes of all the coefficients in this polynomial matrix.

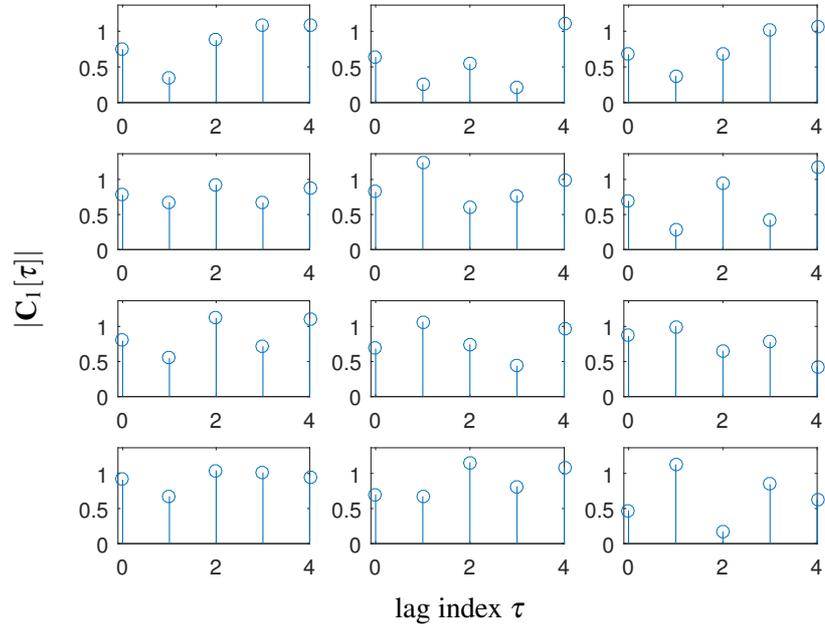


Figure 7.4 The stem plot of the 4×3 broadband MIMO channel matrix $\underline{\mathbf{C}}_1(z)$, showing the magnitudes of the channel impulse responses at different time lags.

To compute the PSVD of $\underline{\mathbf{C}}_1(z)$, the MS-SBR2 algorithm was respectively applied to the polynomial matrices $\underline{\mathbf{C}}_1(z)\tilde{\underline{\mathbf{C}}}_1(z)$ and $\tilde{\underline{\mathbf{C}}}_1(z)\underline{\mathbf{C}}_1(z)$ in turn to obtain the paraunitary matrices $\underline{\mathbf{U}}_1(z)$ and $\tilde{\underline{\mathbf{V}}}_1(z)$. The stopping condition for each implementation of the MS-SBR2 algorithm was set as $\varepsilon = 10^{-3}$, and the two truncation functions mentioned in Sections 3.6.2 and 3.6.3 were employed as part of the iterative routine in MS-SBR2 with $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-4}$, which allowed at most, this proportion of the squared Frobenius norm of the matrix to be lost after each iteration. Using the truncation functions prevents the order of the polynomial matrices from growing unnecessarily large and therefore reduces the computational time of the algorithm. The MS-SBR2 algorithm stopped when the magnitude of each off-diagonal coefficient of the resulting diagonalised matrices $\underline{\Sigma}(z)\tilde{\underline{\Sigma}}(z)$ and $\tilde{\underline{\Sigma}}(z)\underline{\Sigma}(z)$ was found to be smaller than 10^{-3} , which resulted in a total of 478 iterations over both applications of the MS-SBR2 algorithm. The resulting paraunitary matrices $\underline{\mathbf{U}}_1(z)$ and $\underline{\mathbf{V}}_1(z)$ are illustrated in Figures 7.5 and

7.6, respectively. Accordingly, the approximately diagonalised matrix $\underline{\Sigma}_1(z)$ can then be obtained according to (7.4), which is shown in Figure 7.7. with order of 71.

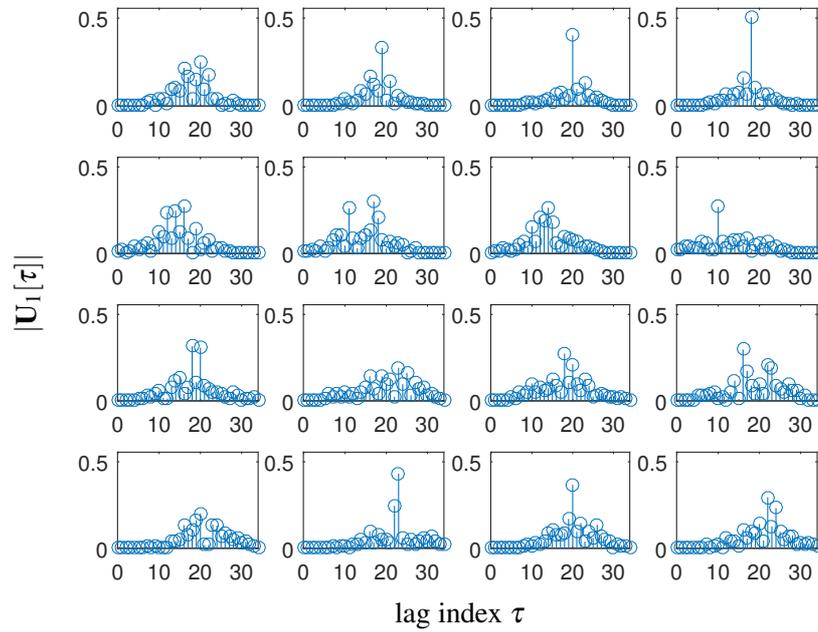


Figure 7.5 The stem plot of the paraunitary matrix $\underline{U}_1(z)$ obtained from the PSVD by MS-SBR2 method, showing the magnitudes of the coefficients.

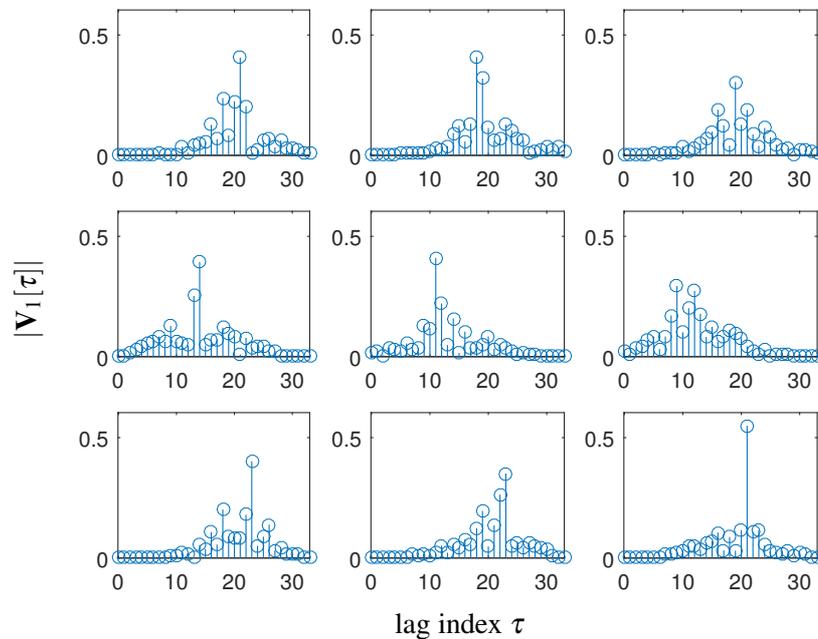


Figure 7.6 The stem plot of the paraunitary matrix $\underline{V}_1(z)$ obtained from the PSVD by MS-SBR2 method, showing the magnitudes of the coefficients.

