KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Time Series Clustering

Oliver Lauwers

Supervisor:
Prof. dr. ir. Bart De Moor

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

May 2020

Time Series Clustering

Oliver LAUWERS

Examination committee:
Prof. dr. ir. Hendrik Van Brussel, chair
Prof. dr. ir. Bart De Moor, supervisor
Prof. dr. ir. Jan Beirlant
Dr. ir. Oscar Mauricio Agudelo
Dr. ir. Katrien De Cock
Prof. dr. ir. Tijl De Bie
(Universiteit Gent)

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

May 2020

© 2020 KU Leuven – Faculty of Engineering Science
Uitgegeven in eigen beheer, Oliver Lauwers, Kasteelpark Arenberg 10 box 2446, B-3001 Leuven (Belgium)

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

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

Preface

Going through this text, I see the influence of many different people. While this book might show my name as author, it is very much a collective effort by a variety of silent contributors, who have indirectly added their wisdom to its pages. Every single one of them has my sincerest gratitude. No list I make can ever be exhaustive, but I will attempt to explicitly mention and thank some of the main contributors.

First and foremost, I thank professor Bart De Moor, my supervisor, for the many opportunities and insights he has offered me. He is a man of many talents and interests, and one of the most intelligent people I have ever had the luck to encounter. Even throughout the worst and darkest times of the past few years, he supported me without doubt, and without hesitation. The freedom he gave me in balancing the professional and private aspects of life has meant and means more to me than I could ever express. For this, and much more, he has my everlasting respect and gratitude.

I also very much thank the examination committee: the chairman, professor Hendrik Van Brussel, and members, professor Jan Beirlant, professor Tijl De Bie, doctor Katrien De Cock and doctor Oscar Mauricio Agudelo. Under the trying circumstances of a global pandemic and quarantine, they still carved time out of their busy schedules to carefully read and assess this thesis. I am grateful for their detailed and thoughtful remarks, comments and questions, which have greatly improved this text.

Allow me to emphasize once again my thanks to Katrien De Cock, who not only provided most of the relevant previous work to this thesis in her own PhD, but was of tremendous importance in proofreading this work in its earliest versions. She has spent more time reading and rereading this text than is healthy for one person, and I thank her for it.

Let me also take the time to thank the IWT and FWO, for funding my research. The confidence they had in my initial project, and the empathy they exhibited

when times were tough, have made it clear that I could not have wished for a better employer. I have been also very fortunate to be allowed into the InnoEnergy PhD School programme, which I thank for offering not only many interesting courses on all aspects of the energy industry, but also an invaluable network of colourful people. We have spent many days working together, and many evenings decompressing. I hope there are many more to come.

Through this programme, I had the wonderful opportunity to collaborate with the EFESE department of EDF on a real-world application, discussed later in this text. I thank all the people there for the delightful four months I spent with them, and the delicious choucroute in the cantine. Let me highlight perhaps Tamsir Diop, who closely guided me through the tangle of available data, and Cyril Gisbert, for the interesting discussions on the many issues involved with the international energy markets.

Of course, I cannot forget my fantastic colleagues at home, some of which have become friends for life. There are much too many to thank explicitly, yet they all have added in one way or another to the past five years. While offering my gratitude to all of them, let me mention a few of my fellow researchers by name: Marc Claesen, for offering me guidance and support from the very beginning, and keeping me sane until the very end; Supinya Piampongsant, for being the best floor manager we have ever had; Bob Vergauwen, Christof Vermeersch and Bram Geelen, proud members of snooker team *De Blinkende Ballen*; Tina Smets, for putting up with the ever more convoluted rules of our local variant on table football, *Chococrèmebal*; and Philippe Dreesen, for helping me take my first steps as a teaching assistent, and having the best conference theme song.

I also thank the colleagues that offered administrative and technical support, Elsy Vermoesen, Maarten Truyens, Wim Devroye, John Vos and Aldona Niemiro Sznajder. Most deserving of praise and gratitude, is Ida Tassens. With her exceptional organization skills, silent persistence and unbelievable resilience, she does more to keep our research unit running smoothly than we will ever know.

Juan Diaz Dorronsoro and Oliver Janssen are also of exceptional importance, not only to this PhD, but to my time at the university in general. Their wisdom, intelligence and research-mindedness were inspirational for my choice to pursue a PhD. I thank them for their friendship.

My life has been enriched by many friends, who have supported me every step of the way. I owe a great debt to all of them, and to four of them in particular: Anneleen Verlinden, Claïs Lemmens, Oliver Van Oekelen, and Mathias Anthonissen. Through the good and the bad, I can always count on them. They are my safe haven. Let us hope they will put up with me for much, much longer still, for I am lost without them!

So I come to my family. They have all formed me in one way or another, and I would not be where I am today without them. I thank my dear parents, Peter Lauwers en Maryse Vyncke, for giving me everything I have. Their love, care and guidance carry me forward every day; anything I have achieved is their achievement. I also thank my brother and sister, Peter-Jan and Elise Lauwers, for being the wetstone on which my research skills were honed. Without them, persistence would have been stubbornness, and inquisitiveness would have been disputatiousness. I thank them for showing me how to use these tools appropriately, by being on the receiving end of them too many times to count.

A special word of thanks as well to Raphael Mortier, my godson, and his brother Manu. They have a way of intuitively grasping the important aspects of life, that I envy, and have shown me many times how to be joyful. When the complexity of academic research became too much to bear, they were there to show its relative insignificance in relation to a water balloon fight, a game of hide-and-seek, and an evening of bargaining over an extra cookie or piece of candy. May they remain forever happy people, and take good care of each other.

I want to thank the staff of the National Multiple Sclerosis Center, and especially division C2 – they have too many names to all write down here, but they are written in my heart – and doctor Tom Meurrens, for their silent support the past year and a half. They do not know how instrumental their tireless effort and compassionate care – for Gwen, as well as for me – was to achieve the peace of mind I needed to write this thesis. While the circumstances could have been better, I am honored to have met all of them, and spent a lot of time with them. There is no way to thank them enough.

This brings me, finally, to thanking Gwen Verguts, for her unconditional love, mental support and proverbial cheerfulness. She has taught me true perseverance, love without limits, and how to be simply happy, whatever the circumstances. I am humbled and inspired by her example, and feel very fortunate to share our love. In the face of the worst conditions, she manages not only to keep, but also to spread her optimism, with a flick of an eyebrow, a grin on her face, or a kiss of her lips.

She has put a smile in my heart that will live there forever. This text, and all the work that went into it, is dedicated to her, and to that smile. It is as much hers as mine.

Oliver

Leuven
May 2020

Abstract

The notion of similarity is becoming more and more important in data analysis. Comparing two objects quantitatively is a fundamental building block of most machine learning algorithms, be it supervised or unsupervised. While for simple mathematical objects, the notion of similarity might be intuitively clear (when comparing how different two integers are, for example, we can just look at the absolute difference between them), the situation becomes very cloudy very quickly as we look at more complex data types. In this thesis, we focus on data evolving throughout time, known as signals or time series.

The past few decades have seen an incredible increase in time series data. Larger data storage capacity along with new technological solutions to industrial problems has led to more and more data being generated in industrial contexts, as exemplified by large equipment in a busy factory having all sorts of sensors monitored by a control room engineer, or a wind turbine recording information on wind speed and electricity output, as well as internal data such as temperatures of different components.

Also on the level of nation-wide, or even cross-nation structures like the electricity grid, the storing, processing and transmission of information is crucial to make the different parts and pieces work together. This has given rise to things like smart meters, which monitor energy usage of every household on a very fine-grained scale, or the European transnational electricity network, where data for all the different power lines is necessary to clear the international energy market. More and more, even, individuals are connecting themselves up with different sensors, generating terabytes of detailed data on their movement, interactions, and even medical parameters (blood pressure, heart rate, ...).

Given these evolutions (and some of them will return later on in the text as illustrations and applications of the theoretical work done in this research project), it is evident that similarity measures for time series are a timely and relevant topic. It is also true that it is inherently challenging: the right

distance measure depends on the specific application and what we are trying to accomplish. Traditional vector distances, that compare the shapes of two signals, or more statistical measures, comparing the distribution of data points in the series can be sensible, but even very simple tasks, such as discerning a sine wave from a Gaussian noise signal, can be deceptively hard, as we show later on in the text. There is thus an ever-increasing need for similarity measures that base their quantitative comparison on interesting, sensible and interpretable aspects of the time series.

This need is well-understood in fields of engineering such as signal processing and control systems theory, where a signal is seen as being produced by some underlying dynamical system. The system is the fundamental object of interest, and the signal is just a particular (often noisy) realization that allows us to interact with the underlying system. While this readily allows interpretation and comparison of different dynamics in any way we please, explicitly estimating or finding mathematical models for these underlying dynamics often is an arduous, resource-intensive, and computationally expensive task.

In this thesis we investigate a distance measure, known as the weighted cepstral distance measure, that can be calculated from just the time series data alone, yet can be interpreted in terms of the underlying dynamics of the signal. In this way, we can quantify similarity between two different dynamical systems, without having to explicitly construct mathematical models for them and estimate their model parameters. In its original formulation, this measure made some constraining assumptions on the underlying dynamics of two signals it tried to compare. The work presented here relaxes these assumptions, allowing for general linear time invariant single-input single-output (LTI SISO) systems with stochastic or deterministic inputs, and makes progress in extending it further to multiple-input multiple-output (MIMO) systems, by extending some of the theoretical notions used by the distance measure.

We develop a clustering framework based on the extended versions of the weighted cepstral distance measure, and provide interpretations for the cluster centers and the variance within a cluster, in terms of the underlying dynamics. We show that this clustering algorithm solves the problem of time series subsequence clustering, a notoriously hard challenge in time series research. The theoretical notions, claims, and techniques are amply tested and shown to work in many real-life applications, the most ambitious one being an industrial project in collaboration with Électricité De France (EDF), a multinational utility company, on the clearance of international energy markets.