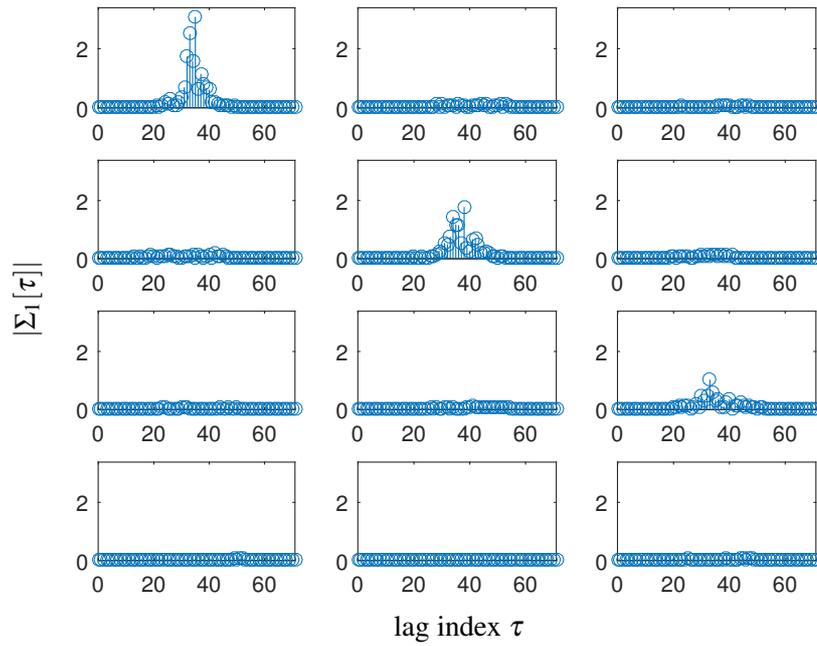


Figure 7.7 The stem plot of the diagonalised MIMO channel matrix $\underline{\Sigma}_1(z)$ obtained from the PSVD by MS-SBR2 method, showing the magnitudes of the coefficients.

The magnitude of the maximum off-diagonal element, found in the approximately diagonalised matrix $\underline{\Sigma}_1(z)$, was given by $|\sigma_{jk}[\tau]|_{\max} = 0.1939$, and the off-diagonal energy $\|\text{off}\{\underline{\Sigma}_1(z)\}\|_{\text{F}}^2$ was calculated as 1.2248, accounting for 3.03% of the total squared Frobenius norm of the matrix. Certainly, this can be reduced by setting a tighter convergence bound, but the order of the polynomial matrices will accordingly increase. Due to the truncation process, some small coefficients were removed, which resulted in the squared Frobenius norm of $\underline{\Sigma}_1(z)$ decreasing from 40.6462 to 40.4716. Furthermore, the relative error \mathcal{E}_1 was calculated as 0.0377 using (7.13), which shows that the proposed PSVD method has achieved a good level of accuracy. All the performance measures obtained from applying the PSVD by MS-SBR2 method to $\underline{\mathbf{C}}_1(z)$ are shown in Table 7.1.

To compare our proposed method with the PSVD by SBR2 method, we chose the same truncation and stopping parameters for each implementation of the SBR2 algorithm as when calculating the PSVD of $\underline{\mathbf{C}}_1(z)$. The results obtained from the PSVD by SBR2 method are also presented in Table 7.1. Upon comparison of these results,

it can be seen that the PSVD by MS-SBR2 method took less computational time than did using the SBR2 algorithm, whilst achieving a better diagonalisation measure. In addition, the relative error of the decomposition obtained using the MS-SBR2 method is slightly less than that found by the SBR2 algorithm based PSVD. Note that as the channel matrix $\underline{\mathbf{C}}_1(z)$ is 4×3 , the advantage of using the MS-SBR2 algorithm lies in the multiple shift effect when calculating the PEVD of $[\underline{\mathbf{C}}_1(z)\tilde{\underline{\mathbf{C}}}_1(z)]_{4 \times 4}$. As to the polynomial matrix $[\tilde{\underline{\mathbf{C}}}_1(z)\underline{\mathbf{C}}_1(z)]_{3 \times 3}$, the MS-SBR2 algorithm is simply equivalent to SBR2. It is self-evident that for a large MIMO channel matrix, the PSVD by MS-SBR2 method can potentially provide better performance than SBR2 in terms of the computational speed and the diagonalisation measure.

Table 7.1 Results obtained from applying the SBR2 and MS-SBR2 algorithms for calculating the PSVD to the polynomial matrix $\underline{\mathbf{C}}_1(z)$ in Figure 7.4, with the truncation parameters set as $\mu_{\text{PH}} = \mu_{\text{PU}} = 10^{-4}$ and the stopping condition as $\varepsilon = 10^{-3}$ in both methods.

Measures	PSVD by	
	SBR2	MS-SBR2
number of iterations	561	478
$ \sigma_{jk}[\tau] _{\max}$	0.2271	0.1939
$\ \underline{\mathbf{C}}_1(z)\ _{\text{F}}^2$	40.6462	40.6462
$\ \underline{\Sigma}_1(z)\ _{\text{F}}^2$	40.3253	40.4716
$\ \text{off}\{\underline{\Sigma}_1(z)\}\ _{\text{F}}^2$	1.3948	1.2248
order of $\underline{\Sigma}_1(z)$	64	71
order of $\underline{\mathbf{U}}_1(z)$	27	34
order of $\underline{\mathbf{V}}_1(z)$	33	33
relative error \mathcal{E}_1	0.0428	0.0377
computational time (sec.) ¹	1.98	1.74

¹ Computations undertaken on a PC with Intel(R) Core(TM) i7-3770T CPU @ 2.50GHz and 16 GB RAM.

Spectral Ordering

Akin to an ordered SVD with the singular values in descending order, the power spectral densities of the on-diagonal polynomials in the diagonalised matrix $\underline{\Sigma}(z)$, obtained

from the PSVD, satisfy the spectral majorisation property [27], such that

$$\underline{\sigma}_{11}(e^{j\Omega}) \geq \underline{\sigma}_{22}(e^{j\Omega}) \geq \dots \geq \underline{\sigma}_{MM}(e^{j\Omega}), \quad \forall \Omega \in [-\pi, \pi). \quad (7.21)$$

where $\underline{\sigma}_{mm}(e^{j\Omega}) = \underline{\sigma}_{mm}(z)|_{z=e^{j\Omega}}$ for $m = 1, \dots, M$. This property follows directly from the proof of the spectral majorisation for SBR2, as described in Section 3.4.2.

For this example, Figures 7.8(a) and (b) illustrate the power spectra of the channel matrix $\underline{\mathbf{C}}_1(z)$ and the ordered power spectra in the diagonalised matrix $\underline{\Sigma}_1(z)$, respectively. This intuitively indicates that the broadband MIMO channel has been decoupled into a set of independent SISO channels, with no CCI.

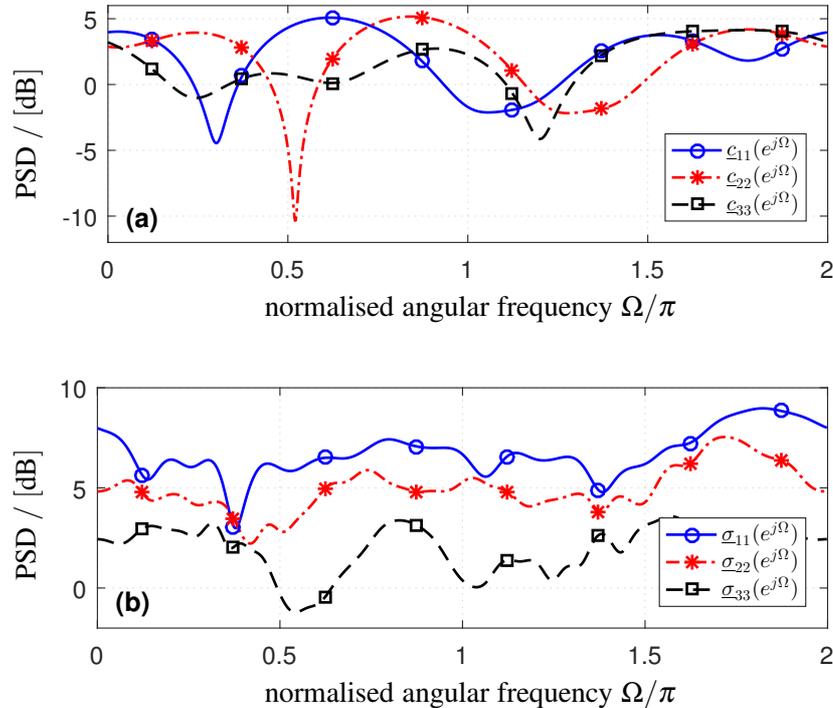


Figure 7.8 Power spectral densities of the on-diagonal polynomials in the 4×3 MIMO channel matrix, showing (a) before diagonalisation and (b) after diagonalisation using the PSVD by MS-SBR2.

7.4.2 Example 2

In this example, we consider a more realistic scenario, where a measured 2×2 optical MIMO channel is set up to demonstrate the proposed method, and the BER perfor-

mance of the system is examined based on different modulation and power allocation schemes.

Optical MIMO Channel Set-Up

In fibre optic communications, one way to implement a MIMO transmission is to carry the data streams using different optical modes through an MMF [114, 122]. For the excitation of different modes, certain single mode fibre (SMF) to MMF alignments with varying radial offsets δ are used in this experiment. The spatial diversity of the optical MIMO channel is visualised by showing the measured spatial intensity distributions when exciting the two optical MIMO inputs of the 2×2 system separately. The measured patterns depicted in Figure 7.9 show that spatially diverse channels are generated. However, an ideal separation of the two channels is hard to achieve as mode mixing usually occurs in the mode multiplexing and demultiplexing process which is implemented by fusion couplers, and during the transmission through the fibre. In optical MIMO systems, it should be noticed that the group delays in an MMF optical channel belong to a fixed set of values in contrast to wireless channels, which can change from one realisation to another [123].

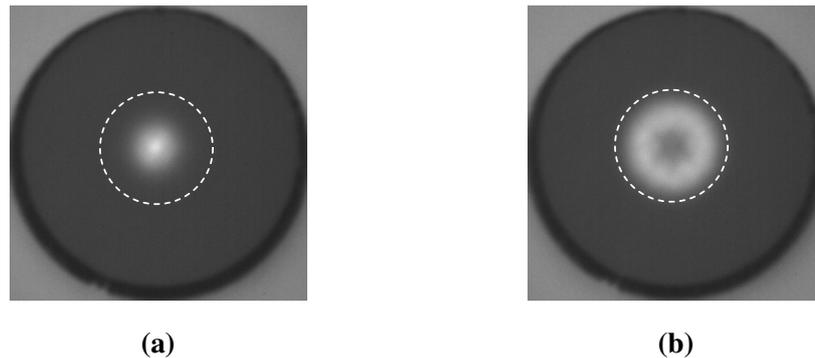


Figure 7.9 Intensity distribution patterns of an MMF when launching the light with the radial offsets **(a)** $\delta = 0 \mu\text{m}$ (centric) and **(b)** $\delta = 15 \mu\text{m}$ (eccentric), where the dashed circle has a diameter of $50 \mu\text{m}$.

An overview of the testbed used for measuring the impulse responses of a 2×2 optical MIMO channel is shown in Figure 7.10. Here the optical channel is made up

7.4 Numerical Examples

of a 1.4 km MMF, fusion couplers, and differently aligned SMFs, and the impulse responses are measured at an operating wavelength of 1576 nm using the signal deconvolution method proposed by Sandmann and Ahrens in [16]. As the aim of this chapter is to demonstrate the ability of the PSVD by MS-SBR2 method to transform a broadband MIMO channel matrix into a set of SISO channels, the details of how to obtain the optical channel impulse responses is not the focus of this thesis.

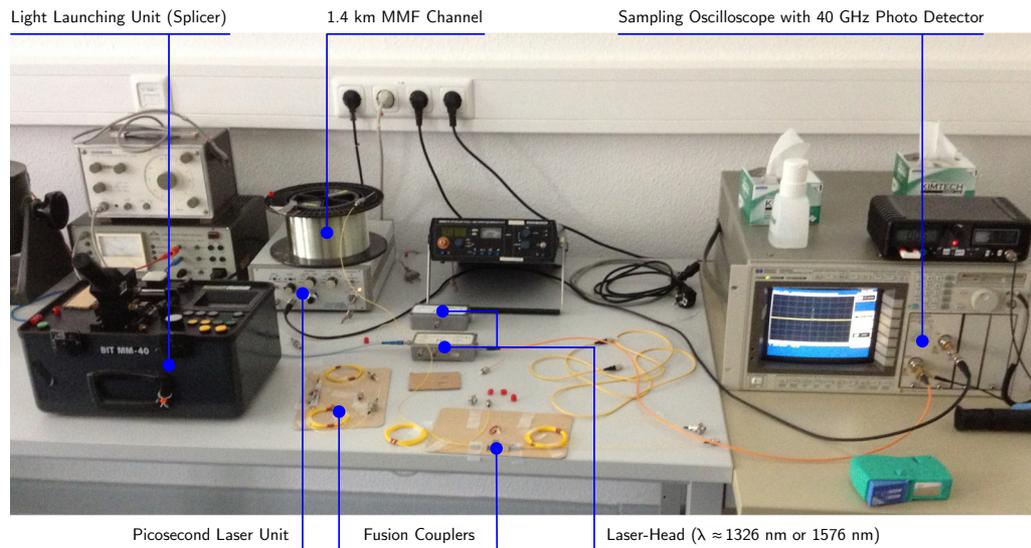


Figure 7.10 An overview of the testbed for measuring the impulse responses of a 2×2 optical MIMO channel [124, 32].

Decoupling of the MIMO Channel

The measured impulse responses are then sampled at a symbol rate of 620 MHz and used to constitute the channel matrix $\underline{\mathbf{C}}_2(z)$ as plotted in Figure 7.11. The PSVD by MS-SBR2 algorithm is then applied to the two para-Hermitian matrices $\underline{\mathbf{C}}_2(z)\tilde{\underline{\mathbf{C}}}_2(z)$ and $\tilde{\underline{\mathbf{C}}}_2(z)\underline{\mathbf{C}}_2(z)$ to obtain the paraunitary matrices $\underline{\mathbf{U}}_2(z)$ and $\underline{\mathbf{V}}_2(z)$. Note that each implementation of the MS-SBR2 algorithm stops when the magnitude of the maximum off-diagonal coefficient found in the matrix is less than $\varepsilon = 10^{-4}$, and the truncation with $\mu_{\text{PH}} = 10^{-4}$ is only applied once after the algorithm converges.

It took 24 iterations in total to perform the two PEVDs using MS-SBR2, and the resulting paraunitary matrices $\underline{\mathbf{U}}_2(z)$ and $\underline{\mathbf{V}}_2(z)$ are plotted in Figures 7.12(a) and 7.12(b),

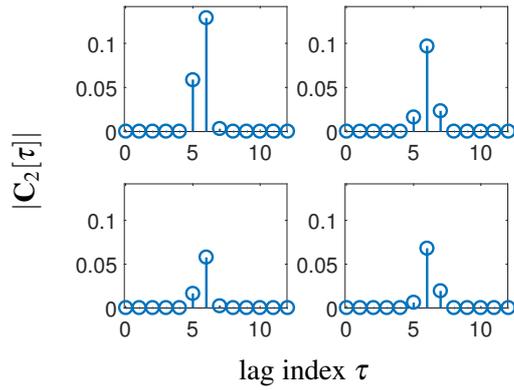


Figure 7.11 The stem plot of the measured 2×2 optical MIMO channel matrix $\underline{\mathbf{C}}_2(z)$, showing the magnitudes of the channel impulse responses.

respectively. Accordingly, the approximately diagonalised matrix $\underline{\Sigma}_2(z)$, obtained from applying the two paraunitary matrices according to (7.4), is shown in Figure 7.13. Furthermore, the magnitude of the maximum off-diagonal coefficient in $\underline{\Sigma}_2(z)$ was found to be 5.1359×10^{-4} , which indicates a decent diagonalisation measure. Finally, the relative error of this decomposition is calculated as $\mathcal{E}_2 = 4.7018 \times 10^{-5}$ according to (7.13). As the size of the channel matrix $\underline{\mathbf{C}}_2(z)$ is only 2×2 , the PSVD by MS-SBR2 algorithm performs the same as using SBR2. Nonetheless, the main goal of this example is to test our proposed method in a more realistic scenario and assess the optical MIMO system in terms of the transmission quality, as discussed in Section 7.3.2.