In doing so, we exhibit the applicability of the techniques developed here, and show that the work presented contributes meaningfully to both academic research on time series analysis, and to industrial solutions for real-life problems.

Beknopte samenvatting

Het begrip gelijkaardigheid wordt steeds belangrijker in data-analyse. Kwantificeren hoe twee objecten van elkaar verschillen of op elkaar gelijken is van fundamenteel belang in de meeste zelflerende algoritmes. Voor eenvoudige mathematische structuren, kan gelijkaardigheid intuïtief duidelijk zijn (bij twee gehele getallen volstaat het vaak om naar hun absolute verschil te kijken, bijvoorbeeld). Voor complexere gegevenssoorten, wordt dit echter stelselmatig moeilijker. In deze thesis spitsen we ons toe op gegevens die evolueren doorheen de tijd, gekend als signalen of tijdreeksen.

De voorbije decennia kenden een fenomenale toename in het genereren van tijdreeksen. Grotere gegevensopslagcapaciteit en technologische vooruitgang leidden in industriële sectoren tot een grote toename van de hoeveelheid gemeten data, waarbij, bijvoorbeeld, grote machinerie in fabrieken door middel van sensoren allerhande continu in het oog gehouden wordt vanuit een controlekamer. Windturbines slaan informatie over windsnelheid en electriciteitsproductie op, net zoals interne data over bijvoorbeeld de temperatuur van verschillende componenten.

Ook op het niveau van nationale of supranationale structuren, zoals het elektriciteitsnet, wordt opslag, verwerking en verzending van informatie cruciaal om de verschillende onderdelen te laten samenwerken. We zien dan ook de ontwikkeling en uitrol van zaken als slimme meters, die elektriciteitsverbruik van individuele huishoudens zeer gedetailleerd opmeten. In het transnationale Europese energienetwerk worden gegevens over individuele electriciteitskabels gebruikt om de internationale energieprijzen te bepalen. Zelfs op persoonsniveau houden mensen vaak gedetailleerde gegevens doorheen de tijd bij, over hun locatie, interacties en zelfs medische parameters (bloeddruk, hartslag, ...).

De voorbeelden hierboven (waarvan sommige verderop in de tekst nog zullen terugkeren als illustraties en toepassingen van de theoretische ontwikkelingen in dit werk) tonen het belang en de actuele relevantie van gelijkaardigheidsmaten

voor tijdreeksen aan. Tegelijk is dit ook een uitdagend onderwerp: de keuze van afstandsmaat hangt af van de specifieke toepassing en doelstellingen. Traditionele vectorafstanden, die de vorm van twee tijdreeksen vergelijken, en statistische maten, die de verdeling van datapunten in de signalen in rekening brengen, kunnen soms zinvol zijn. Zelfs eenvoudige taken, echter, zoals onderscheid maken tussen Gaussissche ruis en sinusgolven, kunnen onverwacht moeilijk zijn, zoals we verderop zullen zien. Er blijft dus een groeiende nood aan afstandsmaten die zich baseren op interessante, zinvolle en interpreteerbare aspecten van de tijdreeksen.

In de ingenieurswetenschappen is dit niets nieuws. In velden als signaalverwerking en regeltechniek worden signalen beschouwd als het eindproduct van een onderliggend dynamisch systeem. Daarbij zijn we fundamenteel geïnteresseerd in dit systeem, eerder dan in het signaal, dat slecht een ruizige observatie van het onderliggende model is. De dynamica van deze onderliggende systemen kan gemakkelijk geïnterpreteerd en onderling vergeleken worden, maar het expliciet opstellen en schatten van wiskundige modellen en hun parameters is een moeilijke en computationeel dure taak, waar veel expertise voor nodig is.

In dit werk onderzoeken we een afstandsmaat, de gewogen cepstrale afstandsmaat, die we kunnen berekenen op basis van enkel de tijdreeksgegevens, maar toch interpreteerbaar is volgens de onderliggende dynamica. Zo kunnen we dus gelijkenissen tussen verschillende dynamische systemen kwantificeren, zonder daarom expliciet wiskundige modellen te moeten opstellen en parameters te schatten. In zijn originele formulering werden een aantal aannames gemaakt op de toegelaten onderliggende dynamica van te vergelijken signalen. In deze thesis lossen we enkele van deze aannames, en breiden het toepassingsgebied van deze maat uit naar algemene lineaire, tijdsinvariante system met één (deterministische of stochastische) ingang en één uitgang, en boeken vooruitgang in een uitbreiding naar systemen met meerdere in- en uitgangen.

We werken verder een formeel kader voor tijdreeksclusteren uit, gebaseerd op deze uitbreidingen van de gewogen cepstrale afstand, en interpreteren de resulterende clustercentra en -variantie in termen van de onderliggende dynamica. Dit clusteralgoritme lost ook het probleem van tijdreeksdeelsequentieclusteranalyse op, een gekende moeilijke uitdaging in tijdreeksonderzoek. De theoretische ontwikkelingen in deze tekst worden veelvuldig getest door middel van realistische toepassingen. De meest ambitieuze daarbij is een grotere samenwerking met Électricité De France (EDF), een internationaal nutsbedrijf, rond de internationale energiemarkten.

Zo expliciteren we de toepasbaarheid en industriële relevantie van de uitgewerkte technieken. We leveren dus een betekenisvolle bijdrage aan zowel academisch tijdreeksenonderzoek en industriële oplossingen voor reële problemen.

List of Abbreviations

AR Autoregressive. 10

ARMA Autoregressive moving-average. 47

ATC Available transfer capacity. 159

CB Critical branch. 160

CSTR Continuous stirred tank reactor. 150

DTW Dynamic time warping. 47

ECG Electrocardiogram. 4

EDF Électricité de France. 8

EEG Electroencephalogram. 4

EFESE Économie, fonctionnement et études des systèmes énergétiques. 159

FB Flow based. 160

FFT Fast Fourier transform. 20

IFFT Inverse fast Fourier transform. 28

LTI Linear time invariant. 13

MIMO Multiple-input multiple-output. 2

PSD Power spectral density. 86

SCADA Supervisory control and data acquisition. 110

SISO Single-input single-output. 13

SVD Singular value decomposition. 76

TSO Transmission system operator. 159

List of Symbols

$[A \lhd B]$	The principal angles between the column spaces of matrices A and B.
α	Roots of $a(z)$, usually a stable pole of a system.
$*$	Convolution operator.
$\bar{\cdot}$	Complex conjugate.
β	Roots of $b(z)$, usually a stable zero of a system.
$.^\intercal$	Transpose.
$.^H$	Hermitian conjugate.
δ	Roots of $d(z)$, usually an unstable zero of a system.
γ	Roots of $c(z)$, usually an unstable pole of a system.
Γ^i	Observability matrix with i (block) rows.
Γ^i_{inv}	Observability matrix of the inverse system, with i rows.
Γ^i_V	Vandermonde matrix of the poles of a system, with i rows.
$\hat{c}(k)$	Complex cepstrum.
κ	Kernel function.
$\langle \cdot, \cdot \rangle$	Inproduct.
$\mathbb{1}$	Unity matrix.
$\mathbb{D}$	Unit disc.
$\mathbb{R}$	Real numbers.
$\mathbb{R}^N$	n-dimensional vector space over $\mathbb{R}$.

$\mathcal{C}$ Partitioning of a set.

$\mathcal{D}$ Dirichlet space.

$\mathcal{F}$ Fourier transform.

$\mathcal{F}^{-1}$ Inverse Fourier transform.

Φ Power spectral density.

$\Pi_U^{\perp}$ Orthogonal projection matrix on the orthogonal complement of the row space of U.

Π_U Orthogonal projection matrix on the row space of U.

ρ Autocorrelation function.

σ Standard deviation.

$\mathrm{E}[\cdot]$ Expected value.

$\mathrm{range}(A)$ Column space of the matrix A.

θ Angle

A, B, C, D State space description matrices.

$a(z)$ Pole polynomial

B_l Complete exponential Bell polynomial of order l.

$c(k)$ Power cepstrum.

C_i Cluster i.

$d(x, y)$ Distance function, d, evaluated in the elements x and y.

$d : X \times X \to [0, \infty)$ Metric or distance function over the set, X.

d_c Weighted cepstral distance.

d_e Euclidean distance.

e_l Elementary symmetric polynomial of order l.

g Gain of a system.

$h(k)$ Impulse response.

$H(z)$ Transfer function.

$H^{-1}(z)$ Transfer function of the inverse system.

k Discrete time variable or cepstrum domain variable.

L_2-norm Synonym of the Euclidean distance.

$P(x_1, \ldots, x_n)$ Polynomial in the variables $x_1, \ldots, x_n$.

p_l Power sum polynomial of order l.

t Continuous time variable.

T_s Sampling time.

u Signal, usually input signal.

$U(z)$ z-transform of the input signal u.

w White noise signal.

X Data set.

X^j State sequence of a system.

y Signal, usually output signal.

$Y(z)$ z-transform of the output signal y.

$Y^{i,j}$ Output Hankel matrix, with i rows and j columns, $i << j$.

z Variable in the complex plane.