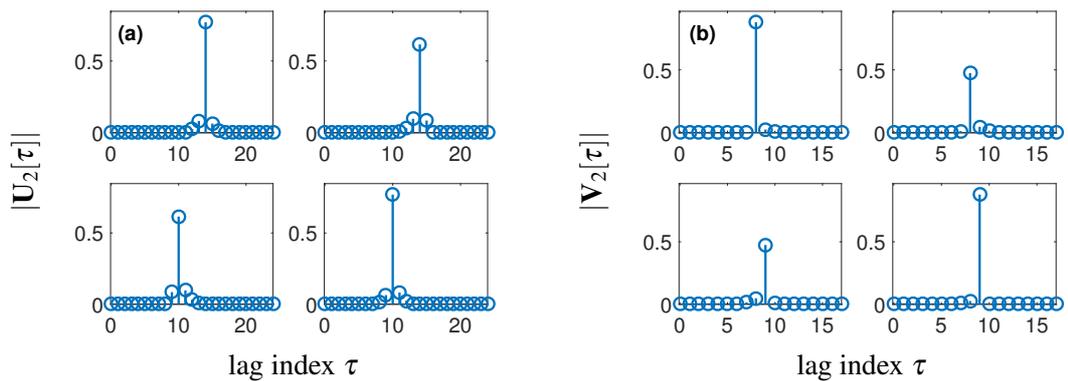


Figure 7.12 The resulting paraunitary matrices obtained from the MS-SBR2 algorithm, showing (a) the magnitudes of the coefficients in $\underline{\mathbf{U}}_2(z)$ and (b) the magnitudes of the coefficients in $\underline{\mathbf{V}}_2(z)$.

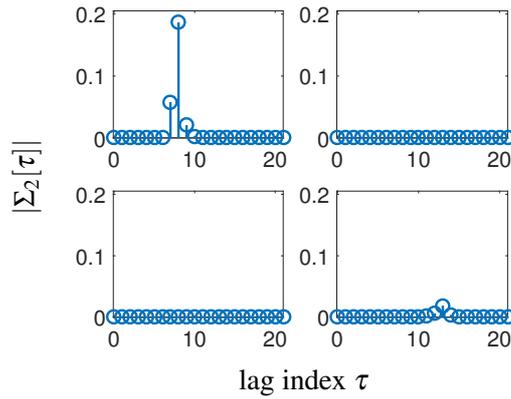


Figure 7.13 The stem plot of the diagonalised channel matrix $\Sigma_2(z)$ obtained from applying the PSVD by MS-SBR2 algorithm to the example in Figure 7.11.

Applying the PSVD to the channel matrix $\mathbf{C}_2(z)$ results in two independent SISO channels with time-dispersive characteristics and hence ISI occurs on each SISO channel (or layer) as shown in the diagonalised matrix $\Sigma_2(z)$. The ISI is removed by applying the ZF equalisation as described in Section 7.3. The equaliser modifies the noise power as shown in (7.19), which affects the layer-specific SNR $\rho^{(\ell)}$ and hence the final average BER performance P_{BER} . In this example, the length of the equalisation filter is chosen to be 20. However, it should be noticed that a complete elimination of ISI is in general not possible because of the finite filter length. The noise weighting factors for each layer are then computed as $\theta_1 = 37.22$ and $\theta_2 = 4243.46$ according to (7.19). Clearly, the noise power at both layers has been enhanced, and in particular, the noise power of the higher channel ($\ell = 2$) is weighted much more than that of the lower channel ($\ell = 1$). If the same QAM constellations are adopted for both layers, the SNR found at the second layer, $\rho^{(2)}$, would be much less than that found at $\rho^{(1)}$.

The BER performance is studied by means of a chosen set of transmission modes with a fixed spectral efficiency of 8 bit/s/Hz, and the QAM constellation arrangements are depicted in Table 7.2. Note that the overall BER of the decoupled MIMO system is dominated by the layer with the largest BER. One way to optimise the overall BER performance is to equalise the SNR, $\rho^{(\ell)}, \forall \ell$, across all layers using the power allocation (PA) scheme proposed in [16].

Table 7.2 Investigated transmission modes of the 2×2 optical MIMO system.

Throughput	layer 1	layer 2
8 bit/s/Hz	256	0
8 bit/s/Hz	64	4
8 bit/s/Hz	16	16

The BER results, calculated for a chosen range of SNRs (E_s/N_0), are depicted in Figure 7.14. Note that no PA is needed for the (256,0) QAM transmission mode. As seen from the graph, the (256,0) QAM transmission scheme shows the best BER performance among all chosen schemes. Furthermore, when activating both layers, the benefit of using the PA method is clearly visible. As suggested by the results presented in [117], it is expected that for the achievable spectral efficiency, the PSVD based MIMO systems can offer the same BER performance as systems based on the equalisation method using STVC with SVD. The main advantage of using the PSVD based equalisation over the STVC based MIMO systems is that no block-wise transmission is needed.

7.5 Chapter Summary

In this chapter, we demonstrated a potential application of the proposed PSVD by MS-SBR2 algorithm. The algorithm can be used to transfer a broadband MIMO channel into a set of independent SISO channels, where the conventional equalisation methods, such as ZF equalisation, can then be applied to each SISO channel in order to estimate one of the source signals. Two worked examples are presented to show the ability of the proposed method. The first example is designed to compare the MS-SBR2 with the SBR2 algorithm when used to calculate the PSVD of a 4×3 polynomial channel matrix. Simulation results show that the proposed PSVD by MS-SBR2 algorithm is advantageous in terms of the convergence speed and the decomposition accuracy.

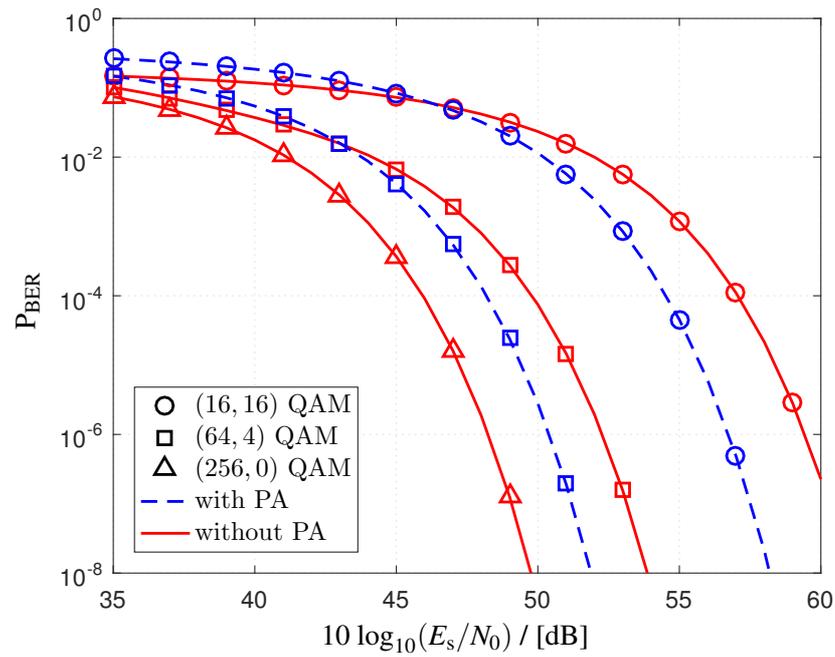


Figure 7.14 Results of BER obtained from applying the proposed PSVD based equalisation scheme to a measured 2×2 optical MIMO channel, showing the comparison between different QAM transmission modes with and without the power allocation scheme.

Furthermore, we tested the proposed algorithm using a measured 2×2 broadband optical MIMO channel matrix, and the BER performance was examined based on the chosen transmission schemes with a fixed spectral efficiency. Results have shown that the overall BER can be further reduced if the power allocation method is used. The activation of all transmission layers does not necessarily lead to the best BER performance. On the contrary, the $(256,0)$ QAM transmission scheme appears to have the best performance in the studied example.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

The first contribution of this thesis lies in the proof of the spectral majorisation property for the SBR2 algorithm. As the spectrally majorised signals tend to have most of the signal power compacted into the least number of signal channels [7, 35], this property plays a very important role in applications that require techniques of broadband subspace decomposition to identify the signal subspace, such as broadband BSS and beamforming. Subsequently, a modified SBR2 algorithm was designed to optimise the subband coding gain. Based on the monotonically increasing behaviour of the coding gain (as shown in Figure 3.4), this modified SBR2 algorithm provides a more reliable convergence test compared to the original SBR2 algorithm, whose convergence parameter is the maximum off-diagonal element $|r_{jk}^{(i)}[\tau]|$, which in general does not monotonically decrease as the iteration index i goes up.