Contents

Abstract v

Beknopte samenvatting vii

List of Abbreviations x

List of Symbols xiii

Contents xv

List of Figures xix

List of Tables xxix

1 Introduction 1
- 1.1 The weighted cepstral distance: a hybrid approach 4
  - 1.1.1 Time series analysis 4
  - 1.1.2 Similarity measures 5
  - 1.1.3 Positioning of this thesis 7
- 1.2 Industrial relevance 8
- 1.3 A simple example 9
  - 1.3.1 Commonly used distance measures 9
  - 1.3.2 The weighted cepstral distance 12
- 1.4 Scientific programme 14

2 The cepstrum 17
- 2.1 Dynamical systems theory 17
- 2.2 The power cepstrum 20
  - 2.2.1 The model power cepstrum 23
  - 2.2.2 Interpretation in poles and zeros 23
- 2.3 The complex cepstrum 26

2.4 Computational aspects . . . 27
2.4.1 Power cepstrum . . . 28
2.4.2 Complex cepstrum . . . 28
2.5 The weighted power cepstral distance . . . 30
2.5.1 The weighted (power) cepstral norm . . . 32
2.5.2 The weight of the zero-th cepstrum coefficient . . . 33
2.6 Model assumptions . . . 35
2.6.1 Models with poles and zeros . . . 35
2.6.2 Stability . . . 36
2.7 Subspace angles . . . 37
2.7.1 Principal angles . . . 37
2.7.2 Subspace angles . . . 39
2.7.3 Subspace angles kernel . . . 39
2.7.4 Subspace angles kernel from output signals . . . 42
2.7.5 Models with poles and zeros . . . 43

3 A time series distance measure for efficient clustering of input/output signals by their underlying dynamics 45
3.1 Introduction . . . 46
3.2 Existing methods . . . 47
3.2.1 Raw data distance measures . . . 48
3.2.2 Model-based distance measures . . . 50
3.3 Cepstral distance . . . 51
3.3.1 Original cepstral norm . . . 52
3.3.2 Extended cepstral distance . . . 53
3.3.3 Computational overview . . . 55
3.4 Application on electrical circuits . . . 55
3.4.1 Simulation set-up . . . 56
3.4.2 Results . . . 57
3.4.3 Non-linear loads . . . 60
3.5 Conclusion and further research . . . 61
3.A Non-linear loads . . . 63

4 Subspace angles between input/output systems 67
4.1 Deterministic weighted cepstral distance . . . 68
4.2 Subspace angles in the deterministic case . . . 70
4.2.1 Kernel functions . . . 71
4.2.2 The subspace angles kernel . . . 71
4.3 Computing the subspace angles . . . 72
4.3.1 Subspace angles from input and output signals . . . 73
4.3.2 Computing the column space of the observability matrix from input and output signals: LQ decomposition . . . 76
4.4 Model assumptions . . . 78

4.4.1 Models with poles and zeros . . . 80
4.4.2 Stability . . . 82
4.5 Numerical illustration of the relation between the weighted cepstral distance and the subspace angles kernel . . . 84
4.6 Computational aspects of the weighted power cepstral distance 86
4.7 Conclusion . . . 91

5 Weighted cepstral clustering 93
5.1 A cepstral clustering algorithm . . . 95
5.1.1 k-means clustering . . . 95
5.1.2 Cepstral k-means clustering . . . 97
5.1.3 Interpretation of clustering results . . . 98
5.2 Windowing . . . 102
5.2.1 Time series subsequence clustering . . . 103
5.3 Unsupervised anomaly detection in wind turbines . . . 110
5.3.1 Data . . . 110
5.3.2 Clustering results . . . 111

6 Cepstral identification of autoregressive systems 115
6.1 Estimating dynamics present in a cluster . . . 116
6.2 Symmetric polynomials . . . 117
6.2.1 Elementary symmetric polynomials . . . 117
6.2.2 Power sum polynomials . . . 118
6.2.3 Newton's identities . . . 119
6.3 Cepstral identification of autoregressive systems . . . 120
6.4 Stability . . . 123
6.5 Numerical examples . . . 123
6.5.1 Numerical illustration . . . 124
6.5.2 Synthetic data . . . 124
6.5.3 Mean of cepstral cluster . . . 127
6.6 Applications on real-life data . . . 128
6.6.1 Yule's historic sunspot numbers . . . 130
6.6.2 Structural health monitoring via autoregressive models . 132

7 A multiple-input multiple-output cepstrum 139
7.1 Introduction . . . 140
7.2 Concepts, notation and definitions . . . 141
7.2.1 SISO cepstrum . . . 142
7.2.2 Smith-McMillan form . . . 144
7.3 MIMO cepstrum . . . 145
7.3.1 Definition of the MIMO cepstrum . . . 146
7.3.2 Interpretation in terms of poles and zeros . . . 146
7.3.3 Computation when $m = l$ . . . 147

7.4 Illustration and application 148
7.4.1 Numerical illustration 149
7.4.2 Case study: fouling in non-isothermal continuous stirred tank reactor 150
7.5 Conclusions and future work 151
7.A Interpretation of SISO cepstrum in terms of poles and zeros . . 153

8 The MIMO weighted cepstral distance 155
8.1 Relation to model parameters 156
8.2 Relation to subspace angles 157
8.3 International electricity markets 159
8.3.1 Data & objective 160
8.3.2 Clustering results 162
8.3.3 Comparison 164

9 Conclusion 167
9.1 Impact on other time series analysis applications 168
9.2 Future work 169

Bibliography 173

List of publications 181

List of Figures

1.1 A damped sine, cosine and random Gaussian noise signal. We have $y_s = \sin(10t)\mathrm{e}^{-0.05t}$, $y_s = \cos(10t)\mathrm{e}^{-0.05t}$, and $y_r = w(t)\mathrm{e}^{-0.05t}$, with $w(t)$ a white noise signal of mean 0 and standard deviation $\sigma_w = 0.7071$. The signal was sampled for 11 seconds (only part of it is shown), with a sampling timestep of 0.01 second. They serve as a toy problem to show some of the limitations of data-driven approaches to assess time series similarity. 9

1.2 The *power cepstrum* of a signal, $y(k)$, can be computed by calculating the Fast Fourier Transform, $\mathcal{F}$, of the signal, squaring its magnitude and dividing by N, the length of the signal, then taking the logarithm, and finally applying the Inverse Fast Fourier Transform, $\mathcal{F}^{-1}$. This ensures that the convolution in time domain changes into a multiplication (through the Fourier transform) in frequency domain. The logarithm takes this multiplication to an addition, which remains an addition through the inverse Fourier transform. 13

1.3 Scientific programme of this thesis, with references to the relevant Chapters for each node. Numerical illustrations permeate the different Chapters, corroborating the results on known, synthetic data, and showing their practical applicability on real-life problems. The red line between MIMO systems and Subspace angles indicates a missing link, which we will discuss in Chapter 8. An overview of what the different nodes entail is given in Section 1.4. 15

2.1 The *power cepstrum* of a signal, $y(k)$, can be computed by calculating the Fast Fourier Transform of the signal, $\mathcal{F}$, squaring its magnitude and dividing by N, the length of the signal, then taking the logarithm, and finally applying the Inverse Fast Fourier Transform, $\mathcal{F}^{-1}$. This ensures that the convolution in time domain changes into a multiplication (through the Fourier transform) in frequency domain. The logarithm takes this multiplication to an addition, which remains an addition through the inverse Fourier transform. For a more detailed discussion considering the computational issues, see Section 2.4. 21
2.2 The convolutional structure in the time domains changes into an additive one in the cepstrum domain. 22
2.3 The first five positive and negative complex cepstrum coefficients of respectively a stable system (poles at 0.9, 0.7, 0.4, blue line), an unstable system (poles at 1/0.9, 1/0.7, 1/0.4, red line) and a mixed system (poles at 1/0.9, 0.7, 1/0.4, yellow line). The zeroth cepstrum coefficient has been artificially set to 0 in all three cases to improve readability of the graph. This coefficient can take on large (positive or negative) values, but is not important for the discussion here. 27
2.4 Part of the phase of the Fourier transform of an output signal y. The principal values (shown in blue dots) are constrained to be between $-\pi$ and π. To compute the complex cepstrum, we *unwrap* these principal values to obtain samples (shown by the red circles) from the continuous phase curve (an estimate of which is fitted through the samples with a red line). Where the dashed line between blue dots is interrupted, a multiple of 2π was added or subtracted to keep values between $-\pi$ and π. . . 30
2.5 Given two subspaces V and W, we define the principal angles between them by first choosing unit vectors, called principal directions, $v_1 \in V$ and $w_1 \in W$, such that $\theta_1 = \angle(v_1, w_1)$, the angle between the two vectors, is minimised (in this case, it is 0). We then iteratively find new principal directions $v_k \in V$ and $w_k \in W$ in V and W, perpendicular to all previous principal directions, such that the angle $\theta_k = \angle(v_k, w_k)$ between them is minimised. These angles θ_k are the principal angles between subspaces V and W and can be calculated in general from an eigenvalue problem. 38

3.1 Electric circuit that was used for the experiments. Two sets, S_1 and S_2, of values were chosen for the components, namely $S_1 = \{R = 100\Omega, L_1 = 60\text{H}, L_2 = 20\text{H}, C = 50\text{F}\}$ and $S_2 = \{R = 100\Omega, L_1 = 160\text{H}, L_2 = 200\text{H}, C = 75\text{F}\}$. These two electrical circuits were used to perform the simulations in Section 3.4. 57

3.2 Performance of the different clustering algorithms, as measured by the ARI. For each time series length, shown on the x-axis, the average ARI over 100 experiments of finding 2 clusters in 400 time series is depicted as the height of the bar. The error bars show the standard deviation for the performance on these 100 experiments. Note that the Euclidean, DTW LB and cepstral distance have an ARI of 0, i.e., they amount to random guessing. The extended cepstral distance performs best for all series lengths. The model based distances were given a wrong model order, but still give good performance for longer time series. 58

3.3 Execution time of the different clustering algorithms, measured in seconds. For each time series length, shown on the x-axis, the average time over 100 experiments of finding 2 clusters in 400 time series is depicted as the height of the bar. The error bars show the standard deviation for the execution time on these 100 experiments. Note that the y-axis has logarithmic scale. The extended cepstral distance remains several orders of magnitudes faster than the model-based distances. Note that DTW LB quickly becomes the computationally most expensive technique. The Euclidean distance is always fastest. 59

3.4 On the left, the performance is shown of the different raw data distance measures, as measured by the Adjusted Rand Index (ARI), in the case of white noise as an input, and time series of length 2^{10}. Here, the average over 100 experiments with 400 output signals is shown. Note that the original cepstral distance now shows the same performance as the extended one. On the right, results of an experiment where we provided the system identification step with the correct orders of the models are shown. Here, we calculated an average over 100 experiments with 40 output signals, to reduce computation time. Again, we simulated time series of length 2^{10}. The model-based distances now show better performance. 61

3.5 Inductance profile of the non-linear, saturating inductor considered in this Appendix. The inductor has a maximum value for the inductance, L, which it reaches for currents between $-b$ and b. At $\pm b$, the inductor starts saturating, and it is saturated completely at $\pm a$. When the saturation is hit, the inductance goes to 1. 64

4.1 We will study the extended weighted cepstral distance from Equation (3.11), and extend the equivalences that were proven for the stochastic case in this Chapter. We will extend the interpretation of the distance in terms of poles and zeros in Theorem 5. A kernel function on subspace angles is defined in Subsection 4.2.2. Using the equivalences above, we relate the weighted cepstral distance and subspace angles kernel in Theorem 6. A novel way to relate these angles to input and output signals is given in Theorem 7, and a computational way of finding them is given in 4.3.2. 69

4.2 The difference in input and output behaviour between the stochastic and the deterministic case. In the stochastic case, in 4.2a, the two systems are driven by the same input sequence. The deterministic case allows for different inputs for the two systems, as in 4.2b. 73

4.3 We generate signals for system H_i by applying a white noise signal, $w_i(k)$ to a random input system H_{u_i}, generating an input $u_i(k)$. This input is then fed to the system H_i, to generate an output $y_i(k)$. 78

4.4 Convergence behaviour for the squared cosines of the subspace angles between the systems in Equation (4.35), using the algorithm presented in Subsection 4.3.2, for different signal lengths, N_l. The amount of rows of the data Hankel matrices was chosen $i = \lfloor \frac{N_l}{10} \rfloor$. The black horizontal lines represent the estimates of the true values of the squared cosines of the angles, as calculated through the Vandermonde matrices. We see that, from signals of length about 300, the estimates have stabilized. 79

4.5 Probability density functions of the computed weighted cepstral distance for 10000 repetitions of the experiment in Subsection 4.5, for different values of `NFFT`. The other parameters are set at $\texttt{cutoff} = \min(\texttt{NFFT}/2 - 1, 100)$, $\texttt{inputOrder} = 1$. The black vertical line is at 6.5352, the theoretical value calculated via the subspace angles kernel. Too small values for `NFFT` results in a bias, and overestimation of the distance. Starting from about $\texttt{NFFT} = 2^{11}$, no further change in the distribution occurs, indicating an adequate choice of `NFFT`. 88

4.6 Probability density functions of the computed weighted cepstral distance for 10000 repetitions of the experiment in Subsection 4.5, for different values of `cutoff`. The other parameters are set as $\texttt{NFFT} = 2^{11}$, $\texttt{inputOrder} = 1$. The black vertical line is at 6.5352, the theoretical value calculated via the subspace angles kernel. We need `cutoff` large enough for the infinite sum in the distance calculation to converge, but small enough to avoid numerical errors to dominate. A good choice for this experiment is $\texttt{cutoff} = 100$. 89
4.7 Probability density functions of the computed weighted cepstral distance for 10000 repetitions of the experiment in Subsection 4.5, for different values of `inputOrder`. The other parameters are set as $\texttt{NFFT} = 2^{11}$, $\texttt{cutoff} = \min(\texttt{NFFT}/2 - 1, 100)$. The black vertical line is at 6.5352, the theoretical value calculated via the subspace angles kernel. For larger and larger orders of the input process, the deterministic weighted cepstral distance becomes a worse and worse estimate of the subspace angles kernel. 90

5.1 General summary of a cepstral clustering approach. We discussed some notation on input and output signals in Section 2.1. Computing the power cepstrum was discussed in Section 2.4, while the choice of weighting for the zero-th cepstrum coefficient was discussed in Subsection 2.5.2. In this Chapter, we will discuss the k-means algorithm in Subsection 5.1.1. The different nodes inside the dotted box constitute the cepstral k-means algorithm, discussed in Subsection 5.1.2. Subsection 5.1.3 will give an interpretation to the clustering results, while Section 5.2 will explain windowing of time series, and some issues related to that. 94
5.2 A block diagram of the systems giving rise to the signals associated with the power cepstra in Equation (5.11). 101
5.3 Examples of the cylinder (Equation (5.21)), bell (Equation (5.22)) and funnel (Equation (5.23)) signals, first introduced in [69]. . 105
5.4 A concatenated series of 12 random instances of CBF-signals. We do subsequence clustering on this dataset. 105
5.5 Traditional k-means clustering on windows from the time series in Figure 5.4, alongside the ground truth. We see the results from the clustering algorithm are quite random, and certainly do not coincide with the ground truth at all. 106
5.6 Cluster centers for the k-means algorithm, as in Figure 5.5. Note that the cluster centers converge to sine waves, maximally differing in phase. 106

5.7 Cepstral k-means clustering results for a time series consisting of concatenated cylinder and bell signals. This algorithm almost perfectly returns the correct clusters. 108
5.8 Average of the signals present in the two clusters from Figure 5.7. In contrast with the traditional k-means method, these averages are no longer maximally spaced sine waves. 108
5.9 Cluster centers of the cepstral k-means clustering in Figure 5.7. These are the averages over all w_i^c from Equation (5.24). 109
5.10 Cepstral k-means clustering results. For each window number, the assigned cluster label is marked. The blue rectangles span all windows containing the error log. There are seven such windows per error. Each error log is preceded by some windows corresponding to cluster 0, with the two first errors close together in time, and following one period of anomalous behavior. 112
5.11 The same experiment as in Figure 5.10, but now working on only the temperature data, doing k-means clustering on $\sqrt{k}c_{T_{tot},i}(k)$. The clustering algorithm now misses most of the anomalous behavior. 112
5.12 The same experiment as in Figure 5.10, but now working with the Euclidean distance on the total temperature data. The clustering algorithm does not separate the normal and faulty behavior. . . 113

6.1 The model identified via the above described cepstral system identification procedure closely resembles the original system $H(z)$. The original poles are $z_1 = 0.5$, $z_2 = 0.5i$, and $z_3 = -0.5i$. 125
6.2 Both the cepstral and the least-squares procedure are applied 100 times to different input-output data of the same length coming from $H(z)$ in Equation (6.25) with white Gaussian input. The solid line shows how the mean of the $\mathcal{H}_2$-error decreases for longer data lengths, while the dashed lines represent the standard deviation. The cepstral system identification procedure seems to cope better with shorter time series than its least-squares counterpart. 126
6.3 The figure above visualizes the influence of the signal-to-noise ratio (SNR) on the identification results. For high SNRs, the identified transfer function behaves almost exactly as the original one. Down to 10 dB, the results remain quite satisfactory. However, when the SNR drops below 10 dB, the cepstral identification procedure fails to identify the system. 127
6.4 Two output signals, which are realizations of systems $H_1(z)$ in Equation (6.25) and $H_2(z)$ in Equation (6.28), respectively, with Gaussian white noise as inputs. 129

6.5 Results of cepstral k-means. The first 100 signals (0-99) are outputs of $H_1(z)$ in Equation (6.25), the last 100 belong to system $H_2(z)$ in Equation (6.28). Cepstral clustering separates them perfectly in two clusters, each representing one of the two systems. Examples of signals belonging to both clusters are shown in Figure 6.4. 129
6.6 Power spectral density estimates of some of the signals in $y_{1,i}$, the output signals of system 1. There is some variance in these estimates. 130
6.7 Power spectral density (PSD) estimates for various quantities, compared to the actual PSD. All of these results are for the output signals of system $H_1(z)$, in Equation (6.25). The geometric average PSD is the geometric average over the PSD's belonging to all the output signals of system $H_1(z)$ (for this plot, we identified a model for each signal, using the cepstral system identification proposed in this Chapter, computed their power spectral densities, and took the geometric average). The cluster center PSD is the one belonging to the cepstral cluster center. Note that the latter is equal to the geometric average, illustrating Equation (5.9), and they are both good estimators of the actual power spectral density. The Euclidean average PSD is the power spectral density belonging to the euclidean average over all signals in cluster 1. Note that this deviates from the actual PSD, showing the merit of cepstral clustering over traditional approaches, for the purpose of finding average dynamics. 131
6.8 This figure visualizes Yule's historic sunspot numbers before and after detrending. 132
6.9 The identification of Yule's historic sunspot numbers via cepstral system identification and a least-squares approach. Both methods yield a close match to Yule's model. 133
6.10 We reran the estimation experiment, but now without any data preprocessing. The data used here is the data as it was made available by Wolf and Wolfer. When we do not detrend the sunspot number data before identification, the least-squares approach fails completely to identify the system. However, the cepstrum results in a satisfactory result that captures most of the system dynamics. 133

6.11 The three-story test setup of which we monitor its structural health, consists of an aluminium frame and four aluminium floor plates, connected through bolted connections. In undamaged state, all bolted connections are tightened properly. However, to simulate structural damage, we loosen or remove some bolts. At its bottom, a shaker vibrates the structure and several accelerometers measure the resulting movements of the different aluminium parts. 134
6.12 The coefficients of the third order autoregressive model fitted on the output of one accelerometer attached to the excited three-story frame clearly indicate when the frame contains some structural damage. Results of two experiments without, and one experiment with structural damage are shown. Especially the second and third damaged autoregressive coefficients differ clearly from their undamaged counterparts. 136
6.13 Third autoregressive coefficient for each window, now obtained with least-squares estimation. The differences of the autoregressive coefficients are less distinctive when we apply the least-squares approach instead of the cepstrum. We suspect that the length of the non-overlapping windows is the main culprit of this behaviour, as this problem seems to disappear with longer window sizes. . 137

7.1 The theoretical cepstrum from Equation (7.15) and the estimate presented in Subsection 7.3.3 for the synthetic data from Subsection 7.4.1. We see that the computational cepstrum is a very good estimate of the exact one. The differences (for $k \neq 0$) are of order 10^{-4}. 149
7.2 The cepstrum coefficients for input, output and process. We show both normal and faulty operating behaviour, with (CL) and without (OL) the controller. The cepstral coefficients for the process in (c) clearly show different fingerprints for the different cases. The normal operating behaviour changes only slightly when turning on the controller, capturing the extra dynamics of the controller. However, faulty operating behaviour results in a deviation from the normal operating behaviour, both in OL and, notably, in CL, capturing faults hidden by the controller. We can see the different contributions to this process cepstrum (see Equation (7.24)) in (a) and (b): (a) shows the change in the input dynamics in the faulty regime, when the controller starts compensating. (b) shows the change in the output in the faulty regime when the controller is turned off. 152