Furthermore, an improved version of the SBR2 algorithm, called MS-SBR2, has been proposed to compute the PEVD of para-Hermitian matrices. The MS-SBR2 algorithm, which has been proven to converge, utilises a multiple-shift strategy that can transfer more off-diagonal elements onto the diagonal at each iteration than SBR2 can. Therefore, it can diagonalise a para-Hermitian matrix in fewer iterations. Subse-

quently, two different time-shift methods have been introduced for MS-SBR2. With the direct-fixed shift method, the off-diagonal elements can be kept closer to the zero-lag plane compared to using the conventional shift method, and all zero-filled outer coefficient matrices can be easily tracked and therefore removed without compromising the accuracy of the decomposition. Removing the zero-filled outer coefficient matrices brings the benefit of reducing the computational load of the MS-SBR2 algorithm, which makes the algorithm run more rapidly. This is illustrated by the numerical examples in Chapter 4. Due to the common limitation that the order of polynomial matrices will increase as the iterations increase in the PEVD algorithms, truncation methods have been used to reduce the order of the resulting polynomial matrices. Simulation results have shown that the order of polynomial matrices can be dramatically reduced while removing only a small proportion of the total energy of the matrices and still maintaining a decent level of accuracy. Truncating polynomial matrices is very helpful to some applications such as designing FIR paraunitary filter banks for subband coding and decoupling broadband MIMO systems, as the costs of these applications are directly proportional to the order of the resulting polynomial matrices.

The MS-SBR2 algorithm has been fully tested using various para-Hermitian matrix examples that are generated by differently conditioned source models. The algorithm has also been applied to decorrelate convolutively mixed signals. Results have shown that MS-SBR2 outperforms SBR2 in terms of the diagonalisation measure, the resulting polynomial order, and the computational speed, as detailed in Chapter 4. Note that the MS-SBR2 algorithm reduces to the SBR2 algorithm if the size of the input para-Hermitian matrix is less than 4×4 or if there exists only one paraunitary shift operation at each iteration.

Then, the MS-SBR2 algorithm is compared to other existing PEVD algorithms in terms of computational complexity and various performance metrics, as presented in Chapter 5. The results show that the fastest of these algorithms is the MS-SBR2

algorithm when considering the same level of diagonalisation, and the MSME-SMD algorithm performs the best in terms of reducing the off-diagonal energy.

Last but not least, two potential applications presented in Chapters 6 and 7 demonstrate the effectiveness of the MS-SBR2 algorithm. The first application demonstrates how the MS-SBR2 algorithm can be used to address the multichannel spectral factorisation problem, and the second application demonstrates the MS-SBR2 algorithm to formulate the PSVD, and therefore decoupling broadband MIMO channels.

8.2 Suggestions for Future Work

Suggestions for future work can be summarised in three categories.

Algorithms

By using the multiple-shift strategy, the MS-SBR2 algorithm has been demonstrated to be advantageous over the conventional SBR2 algorithm in terms of the convergence speed and diagonalisation measure. It would be interesting to see if the idea of the multiple-shift method can also be implemented with other polynomial matrix decomposition algorithms such as the PSVD algorithm based on the generalised Kogbetliantz transformations [118] and the polynomial QR decomposition (PQRD) algorithm [84]. How would this affect the convergence of the algorithm? Furthermore, we have highlighted the importance of the spectral majorisation property in applications that are based on broadband subspace decomposition; however, spectral majorisation also creates high paraunitary order when diagonalising the para-Hermitian matrix where ground truth PSDs of sources are not spectrally majorised, as demonstrated in Chapter 4. It would be interesting to see if the non-majorised decomposition can be used in a particular application. The following changes can be made to generate the non-majorised solution for the SBR2 and SMD algorithms.

- In SBR2, Givens rotations with a different ordering may not only lead to a permutation but can perhaps also balance powers between channels.
- In SMD, the EVD could be replaced by a geometric mean decomposition (GMD) [125], which creates a diagonal matrix of equal positive semidefinite and real-valued entries by means of unitary operations.

In addition, the SVD has been generalised to decompose tensors [126], and a number of application areas utilise such expansions. We need to understand the differences and similarities between PEVD and tensor decomposition in order to set our approach apart and encourage its use among the research community that utilises and applies tensor decompositions.

Implementations

Based on the continuously growing interest in polynomial matrix decomposition techniques, a MATLAB polynomial EVD toolbox has been developed [127]. This toolbox only includes the two main PEVD algorithms, i.e., SBR2 and SMD. It might be helpful to the community to also archive the improved versions of these algorithms, including MS-SBR2 and MSME-SMD. Furthermore, it is now time to examine the robustness of these algorithms when implemented via field-programmable gate array (FPGA) architecture or graphical processing unit (GPU). Some recent papers [128–130] have investigated these issues but there is still room for performance improvement. Surely, it is worthwhile to conduct further research on hardware implementation.

Applications

For the application of the PEVD to spectral factorisation, discussed in Chapter 6, we have demonstrated that the spectral factor found by our proposed method is valid, but not unique. Is a unique minimum order solution possible via PEVD?

The results discussed in Chapter 7 could potentially be made more accurate by directly decomposing the channel matrix using a more straightforward PSVD algo-

rithm such as that based on generalised Kogbetliantz transformations [118], instead of formulating the PSVD via PEVD. It would be interesting to see a performance comparison between these two different PSVD approaches. In addition, the PQRD can also be used to transform the broadband MIMO channel equalisation problem into a set of SISO channel equalisation problems using back substitution, which removes the co-channel interference (CCI). Each of the SISO channel equalisation problems can then be solved using a maximum likelihood sequence estimation (MLSE) based on the Viterbi algorithm [15, 67]. Note that unlike the PSVD-based equalisation scheme, the PQRD only needs prior channel knowledge at the receiver. Further work can be done to compare these two schemes, and different equalisation methods could also be applied to this problem to see whether the BER results could be potentially improved.

References

- [1] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, San Diego, California, 1997.
- [2] P. Prandoni and M. Vetterli, *Signal Processing for Communications*. EPFL Press, Lausanne, Switzerland, 2008.
- [3] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of Computation*, vol. 19, pp. 297–301, 1965.
- [4] K. Pope and R. Bogner, "Blind signal separation I: Linear, instantaneous combinations," *Digital Signal Processing*, vol. 6, no. 1, pp. 5 – 16, 1996.
- [5] P. Smaragdis, "Blind separation of convolved mixtures in the frequency domain," *Neurocomputing*, vol. 22, no. 1, pp. 21 – 34, 1998.
- [6] S. Ikeda and N. Murata, "A method of ICA in time-frequency domain," in *Conference on Independent Component Analysis and Signal Separation*, pp. 365–371, 1999.
- [7] J. McWhirter, P. Baxter, T. Cooper, S. Redif, and J. Foster, "An EVD algorithm for para-Hermitian polynomial matrices," *IEEE Transactions on Signal Processing*, vol. 55, pp. 2158–2169, May 2007.
- [8] G. Golub and C. Van Loan, *Matrix Computations, 4th Edition*. Johns Hopkins University Press, 2013.
- [9] S. Redif, J. G. McWhirter, P. D. Baxter, and T. Cooper, "Robust broadband adaptive beamforming via polynomial eigenvalues," in *OCEANS 2006*, pp. 1–6, Sep 2006.
- [10] M. Davies, S. Lambotharan, J. Chambers, and J. McWhirter, "Broadband MIMO beamforming for frequency selective channels using the sequential best rotation algorithm," in *IEEE Vehicular Technology Conference (VTC Spring 2008)*, pp. 1147–1151, May 2008.
- [11] S. Redif and U. Fahirloglu, "Fetal ECG extraction using broadband signal subspace decomposition," in *10th Mediterranean Microwave Symposium*, pp. 381–384, Aug 2010.
- [12] M. A. Alrmah, S. Weiss, and S. Lambotharan, "An extension of the MUSIC algorithm to broadband scenarios using a polynomial eigenvalue decomposition," in *19th European Signal Processing Conference (EUSIPCO)*, pp. 629–633, Aug 2011.