8.1 A multiple-input multiple-output (MIMO) system, with transfer function matrix H, m inputs and m outputs. The input vector at any point in time consists of m scalar components: $u(k) = (u^{(1)}(k), u^{(2)}(k), \ldots, u^{(m)}(k))^\intercal$, and analogously for the output $y(k) = (y^{(1)}(k), y^{(2)}(k), \ldots, y^{(m)}(k))^\intercal$. 156
8.2 The results of clustering input and output time series coming from the international electricity markets, using k-means clustering on the weighted cepstral windows from Equation (8.10). For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. This is the straightforward extension of the deterministic weighted cepstral distance to the rectangular MIMO case. The step between windows is one day. Note that we clearly see seasonal patterns in the results, with clusters 0, 1 and 2 clearly belonging to summer, and clusters 6, 7 and 9 to winter. 163
8.3 The results of clustering input and output time series coming from the international electricity markets, using k-means clustering on the weighted cepstral windows from Equation (8.10), with the $u_i(k)$ omitted. For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. While some structure is still seen in the results, it is less straightforward to interpret the different clusters. 163
8.4 The results of clustering output time series coming from the international electricity markets, using k-means clustering on the concatenated output signals in Equation (8.11). For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. These results do not inspire a lot of confidence from a dynamical point of view. Consecutive windows have a lot of overlap, and thus exhibit similar dynamics. However, we see that consecutive windows are often in different clusters, when using this approach. 165
8.5 The results of clustering input and output time series coming from the international electricity markets, using k-means clustering on the concatenated output and input signals. For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. No clear improvement over Figure 8.4 is seen. 165

List of Tables

1.1 Values for the Euclidean distance and the model norms for the different pairs of signals. It is clear that the Euclidean distance does not discriminate between the different signals. The model norms identify the sine and cosine as being similar, while situating the sine and cosine wave about equally far apart from the Gaussian noise. 12

1.2 Median, mean and standard deviation of the different signals. These statistics of the sine wave, the cosine wave and the Gaussian noise signal, are all very close to each other, making them unsuitable features to devise a distance measure on. 12

1.3 Values of the weighted cepstral distance for the different pairs of signals in Figure 1.1. Compared to the distances in Table 1.1, we see that the weighted cepstral distance mimicks a model norm, putting the sine and cosine waves two orders of magnitude closer together than either the sine or the cosine to the Gaussian noise sequence. 14

3.1 ARI for the non-linear clustering experiments between two circuits, conforming to the topology of Figure 3.1, with parameters $S_1 = \{R = 100\Omega, L_1 = 60\text{H}, L_2 = 20\text{H}, C = 50\text{F}\}$ and $S_2 = \{R = 100\Omega, L_1 = 160\text{H}, L_2 = 200\text{H}, C = 75\text{F}\}$, respectively. The inductors start saturating at 3A and are completely saturated at 5A. We generated 40 input signals, simulated both systems with all these signals, and clustered the resulting input-output signals. These ARI's are for an experiment with the random number generator set at `rng('default')`. . . 65

Chapter 1

Introduction

Analyzing data that is correlated through time is a challenging problem, spanning many scientific fields (where it has been researched under many different names, e.g., time series analysis, signal processing, data stream mining, ...), using tools from many different disciplines (statistics, linear algebra, systems theory, ...), with a myriad of applications in many industries (energy, process industry, medicine, ...). It is, then, no surprise that many papers, monographs, and other academic and industrial works have been produced on the subject, giving rise to a tangle of different branches and subbranches within the field of time series analysis.

The problem we are interested in here is that of time series clustering: partitioning a set of signals in different subsets. Such a clustering algorithm consists of three main parts:

Feature selection We reduce the time series specific characteristics, or *features* of the time series, e.g. mean, maximum, derivatives, ... We can also opt to leave the time series unaltered, the raw data points are then the features considered.

Distance measure The (dis)similarity between two sets of features from time series has to be quantified.

Cluster assignment criterion We have to decide, based on the similarity between all the time series in the set, which cluster any individual time series should belong to.

In this thesis, we will mainly focus on the second part: that of defining a sensible distance measure. To do so, it is important to understand that a time series is not just a vector of values, but that these values exhibit correlation throughout time, which can often be captured in the framework of dynamical systems. Exploiting these correlations and dynamics is of paramount importance in defining similarity between time series.

More precisely, we focus on the weighted cepstral distance on signals coming from dynamical systems [19, 54]. This distance measure combines insights from different scientific fields (mainly systems theory and machine learning) and deals with some of the issues arising in traditional techniques (as we will illustrate with a very simple example in Section 1.3). While this measure was first introduced in [54], and subsequently fleshed out in [19], it is clear that the scope of these first endeavors was constrained by some of the technical assumptions made on the signal.

We will extend the weighted cepstral distance measure to relax some of the theoretical assumptions, to make it applicable in a more general class of problems. Specifically, we will

- extend the weighted cepstral distance to signals coming from systems with both inputs and outputs, based on the dynamics of the underlying models (Chapter 3),
- give system theoretical and geometric interpretations to this distance measure, allowing to compute differences between dynamical models without having to identify them (Chapter 4),
- introduce a novel clustering algorithm based on the weighted cepstral distance, showing both theoretically and through numerical examples that the resulting clusters indeed group time series together based on their underlying dynamics; we also show this clustering algorithm solves the open problem of time series subsequence clustering (Chapter 5),
- develop a system identification algorithm, that can estimate all-pole models from a cepstrum, with a computational complexity independent of the length of the cepstrum, allowing us to extremely efficiently estimate representative dynamics of a cluster (Chapter 6),
- extend the notion of the cepstrum to the case of multiple-input multiple-output (MIMO) systems, allowing us to compute a MIMO version of the distance measure (Chapter 7),
- extend the system theoretical interpretation of this distance to the MIMO case (Chapter 8).

In this way, we develop a rigorous mathematical framework for time series clustering based on underlying dynamics of signals. Such a formal framework for this type of unsupervised and explorative analysis is sorely needed to improve results in relevant, impactful and large-scale real-life applications. We focus on three key points.

Interpretability Since time series clustering is an explorative analysis, we cannot assume much previous knowledge about the data under consideration. A rigorous framework to understand and interpret the clustering algorithm allows us to assess credibility of the analysis. We develop a clear mathematical interpretation of its results, exploiting the power of dynamical systems theory.

Usability To make the developed techniques usable, we should relax assumptions on the available signals, and minimize human intervention or expert knowledge as much as possible. This means, among other things, that we want to be able to handle signals with as general underlying dynamics as possible, without having to explicitly estimate a model for each signal. We want our techniques to be data-driven.

Computability Time series are high-dimensional data objects. Signal collection is growing ever more ubiquitous (both in industrial processes, with sensors on machinery, and on the level of the individual, with wearables and handheld devices collecting data), and time series clustering is often a very large scale endeavour, both in the number of signals and the length of individual time series. We want the developed methods to be computationally very efficient. For almost all examples, numerical illustrations and applications, the code is provided online[1].

These will be the main goals of the techniques developed in this thesis. We explain how the different parts of the thesis attain them in Section 1.4.

[1] `https://github.com/Olauwers`

1.1 The weighted cepstral distance: a hybrid approach

As mentioned, we focus on the weighted cepstral distance. In this Section, we position it within the field of time series analysis, as a hybrid method in between data-driven and model-based approaches. The claim developed in this thesis is that the weighted cepstral distance is therefore the go-to time series (dis)similarity measure in applications where the underlying dynamics of the problem are important.

1.1.1 Time series analysis

Time series analysis is an important and highly relevant topic in many fields, in engineering as well as economics, mathematics, data science and many other disciplines. Ever growing amounts of sensors are producing ever larger amounts of signals, with better time resolution, in classic engineering applications as machinery monitoring (e.g. wind turbine anomaly detection [38]), but also in individuals' homes with smart grid [82] and smart home applications [68], and more and more even on the level of the individual, where wearables and smartphones can track all movement and activities performed by their wearers [83].

With increasing computational power, we now also arrive at much broader use and analysis of this data. It is therefore no surprise that the past decade has generated much scientific interest on this particular subject [22]. Historically, many disciplines have occupied themselves with time series analysis. In engineering, signal processing and systems and control theory revolve around the analysis of signal data, viewing them as representative of some underlying model, containing some unknown dynamics. Economics has the subfield of econometrics, which is an alternative approach with a similar purpose, mainly to model and predict the next few points in a time series. In social sciences, data coming from the aforementioned wearables can generate insight in group dynamics, or reveal what exogenous variables have an impact on social differences, through regression analysis. Many applications exist in medicine as well, where EEG and ECG data analysis is helpful in diagnosis and outcome prediction.

More recently, with the rise of *data science* and machine learning, the different disciplines relevant to this topic are coalescing more and more, resulting in a cross-over of techniques from one discipline to the other. The taxonomy of time series analysis techniques should therefore not be done according to

applicational subfield anymore, but focus more on the type of method used, the particular technical challenge and the type of data.

Traditionally, there have been two ways of dealing with time series data. On the one hand there is the *data-driven* approach, where the time series is treated as just a vector of data points. These are the more statistical approaches of time series analysis. We can view the data points in the series as coming from specific distributions, calculate moments such as the mean, or standard deviation of the series, define application-specific relevant features (e.g. amount of peaks, time to reach a certain threshold), compare the shapes of multiple series, ... These techniques are very powerful, as they are often computationally feasible for gargantuan data sets, allow a lot of flexibility, and in many cases can be almost completely automated. However, most often, these methods are *black-box*, and do not readily allow for a clear and rigorous interpretation of their inner workings, let alone their results.

Alternatively, there is the model-based approach, where the relevant object under consideration is the dynamical model generating a time series, rather than the time series itself. A time series is considered as being an output of an underlying dynamical model or system. While the dynamics of this system may not be accessible directly, the output signal can be used to study properties of the system, and allows us to estimate or identify the underlying model. This approach allows us to exploit the full, powerful theoretical framework behind dynamical system. When done correctly, it can produce excellent results. However, it is also very costly, time-consuming, and requires an extraordinary amount of expert domain knowledge.

> In this thesis, we take a hybrid approach to time series analysis, combining the data-driven concept of the cepstrum with a model-based interpretation in terms of dynamical systems theory. In this way, we retain usability and computability from data-driven methods, yet achieve the interpretability of a full-on model-based approach.
>
> We will use these methods to perform unsupervised and exploratory data analysis tasks, where the merit of interpretability can make the difference between failure and success. Specifically, we focus on the use of similarity measures in time series clustering.

1.1.2 Similarity measures

Before we dive into the specifics of time series applications, let us first formally define a distance measure or metric:

Definition 1. A metric, d, on a set, X, is a function

$$d : X \times X \to [0, \infty), \tag{1.1}$$

that fulfills, $\forall x, y, z \in X$, the following four conditions:

1. $d(x, y) \geq 0$,
2. $d(x, y) = 0 \iff x = y$,
3. $d(x, y) = d(y, x)$,
4. $d(x, y) \leq d(x, z) + d(z, y)$.

In what follows, the set X will consist of time series objects.

One of the main issues with time series data is that of dimensionality. When a time series is treated as an ordinary vector, where the different time points are just viewed as features of the full data object, the vector dimension is as large as the amount of timepoints we have sampled in the series. However, many vector distances, such as the classical Euclidean distance or L_2-norm, exhibit fundamental issues in high-dimensional spaces, measuring all objects to be approximately equally far apart.

High-dimensional spaces, like the ones that time series live in, exhibit some counter-intuitive geometrical properties. For example, when data samples are distributed randomly and uniformly within a hypercube in a high-dimensional space, the odds are overwhelmingly in favor of these samples falling in one of the corners of the cube. Data in high-dimensional spaces therefore lives mainly near the surface of volumes, and it becomes notoriously difficult to define good distance measures between them. This is known as the *curse of dimensionality*. A clear overview of these phenomena, relevant to the topics discussed in this text, is given in [79].

Even though they are still very commonly used in the literature, purely shape-based distances (like the widely used Euclidean distance) will therefore rapidly fail as time series grow longer, and are inappropriate for time series clustering, as shown in [45, 62]. We thus want to go beyond naively comparing shapes of time series, and compute the distance in some more sensible way. We discuss the benefits and pitfalls of the data-driven and model-based approaches to time series distance measures.

In the data-driven approach, we can resolve the curse of dimensionality by distilling the high-dimensional time series in a lower-dimensional set of features.

These could be the average of the time series, its standard deviation, peak-to-peak-distance, ... We then represent the time series by its set of features and perform a classical clustering algorithm on these feature vectors.

This approach could work well for certain applications, when it is a priori very clear what the interesting features of the time series are. However, for more complex problems, with more complex features analyzed by non-linear black-box models, this direct interpretation is lost. As we reduce the information from the whole signal to some small set of features, it also becomes difficult to answer any other questions about the signals with the same data.

Because of their ease of use and computation, these feature-based distances are very widely employed, especially in black-box solutions, where the features are often automatically generated from raw data [26]. Features obtained in such a way often are quite opaque and uninterpretable [89].

Model-based distances, on the other hand, start by estimating a model for the different signals, and define model norms to compute distances. While it now seems we have just shifted the discussion from defining measures on signals to defining them on the underlying models, there is a rich interpretative framework associated with this approach, known as systems theory, and a good overview of system identification methods is given in [49].

If we manage to find a good model, its parameters can give us a lot of insight in what is going on behind the scenes, and allow us to look at differences in dynamics between signals coming from different systems. While in this case, we have more insight in what exactly a given similarity measure is measuring, finding a good model is a very complicated problem in itself, taking up a lot of resources and manpower.

An example of a model-based application, that will return later on in the thesis is that of structural damage monitoring using autoregressive models [10, 24, 72]

1.1.3 Positioning of this thesis

In this thesis, we take a hybrid approach when it comes to the model-free/data-driven distinction: while all techniques are completely data-driven, and require just a numerical computation on the data vector without relying on a system identification step, we still view the time series as the output of some underlying system, and can give some insight and interpretation to the results we get. By doing so, we inherit the advantages of both approaches: the techniques presented here are computationally cheap, can be automated, but allow for

interpretation of results, and therefore reach the three key goals we set out in the scientific programme.

We will focus our attention on unsupervised learning applications, and show the importance of using interpretable techniques to do so. As there are no labels to force the algorithm to take certain decisions, and often not even to check the results with afterwards, a good understanding of what the algorithm does and what decisions it will make is paramount.

More precisely, we will work mainly on time series clustering, with a specific interest in how to quantify similarity between two time series. We will explore a data-driven similarity measure that, nevertheless, allows an interpretation in terms of dynamical systems. Several applications will show the relevance of this approach.

1.2 Industrial relevance

While this work is first and foremost an academic endeavor, we should not forget its industrial and economic relevance. Indeed, as time series research in the past decade has been sparked by increasing amounts of available data, it has been mainly industry that collected said data. Big industrial facilities have millions of sensors on their equipment [53], and they now even face the problem of having too much information: when something goes awry, not one alarm is triggered, but thousands of them are. Assistance in analyzing such signals is a big industry demand, giving rise to companies like TrendMiner [74].

Smart grids and smart homes [43, 75] need techniques that can handle signals coming from many millions of homes, and even millions of appliances within homes [14] at once. An estimated 3.5 billion people nowadays have a smart phone [40], that tracks them everywhere they go. Big multinational internet companies like Google (Alphabet) and Facebook capitalize and depend on this type of data, so the industrial stakes are very high.

While we focus in the rest of the text mainly on the scientific contributions, throughout this PhD mandate we have had collaborations with many companies. We have worked together with TrendMiner (formerly DSquare), Eventigrate and AB InBev, either directly or through master theses.

Finally, we had a large four-month collaboration with the R&D-deparment of Électricité de France (EDF), a multinational electric utility company, based in Paris, France, with a portfolio containing generation capacity all across the globe. In Chapter 9, we will come back in more detail to this industrial application.

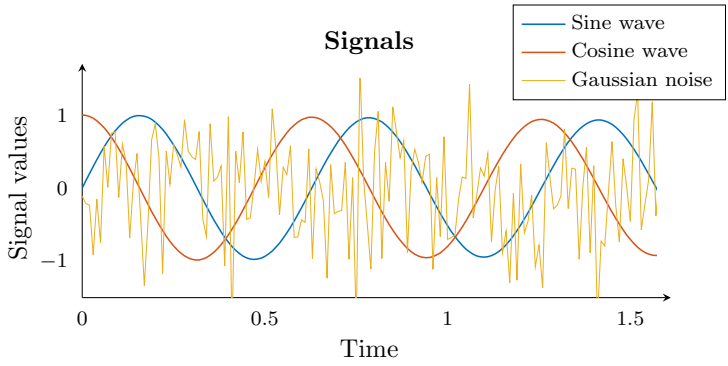


Figure 1.1: A damped sine, cosine and random Gaussian noise signal. We have $y_s = \sin(10t)\mathrm{e}^{-0.05t}$, $y_s = \cos(10t)\mathrm{e}^{-0.05t}$, and $y_r = w(t)\mathrm{e}^{-0.05t}$, with $w(t)$ a white noise signal of mean 0 and standard deviation $\sigma_w = 0.7071$. The signal was sampled for 11 seconds (only part of it is shown), with a sampling timestep of 0.01 second. They serve as a toy problem to show some of the limitations of data-driven approaches to assess time series similarity.

1.3 A simple example

We illustrate the failure of traditional methods, and the benefits offered by the weighted cepstral distance, on a very simple toy problem that exposes the challenges of time series clustering.

Figure 1.1 shows three signals: a sine wave, a cosine wave, and a white Gaussian noise signal, with mean 0 and a standard deviation of $\sigma = 0.7071$. These signals are all multiplied by the same exponential damping[2] factor of $e^{-0.05t}$.

The task is now to calculate how similar each pair of signals is.

1.3.1 Commonly used distance measures

Let us introduce some distance measures.

[2]The damping factor ensures the underlying models are stable and deliver bounded outputs, which is a relevant assumption to make when working with model norms. The next few chapters will repeat this in more detail.

The standard Euclidean distance, which is very often used in time series analysis, between two time series, y_1 and y_2, of length N, is defined as

$$d_e(y_1, y_2) = \sqrt{\sum_{k=0}^{N-1} (y_1(k) - y_2(k))^2}, \tag{1.2}$$

treating the time series as vectors in a Euclidean space.

The Euclidean distance between two real vectors, $v \in \mathbb{R}^N$ and $w \in \mathbb{R}^N$, can also be interpreted in terms of the Euclidean norm of v and w, as

$$d_e^2(v, w) = ||v||^2 + ||w||^2 - 2||v||||w|| \cos(\theta_{v,w}), \tag{1.3}$$

with θ the angle between the vectors.

For infinitely long signals, a white noise vector is orthogonal to any other signal. The sine and cosine are orthogonal to each other. The angles between any combination of two of the three signals considered will then approximate $\frac{\pi}{2}$ (which would be its exact value for infinitely long sequences), and the cosines will vanish. Furthermore, by giving the random noise signal the same average and standard deviation as the sine and cosine signals, we ensured their Euclidean norms are the same in the infinite-length limit.

Computationally, we get for our finite signals

$$\begin{aligned} ||y_s|| &= 18.2616, \\ ||y_c|| &= 18.2747, \\ ||y_r|| &= 18.2652, \end{aligned} \tag{1.4}$$

in good agreement with our theoretical derivations.