- [13] S. Weiss, S. Bendoukha, A. Alzin, F. K. Coutts, I. K. Proudler, and J. Chambers, "MVDR broadband beamforming using polynomial matrix techniques," in *23rd European Signal Processing Conference (EUSIPCO)*, pp. 839–843, Aug 2015.
- [14] C. H. Ta and S. Weiss, "A design of precoding and equalisation for broadband MIMO systems," in *41st Asilomar Conference on Signals, Systems and Computers*, pp. 1616–1620, Nov 2007.
- [15] J. Foster, J. McWhirter, S. Lambouharan, I. Proudler, M. Davies, and J. Chambers, "Polynomial matrix QR decomposition for the decoding of frequency selective multiple-input multiple-output communication channels," *IET Signal Processing*, vol. 6, pp. 704–712, Sep 2012.
- [16] A. Sandmann, A. Ahrens, and S. Lochmann, "Resource allocation in SVD-assisted optical MIMO systems using polynomial matrix factorization," in *Photonic Networks; 16. ITG Symposium*, pp. 1–7, May 2015.
- [17] S. Weiss, S. Redif, T. Cooper, C. Liu, P. D. Baxter, and J. G. McWhirter, "Paraunitary oversampled filter bank design for channel coding," *EURASIP Journal on Advances in Signal Processing*, vol. 2006, pp. 1–10, 2006.
- [18] A. Jafarian and J. G. McWhirter, "A novel method for multichannel spectral factorization," in *20th European Signal Processing Conference (EUSIPCO)*, pp. 1069–1073, Aug 2012.
- [19] Z. Wang and J. G. McWhirter, "A new multichannel spectral factorization algorithm for parahermitian polynomial matrices," in *10th IMA International Conference on Mathematics in Signal Processing*, Dec 2014.
- [20] Z. Wang, J. G. McWhirter, and S. Weiss, "Multichannel spectral factorization algorithm using polynomial matrix eigenvalue decomposition," in *49th Asilomar Conference on Signals, Systems and Computers*, pp. 1714–1718, Nov 2015.
- [21] S. Redif, J. McWhirter, and S. Weiss, "Design of FIR paraunitary filter banks for subband coding using a polynomial eigenvalue decomposition," *IEEE Transactions on Signal Processing*, vol. 59, pp. 5253–5264, Nov 2011.
- [22] P. D. Baxter and J. G. McWhirter, "Blind signal separation of convolutive mixtures," in *37th Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 124–128, 2003.
- [23] P. Comon and L. Rota, "Blind separation of independent sources from convolutive mixtures," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E86-A, pp. 542–549, Mar 2003.
- [24] S. Icart, P. Comon, and L. Rota, "Blind paraunitary equalization," *Signal Processing*, vol. 89, no. 3, pp. 283 – 290, 2009.
- [25] S. Redif, "Fetal electrocardiogram estimation using polynomial eigenvalue decomposition," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 24, pp. 2483–2497, 2016.
- [26] Z. Wang and J. G. McWhirter, "A novel insight to the SBR2 algorithm for diagonalising para-hermitian matrices," in *11th IMA International Conference on Mathematics in Signal Processing*, Dec 2016.

- [27] P. P. Vaidyanathan, "Theory of optimal orthonormal subband coders," *IEEE Transactions on Signal Processing*, vol. 46, no. 6, pp. 1528–1543, 1998.
- [28] Z. Wang, J. G. McWhirter, J. Corr, and S. Weiss, "Multiple shift second order sequential best rotation algorithm for polynomial matrix EVD," in *23rd European Signal Processing Conference (EUSIPCO)*, pp. 844–848, Aug 2015.
- [29] Z. Wang, J. G. McWhirter, J. Corr, and S. Weiss, "Order-controlled multiple shift SBR2 algorithm for para-Hermitian polynomial matrices," in *IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pp. 1–5, Jul 2016.
- [30] J. Corr, K. Thompson, S. Weiss, J. G. McWhirter, S. Redif, and I. K. Proudler, "Multiple shift maximum element sequential matrix diagonalisation for para-hermitian matrices," in *IEEE Workshop on Statistical Signal Processing (SSP)*, pp. 312–315, Jun 2014.
- [31] Z. Wang, A. Sandmann, J. G. McWhirter, and A. Ahrens, "Multiple shift SBR2 algorithm for calculating the SVD of broadband optical MIMO systems," in *39th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 433–436, Jun 2016.
- [32] Z. Wang, A. Sandmann, J. G. McWhirter, and A. Ahrens, "Decoupling of broadband optical MIMO systems using the multiple shift SBR2 algorithm," *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, vol. 6, no. 1, pp. 30–37, 2017.
- [33] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications, 1st*. Cambridge University Press, Cambridge, UK, 2003.
- [34] C. H. Ta and S. Weiss, "A design of precoding and equalisation for broadband MIMO systems," in *15th International Conference on Digital Signal Processing*, pp. 571–574, Jul 2007.
- [35] S. Redif, S. Weiss, and J. G. McWhirter, "Sequential matrix diagonalization algorithms for polynomial EVD of parahermitian matrices," *IEEE Transactions on Signal Processing*, vol. 63, pp. 81–89, Jan 2015.
- [36] D. Yellin and E. Weinstein, "Criteria for multichannel signal separation," *IEEE Transactions on Signal Processing*, vol. 42, pp. 2158–2168, Aug 1994.
- [37] P. Comon, "Independent component analysis, a new concept?," *Signal processing*, vol. 36, no. 3, pp. 287–314, 1994.
- [38] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent component analysis and applications*. Academic Press, New York, USA, 2010.
- [39] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural networks*, vol. 13, no. 4, pp. 411–430, 2000.
- [40] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent component analysis*. John Wiley & Sons, Inc., 2001.
- [41] A. Mansour, N. Bencheekroun, and C. Gervaise, "Blind separation of underwater acoustic signals," in *6th International Conference on Independent Component Analysis and Blind Signal Separation (ICA'06)*, pp. 181–188, 2006.

- [42] K. I. Diamantaras and T. Papadimitriou, "MIMO blind deconvolution using subspace-based filter deflation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 4, pp. 433–436, May 2004.
- [43] D. Nuzillard and A. Bijaoui, "Blind source separation and analysis of multi-spectral astronomical images," *Astronomy and Astrophysics Supplement Series*, vol. 147, no. 1, pp. 129–138, 2000.
- [44] J. Anemüller, T. Sejnowski, and S. Makeig, "Complex independent component analysis of frequency-domain electroencephalographic data," *Neural Networks*.
- [45] S. Makeig, A. J. Bell, T.-P. Jung, and T. J. Sejnowski, "Independent component analysis of electroencephalographic data," in *Advances in Neural Information Processing Systems*, pp. 145–151, MIT Press, 1996.
- [46] C. Vayá, J. J. Rieta, C. Sánchez, and D. Moratal, "Performance study of convolutive bss algorithms applied to the electrocardiogram of atrial fibrillation," in *6th International Conference on Independent Component Analysis and Blind Signal Separation (ICA'06)*, vol. 3889, pp. 495–502, 2006.
- [47] L. De Lathauwer, B. De Moor, and J. Vandewalle, "Fetal electrocardiogram extraction by blind source subspace separation," *IEEE transactions on biomedical engineering*, vol. 47, no. 5, pp. 567–572, 2000.
- [48] L. K. Hansen, "ICA of fMRI based on a convolutive mixture model," in *9th Annual Meeting of the Organization for Human Brain Mapping*, Jun 2003.
- [49] P. Baxter, G. Spence, and J. McWhirter, "Blind signal separation on real data: Tracking and implementation," in *6th International Conference on Independent Component Analysis and Blind Signal Separation (ICA 2006)*, pp. 327–334, 2006.
- [50] M. Congedo, C. Gouy-Pailler, and C. Jutten, "On the blind source separation of human electroencephalogram by approximate joint diagonalization of second order statistics," *Clinical Neurophysiology*, vol. 119, no. 12, pp. 2677 – 2686, 2008.
- [51] V. Reju, S. N. Koh, and I. Y. Soon, "An algorithm for mixing matrix estimation in instantaneous blind source separation," *Signal Processing*, vol. 89, no. 9, pp. 1762 – 1773, 2009.
- [52] J. J. Thiagarajan, K. N. Ramamurthy, and A. Spanias, "Mixing matrix estimation using discriminative clustering for blind source separation," *Digital Signal Processing*, vol. 23, no. 1, pp. 9 – 18, 2013.
- [53] J. F. Cardoso, "Blind signal separation: statistical principles," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2009–2025, 1998.
- [54] J. F. Cardoso and A. Souloumiac, "Blind beamforming for non-Gaussian signals," in *IEE proceedings-F (radar and signal processing)*, vol. 140, pp. 362–370, Dec 1993.
- [55] I. J. Clarke, "Direct exploitation of non-Gaussianity as a discriminant," in *9th European Signal Processing Conference (EUSIPCO)*, pp. 1–3, 1998.