We would thus expect the Euclidean distances to not discern at all between these signals, considering them all to be equally far apart, which is indeed what we get from computational results, shown in the first line of Table 1.1.

We can also take a model-based approach, and estimate an autoregressive (AR) model of order 1 from each of these signals in Matlab. This gives us pole polynomials

$$\begin{aligned} a_s(z) &= z - 0.995, \\ a_c(z) &= z - 0.995, \\ a_r(z) &= z + 0.0445, \end{aligned} \tag{1.5}$$

with $a_s(z)$ the pole polynomial of the sine wave, $a_c(z)$ that of the cosine, and $a_r(z)$ that of the random noise signal.

We can now use model norms to compare them. Define the rational polynomials $H_s(z) = \frac{1}{a_s(z)}$, $H_c(z) = \frac{1}{a_c(z)}$ and $H_r(z) = \frac{1}{a_r(z)}$. This allows us to compute model norms.

The $||\cdot||_2$-norm is defined as

$$||H||_2 = \sqrt{\frac{1}{2\pi}\int_{-\pi}^{\pi} Tr\left\{H^H(\mathrm{e}^{i\omega})H(\mathrm{e}^{i\omega})\right\} d\omega}, \tag{1.6}$$

where $Tr\{\}$ denotes the trace, the superscript $\cdot^H$ denotes the Hermitian conjugate and i denotes the imaginary unit. H is evaluated on the unit circle.

Alternatively, we can compute the $||\cdot||_\infty$-norm

$$||H||_\infty = \max_{\omega\in[0,\pi[} |H(\mathrm{e}^{i\omega})|. \tag{1.7}$$

From these norms, we can define distance measures between two signals, y_1 and y_2, with respective transfer functions H_1 and H_2, by identifying $d_{||\cdot||_2}(H_1, H_2) = ||H_1 - H_2||_2$, and analogously for $||H||_\infty$.

Theoretically, then, we would expect the model norms between the sine and cosine signal to be zero, as they exhibit the same dynamics. Moreover, the norm-based distance between either of them and the random noise signal should be large.

Computational results for the toy problem at hand are summarized in Table 1.1. The norm-based distances ($d_{||\cdot||_2}$, $d_{||\cdot||_\infty}$) make a clear distinction between the sine and cosine wave on the one hand, and the Gaussian noise on the other hand. The small deviation from zero for the norms between the sine and cosine signal has to do with the fact that we perform these computations on finite signals, and with finite precision.

From an engineering perspective, it is reasonable to say that the model-based distances give better results than the Euclidean distance. Indeed, the sine and cosine wave are in fact shifted versions of the same signal, while the Gaussian noise signal is dynamically very different. The Euclidean distance, which was based on the pure shape of the signals, completely misses these dynamics.

While the choice of distance measure can only be right or wrong with respect to the specific application, the toy problem above seems to suggest that in most engineering applications – where generative dynamics are more interesting than the shape of the exact realization of the time series – the model norm-based distances would be more appropriate than the shape-based ones. Throughout both the synthetic and real-life examples presented in the different chapters in this thesis, we will see again and again that this holds true in practice.

	d(sin,cos)	d(cos,Gauss)	d(sin,Gauss)
d_e	25.8	26.0	26.1
$d_{\|\|\cdot\|\|_2}$	$7.6 \cdot 10^{-3}$	9.967	9.972
$d_{\|\|\cdot\|\|_\infty}$	$2.16 \cdot 10^{-1}$	199.0	199.2

Table 1.1: Values for the Euclidean distance and the model norms for the different pairs of signals. It is clear that the Euclidean distance does not discriminate between the different signals. The model norms identify the sine and cosine as being similar, while situating the sine and cosine wave about equally far apart from the Gaussian noise.

	Median	Mean	Std
Sine	$3.2 \cdot 10^{-2}$	$1.4 \cdot 10^{-2}$	$5.5 \cdot 10^{-1}$
Cosine	$2.4 \cdot 10^{-3}$	$5.9 \cdot 10^{-4}$	$5.5 \cdot 10^{-1}$
Gauss	$1.2 \cdot 10^{-2}$	$1.4 \cdot 10^{-2}$	$5.5 \cdot 10^{-1}$

Table 1.2: Median, mean and standard deviation of the different signals. These statistics of the sine wave, the cosine wave and the Gaussian noise signal, are all very close to each other, making them unsuitable features to devise a distance measure on.

In Table 1.2, we show commonly used time series features: the median, mean and standard deviation for all time samples of each signal. Here again, the results do not clearly distinguish between the sine/cosine on the one hand and the Gaussian noise signal on the other.

Of course, the toy problem discussed here is just that, a toy problem, and on top of that one that has been chosen exactly to suit our needs and show the problems we want to show. However, it illustrates a class of similar issues, which arrive in real life applications, and have an important impact on results.

1.3.2 The weighted cepstral distance

Let us now look at the weighted cepstral distance, explored in [19, 54].

Starting from a time series, y, we can calculate its power cepstrum as

$$c_y(k) = \mathcal{F}^{-1}(\log |\mathcal{F}(y)|^2), \tag{1.8}$$

with $\mathcal{F}$ and $\mathcal{F}^{-1}$ the Fourier and inverse Fourier transform, respectively. This procedure is summarized in Figure 1.2.

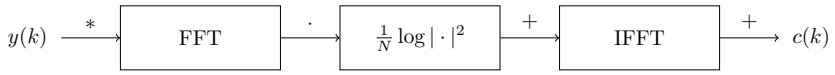


Figure 1.2: The *power cepstrum* of a signal, $y(k)$, can be computed by calculating the Fast Fourier Transform, $\mathcal{F}$, of the signal, squaring its magnitude and dividing by N, the length of the signal, then taking the logarithm, and finally applying the Inverse Fast Fourier Transform, $\mathcal{F}^{-1}$. This ensures that the convolution in time domain changes into a multiplication (through the Fourier transform) in frequency domain. The logarithm takes this multiplication to an addition, which remains an addition through the inverse Fourier transform.

Given two time series, y_1 and y_2, with respective cepstra c_{y_1} and c_{y_2}, we define the weighted cepstral distance as

$$d_c(y_1, y_2)^2 = \sum_{k=0}^{\infty} k \left(c_{y_1}(k) - c_{y_2}(k)\right)^2 . \tag{1.9}$$

It was shown [19], for Single-Input Single-Output (SISO) Linear Time Invariant (LTI) dynamical systems, driven by white noise, that this similarity measure is equivalent to a model norm. These equivalences are described in more detail in Chapter 2. In [7], this distance measure was used in a clustering application on power consumption series.

For the toy problem in Figure 1.1, we would expect that this distance measure mimicks the model norms, returning a distance of zero between the sine and cosine signal, and a non-zero distance for the other combinations. Note, however, that it is computed from the data alone, not using the AR models we identified.

We show computational results for this distance measure in Table 1.3. We see that the weighted cepstral distance indeed mimicks the results of the model norms, rather than those of the Euclidean distance. While the distance between sine and cosine is not exactly zero due to the fact that we are working with finite signals, there is a difference of two orders of magnitudes with the distances calculated between either the sine or the cosine and the random noise signal.

This toy problem shows, in an intuitive way, the motivation of considering underlying dynamics of a time series, rather than the shape of the particular series itself. We will develop a cepstral clustering framework in the rest of this thesis.

	d(sin,cos)	d(cos,Gauss)	d(sin,Gauss)
d_c	1.232	112.5	113.0

Table 1.3: Values of the weighted cepstral distance for the different pairs of signals in Figure 1.1. Compared to the distances in Table 1.1, we see that the weighted cepstral distance mimicks a model norm, putting the sine and cosine waves two orders of magnitude closer together than either the sine or the cosine to the Gaussian noise sequence.

1.4 Scientific programme

The goals set out on p. 3 translate into the scientific programme summarized in Figure 1.3. We will give an overview of what the different blocks in the diagram entail, working top to bottom, left to right. Relevant Chapters are mentioned in the Figure. All relevant preliminary concepts will be formally defined in Chapter 2.

We start from time series, which we assume to come from a linear dynamical system. Such a signal allows for a transformation, called the *power cepstrum*. It is this transformation, the coefficients it results in, and the analysis methods derived from it, that will be the object of study throughout this thesis.

The cepstrum was defined for scalar signals (i.e. where the dynamical system behind the signal produces only one output). To increase the **usability** of the methods developed, we propose an extension to vector signals (i.e. the *MIMO* case).

From the cepstrum, we can define a distance measure, the *weighted cepstral distance*, which was defined for systems with a white noise input. We will extend it to signals belonging to *deterministic systems*, and *MIMO systems*. The extension to the deterministic case allows for **interpretability**: we can find an expression for the distance measure in terms of *subspace angles* and *model parameters*. The MIMO case will be interpreted in terms of model parameters, but we lose the interpretation in terms of subspace angles.

Next, a *cepstral clustering algorithm* is defined, based on this distance measure. It will be an adaptation of k-means, a well-known clustering algorithm, that is available in most modern scientific programming languages, enhancing **computability** (further aided by the cepstral distance being computationally inexpensive).