- [56] J. E. McWhirter, I. J. Clarke, and G. Spence, “Multilinear algebra for independent component analysis,” in *SPIE’s International Symposium on Optical Science, Engineering, and Instrumentation*, pp. 258–266, 1999.
- [57] F. R. Bach and M. I. Jordan, “Kernel independent component analysis,” *Journal of machine learning research*, vol. 3, pp. 1–48, Jul 2002.
- [58] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK User’s Guide 3rd Edition*. SIAM, Philadelphia, 1999. Available at http://www.netlib.org/lapack/lug/lapack_lug.html.
- [59] G. H. Golub and H. A. van der Vorst, “Eigenvalue computation in the 20th century,” *Journal of Computational and Applied Mathematics*, vol. 123, no. 1, pp. 35 – 65, 2000.
- [60] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing, 3rd Edition*. Cambridge University Press, Cambridge, UK, 2007.
- [61] S. S. Haykin, *The Adaptive Filter Theory, 2nd Edition*. Prentice-Hall, Upper Saddle River, NJ, 1991.
- [62] M. H. Hayes, *Statistical digital signal processing and modeling*. John Wiley & Sons, 2009.
- [63] A. Hyvärinen, “Fast and robust fixed-point algorithms for independent component analysis,” *IEEE transactions on Neural Networks*, vol. 10, no. 3, pp. 626–634, 1999.
- [64] M. S. Pedersen, J. Larsen, U. Kjems, and L. C. Parra, “A survey of convolutive blind source separation methods,” in *Springer Handbook on Speech Processing and Speech Communication*, Springer Press, 2007.
- [65] S. V. Gerven and D. V. Compernelle, “Signal separation by symmetric adaptive decorrelation: stability, convergence, and uniqueness,” *IEEE Transactions on Signal Processing*, vol. 43, pp. 1602–1612, Jul 1995.
- [66] G. G. Raleigh and J. M. Cioffi, “Spatio-temporal coding for wireless communication,” *IEEE Transactions on Communications*, vol. 46, pp. 357–366, Mar 1998.
- [67] J. Foster, *Algorithms and Techniques for Polynomial Matrix Decompositions*. PhD thesis, School of Engineering, Cardiff University, 2008.
- [68] T. Kailath, *Linear Systems*. Information and System Sciences Series, Prentice-Hall, Englewood Cliffs, NJ, 1980.
- [69] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Prentice-Hall, Upper Saddle River, NJ, 1993.
- [70] M. A. Alrmah, J. Corr, A. Alzin, K. Thompson, and S. Weiss, “Polynomial subspace decomposition for broadband angle of arrival estimation,” in *4th International Conference on Sensor Signal Processing for Defence (SSPD)*, pp. 1–5, Sep 2014.

- [71] M. Sørensen, L. D. Lathauwer, S. Icart, and L. Deneire, “On Jacobi-type methods for blind equalization of paraunitary channels,” *Signal Processing*, vol. 92, no. 3, pp. 617 – 624, 2012.
- [72] H. J. S. Smith, “On systems of linear indeterminate equations and congruences,” *Philosophical Transactions of the Royal Society of London*, vol. 151, pp. 293–326, 1861.
- [73] M. Davies, *Polynomial Matrix Decomposition Techniques for Frequency Selective MIMO Channels*. PhD thesis, School of Electronic, Electrical and System Engineering, Loughborough University, 2010.
- [74] J. M. Maciejowski, *Multivariable feedback design*. Addison-Wesley, Wokingham, UK, 1989.
- [75] X. Gao, T. Q. Nguyen, and G. Strang, “On factorization of M-channel paraunitary filterbanks,” *IEEE Transactions on Signal Processing*, vol. 49, pp. 1433–1446, Jul 2001.
- [76] P. A. Regalia and D.-Y. Huang, “Attainable error bounds in multirate adaptive lossless FIR filters,” in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, pp. 1460–1463 vol.2, May 1995.
- [77] R. H. Lambert, *Multichannel blind deconvolution: FIR matrix algebra and separation of multipath mixtures*. PhD thesis, University of Southern California, Los Angeles, CA, USA, 1996.
- [78] R. H. Lambert, M. Joho, and H. Mathis, “Polynomial singular values for number of wideband source estimation and principal components analysis,” in *International Conference on Independent Component Analysis*, pp. 379–383, 2001.
- [79] A. Tkacenko, “Approximate eigenvalue decomposition of para-Hermitian systems through successive FIR paraunitary transformations,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4074–4077, Mar 2010.
- [80] S. Redif, S. Weiss, and J. G. McWhirter, “An approximate polynomial matrix eigenvalue decomposition algorithm for para-Hermitian matrices,” in *IEEE International Symposium on Signal Processing and Information Technology (ISPIT)*, pp. 421–425, Dec 2011.
- [81] J. G. McWhirter and P. Baxter, “A novel technique for broadband singular value decomposition,” in *12th Annual Adaptive Sensor Array Workshop*, Mar 2004.
- [82] S. Icart and P. Comon, “Some properties of laurent polynomial matrices,” in *9th IMA International Conference on Mathematics in Signal Processing*, Oct 2012.
- [83] J. Corr, K. Thompson, S. Weiss, I. K. Proudler, and J. G. McWhirter, “Row-shift corrected truncation of paraunitary matrices for PEVD algorithms,” in *23rd European Signal Processing Conference (EUSIPCO)*, pp. 849–853, Aug 2015.
- [84] J. A. Foster, J. G. McWhirter, M. R. Davies, and J. A. Chambers, “An algorithm for calculating the QR and singular value decompositions of polynomial matrices,” *IEEE Transactions on Signal Processing*, vol. 58, pp. 1263–1274, Mar 2010.

- [85] J. Corr, K. Thompson, S. Weiss, I. Proudler, and J. McWhirter, “Shortening of paraunitary matrices obtained by polynomial eigenvalue decomposition algorithms,” in *5th International Conference on Sensor Signal Processing for Defence (SSPD)*, pp. 1–5, Sep 2015.
- [86] J. Foster, J. McWhirter, and J. Chambers, “Limiting the order of polynomial matrices within the SBR2 algorithm,” in *6th IMA International Conference on Mathematics in Signal Processing*, Dec 2006.
- [87] C. H. Ta and S. Weiss, “Shortening the order of paraunitary matrices in SBR2 algorithm,” in *Information, Communications Signal Processing, 2007 6th International Conference on*, pp. 1–5, Dec 2007.
- [88] S. Redif, S. Weiss, and J. G. McWhirter, “Relevance of polynomial matrix decompositions to broadband blind signal separation,” *Signal Processing*, vol. 134, pp. 76 – 86, 2017.
- [89] A. Kirac and P. P. Vaidyanathan, “On existence of FIR principal component filter banks,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, pp. 1329–1332, May 1998.
- [90] J. Corr, K. Thompson, S. Weiss, J. G. McWhirter, and I. K. Proudler, “Causality-constrained multiple shift sequential matrix diagonalisation for parahermitian matrices,” in *22nd European Signal Processing Conference (EUSIPCO)*, pp. 1277–1281, Sep 2014.
- [91] J. Corr, K. Thompson, S. Weiss, J. G. McWhirter, and I. K. Proudler, “Maximum energy sequential matrix diagonalisation for parahermitian matrices,” in *48th Asilomar Conference on Signals, Systems and Computers*, pp. 470–474, Nov 2014.
- [92] J. Corr, K. Thompson, S. Weiss, I. K. Proudler, and J. G. McWhirter, “Impact of source model matrix conditioning on PEVD algorithms,” in *2nd IET International Conference on Intelligent Signal Processing (ISP)*, pp. 1–6, Dec 2015.
- [93] A. Papoulis, *Probability, random variables, and stochastic processes*, 3rd. McGraw-Hill, New York, USA, 1991.
- [94] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics: A Foundation for Computer Science, 2nd Edition*. Boston, MA, USA: Addison-Wesley Longman Publishing, 1994.
- [95] A. S. Householder, “Unitary triangularization of a nonsymmetric matrix,” *Journal of the ACM (JACM)*, vol. 5, pp. 339–342, Oct 1958.
- [96] I. S. Dhillon, *A New $O(n^2)$ Algorithm for the Symmetric Tridiagonal Eigenvalue/Eigenvector Problem*. PhD thesis, Department of Computing Science, University of California, Berkeley, 1997.
- [97] J. Corr, K. Thompson, S. Weiss, J. G. McWhirter, and I. K. Proudler, “Performance trade-offs in sequential matrix diagonalisation search strategies,” in *6th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 25–28, Dec 2015.

- [98] C. Charoenlarnpopparut, "One-dimensional and multidimensional spectral factorization using Grobner basis approach," in *Asia-Pacific Conference on Communications*, pp. 201–204, Oct 2007.
- [99] R. F. Fischer, "Sorted spectral factorization of matrix polynomials in MIMO communications," *IEEE Transactions on Communications*, vol. 12, pp. 945–951, Jun 2005.
- [100] B. Du, X. Xu, and X. Dai, "Minimum-phase FIR precoder design for multicasting over MIMO frequency-selective channels," *Journal of Electronics (China)*, vol. 30, pp. 319–327, 2013.
- [101] H. Han, H. Kim, N. Kim, and H. Park, "An enhanced QAM-FBMC scheme with interference mitigation," *IEEE Communications Letters*, vol. 20, pp. 2237–2240, Nov 2016.
- [102] G. Wilson, "Factorization of the covariance generating function of a pure moving average process," *SIAM Journal on Numerical Analysis*, vol. 6, no. 1, pp. 1–7, 1969.
- [103] G. Wilson, "The factorization of matricial spectral densities," *SIAM Journal on Applied Mathematics*, vol. 23, no. 4, pp. 420–426, 1972.
- [104] G. Janashia, E. Lagvilava, and L. Ephremidze, "A new method of matrix spectral factorization," *IEEE Transactions on Information Theory*, vol. 57, pp. 2318–2326, Apr 2011.
- [105] V. Kucera, "Factorization of rational spectral matrices: a survey of methods," in *International Conference on Control*, vol. 2, pp. 1074–1078, Mar 1991.
- [106] A. H. Sayed and T. Kailath, "A survey of spectral factorization methods," *Numerical Linear Algebra with Applications*, vol. 8, no. 6-7, pp. 467–496, 2001.
- [107] I. Selesnick, "Spectral factorization - polynomial root finding and the Leja ordering." Available at <http://eeweb.poly.edu/iselesni/EL713/sfact/sfact.pdf>.
- [108] L. M. N. Wiener, "The prediction theory of multivariate stochastic processes, Part II," *Acta Math.*, vol. 99, no. 1, pp. 93–137, 1958.
- [109] L. M. N. Wiener, "The prediction theory of multivariate stochastic processes, Part I," *Acta Math.*, vol. 98, no. 1–4, pp. 111–150, 1957.
- [110] A. Papoulis, *Signal Analysis*. McGraw-Hill, New York, USA, 1977.
- [111] G. Janashia, E. Lagvilava, and L. Ephremidze, "Matrix spectral factorization and wavelets," *Journal of Mathematical Sciences*, vol. 195, no. 4, pp. 445–454, 2013.
- [112] PolyX, *The Polynomial Toolbox for MATLAB*. PolyX Ltd, Czech Republic, Mar 1999. Available at <http://www.polyx.com/download/manual.pdf.gz>, version 2.0.
- [113] A. C. Singer, N. R. Shanbhag, and H. M. Bae, "Electronic dispersion compensation: An overview of optical communications systems," *IEEE Signal Processing Magazine*, vol. 25, pp. 110–130, Nov 2008.

- [114] P. J. Winzer and G. J. Foschini, "MIMO capacities and outage probabilities in spatially multiplexed optical transport systems," *Optics Express*, vol. 19, pp. 16680–16696, Aug 2011.
- [115] G. G. Raleigh and V. K. Jones, "Multivariate modulation and coding for wireless communication," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 851–866, May 1999.
- [116] A. Scaglione, P. Stoica, S. Barbarossa, G. B. Giannakis, and H. Sampath, "Optimal designs for space-time linear precoders and decoders," *IEEE Transactions on Signal Processing*, vol. 50, pp. 1051–1064, May 2002.
- [117] A. Sandmann, A. Ahrens, and S. Lochmann, "Performance analysis of polynomial matrix SVD-based broadband MIMO systems," in *5th International Conference on Sensor Signal Processing for Defence (SSPD)*, pp. 1–5, Sep 2015.
- [118] J. G. McWhirter, "An algorithm for polynomial matrix SVD based on generalised Kogbetliantz transformations," in *2010 18th European Signal Processing Conference*, pp. 457–461, Aug 2010.
- [119] J. Proakis and M. Salehi, *Digital Communications*. McGraw-Hill, New York, USA, 2008.
- [120] I. Kalet, "Optimization of linearly equalized qam," *IEEE Transactions on Communications*, vol. 35, pp. 1234–1236, Nov 1987.
- [121] G. Forney, R. Gallager, G. Lang, F. Longstaff, and S. Qureshi, "Efficient modulation for band-limited channels," *IEEE Journal on Selected Areas in Communications*, vol. 2, pp. 632–647, Sep 1984.
- [122] A. Sandmann, A. Ahrens, and S. Lochmann, "Experimental description of multimode MIMO channels utilizing optical couplers," in *Photonic Networks; 15. ITG Symposium*, pp. 1–6, May 2014.
- [123] A. Tarighat, R. C. J. Hsu, A. Shah, A. H. Sayed, and B. Jalali, "Fundamentals and challenges of optical multiple-input multiple-output multimode fiber links," *IEEE Communications Magazine*, vol. 45, pp. 57–63, May 2007.
- [124] A. Sandmann, A. Ahrens, and S. Lochmann, "Zero-forcing equalisation of measured optical multimode MIMO channels," in *Communications in Computer and Information Science*, vol. 554, pp. 115–130, Springer International Publishing, 2015.
- [125] Y. Jiang, W. W. Hager, and J. Li, "The geometric mean decomposition," *Linear Algebra and its Applications*, vol. 396, pp. 373–384, 2005.
- [126] A. Cichocki, D. Mandic, L. D. Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. PHAN, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, pp. 145–163, Mar 2015.
- [127] S. Weiss, J. G. McWhirter, I. K. Proudler, J. Corr, and K. Thompson, *Polynomial EVD Toolbox for MATLAB*. University of Strathclyde, Glasgow, Scotland, 2014. Available at <http://pevd-toolbox.eee.strath.ac.uk/>.

- [128] S. Kasap and S. Redif, “Novel field-programmable gate array architecture for computing the eigenvalue decomposition of para-hermitian polynomial matrices,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, pp. 522–536, Mar 2014.
- [129] S. Redif and S. Kasap, “Novel reconfigurable hardware architecture for polynomial matrix multiplications,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, pp. 454–465, Mar 2015.
- [130] M. Carcenac, S. Redif, and S. Kasap, “GPU parallelization of the sequential matrix diagonalization algorithm and its application to high-dimensional data,” *The Journal of Supercomputing*, pp. 1–32, 2017.