The clustering algorithm produces clusters with *cluster centers* and, between the objects in a given cluster, *cluster variance*. This algorithm will provide

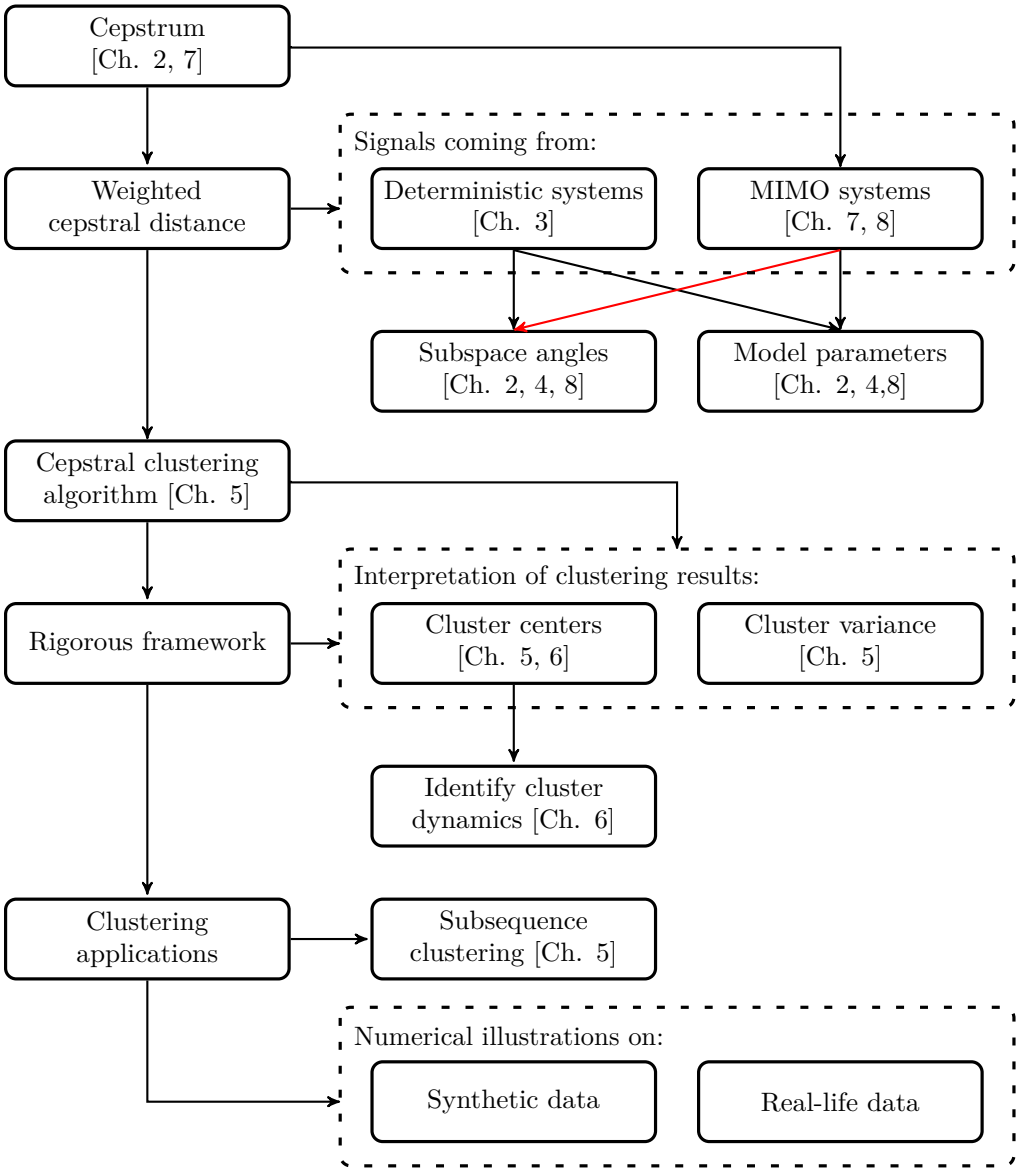


Figure 1.3: Scientific programme of this thesis, with references to the relevant Chapters for each node. Numerical illustrations permeate the different Chapters, corroborating the results on known, synthetic data, and showing their practical applicability on real-life problems. The red line between MIMO systems and Subspace angles indicates a missing link, which we will discuss in Chapter 8. An overview of what the different nodes entail is given in Section 1.4.

a *rigorous framework* for clustering time series. Indeed, conform our goal of **interpretability**, we will figure out what these results mean in a system theoretical context.

All of the above will be done without requiring any system identification step, enhancing **usability**. However, in some contexts, it might be of relevance to be able to *identify cluster dynamics*, resulting from the algorithm above. They can provide extra **interpretability**. We propose an extremely efficient novel system identification algorithm capable of doing exactly that, of computational complexity proportional to the order of the underlying model, rather than to the length of the cepstrum, reaching our goal of **computability**.

Finally, because of the framework we develop to understand our clustering techniques, we will be able to tackle *clustering applications* in a more rigorous way. We show the advantage of this framework by straightforwardly solving an open problem in machine learning: time series *subsequence clustering*, resulting in higher **usability**. We do not need to introduce any new machinery outside of the cepstral clustering algorithm we already discussed, retaining **computability**.

Throughout this thesis, the theoretical framework will be corroborated by many illustrations, examples and larger-scale applications – both on *synthetic data* and *real-life data* – to help understand the relevance of the framework in a practical setting. Most of the code that was used in these applications is made available directly online (`https://github.com/Olauwers`), exactly as it was used to obtain the results as they are described in the text. In this way, we try to apply the three goals of **interpretability**, **usability** and **computability** not only to the developed methods, but also to the way we present them.

Specifically, we provide code for all the algorithms presented in this thesis, including the cepstral k-means clustering algorithm, the computation of subspace angles between systems for different inputs, and the computation of the MIMO cepstrum.

Three papers have been published in the context of this PhD mandate, they can be found in the list of publications on p. 181.

Chapter 2

The cepstrum

In this Chapter, we give an overview of recurring notation, definitions and assumptions. We rigorously define and explore elements from dynamical systems theory, the power cepstrum and weighted cepstral distance, and the notion of subspace angles.

We show equivalences between the weighted cepstral distance and a combination of model parameters, and an equivalence between these two and a positive definite kernel computed from subspace angles between the underlying models, in the case of stochastic systems..

2.1 Dynamical systems theory

The data objects under consideration in this thesis are that of finite sequences of data points, which exhibit correlations across time. We will call these data objects signals or time series.

An output signal y is assumed to be generated by some minimal, stable and inversely stable[1], single-input single-output (SISO) linear time invariant (LTI) dynamical system, described by the discrete-time state-space model

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) + Du(k) \end{cases}, \tag{2.1}$$

[1]Subsection 2.6.2 gives some pointers on how to relax these stability assumptions.

with $x(k) \in \mathbb{R}^n$ the states of the model and $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times 1}$, $C \in \mathbb{R}^{1\times n}$ and $D \in \mathbb{R}$ system matrices, with $D \neq 0$ invertible. The parameter k denotes time and u is the input of the model, which is itself a signal, assumed to be persistently exciting (i.e. exciting all frequencies; if this was not the case, certain parts of the dynamics would not be excited, as proven in [87]).

In the time domain, the input and output signals are related as

$$y(k) = h(k) * u(k), \tag{2.2}$$

where we assume $x(0) = 0$ and $h(k)$ is the impulse response of the system. The symbol $*$ denotes the convolution, which in discrete time is given by the infinite sum

$$h(k) * u(k) = \sum_{m=-\infty}^{\infty} h(m)u(k-m). \tag{2.3}$$

For finite signals, we can approximate this sum as

$$h(k) * u(k) = \sum_{m=-M}^{M} h(m)u(k-m), \tag{2.4}$$

for a suitable M.

In the frequency domain, or z-domain, the relation between the input and output becomes, again assuming $x(0) = 0$,

$$Y(z) = H(z)U(z), \tag{2.5}$$

with $Y(z)$ and $U(z)$ the z-transforms of output and input signals respectively, $H(z)$ the z-domain *transfer function* of the system, which is the z-transform of the impulse response $h(k)$. Written in terms of the state space matrices, we have for the transfer function:

$$H(z) = D + C\left(z\mathbb{1} - A\right)^{-1} B. \tag{2.6}$$

In general, $H(z)$ is a rational polynomial, and we can write it as

$$H(z) = \frac{b(z)}{a(z)}, \tag{2.7}$$

where $a(z)$ is the pole polynomial and $b(z)$ the zero polynomial. The roots of $a(z)$ (i.e. the solutions of $a(z) = 0$) are the poles of the model, which we will write as α_i, $\forall i \in \{1, \ldots, n\}$, and equivalently the roots of $b(z)$ are known as the zeros and denoted by β_i, $\forall i \in \{1, \ldots, q\}$, with $q \leq n$.

The power spectral density, Φ_y, of an infinite-length stationary signal is formally defined as the Fourier transform, $\mathcal{F}(\cdot)$, of the autocorrelation, ρ_y, of the signal:

$$\Phi_y = \mathcal{F}(\rho_y), \tag{2.8}$$

where the autocorrelation function is defined as the expected value,

$$\rho_y(\tau) = \mathrm{E}[y(k)y(k+\tau)]. \tag{2.9}$$

For the finite signals considered in this thesis, we define the power spectral density as the square magnitude of the Fourier transform of the signal:

$$\Phi_y\left(\mathrm{e}^{i\omega}\right) = Y\left(\mathrm{e}^{i\omega}\right)\overline{Y\left(\mathrm{e}^{i\omega}\right)} = \left|Y\left(\mathrm{e}^{i\omega}\right)\right|^2, \tag{2.10}$$

where $\overline{\cdot}$ denotes the complex conjugate, and analogously $\Phi_u(\mathrm{e}^{i\omega}) = |U(\mathrm{e}^{i\omega})|^2$. Details on the differences between power spectral densities of infinite-length and finite-length signals can be found in [49].

We can define the power spectral density related to the transfer function $H(z)$:

$$\Phi_h\left(\mathrm{e}^{i\omega}\right) = H\left(\mathrm{e}^{i\omega}\right)\overline{H\left(\mathrm{e}^{i\omega}\right)} = \left|H\left(\mathrm{e}^{i\omega}\right)\right|^2. \tag{2.11}$$

For the SISO systems we consider throughout most of this thesis, we then have

$$\Phi_y\left(\mathrm{e}^{i\omega}\right) = \Phi_h\left(\mathrm{e}^{i\omega}\right)\Phi_u\left(\mathrm{e}^{i\omega}\right) \tag{2.12}$$

In this thesis, we mainly show results for all-pole or autoregressive (AR) models, represented in the z-domain as

$$Y(z) = \frac{z^n}{a(z)}U(z). \tag{2.13}$$

The pole polynomial of this model is given by

$$a(z) = z^n + \sum_{i=1}^{n-1} a_i z^i = \prod_{i=1}^{n}(z-\alpha_i). \tag{2.14}$$

This all-pole assumption is mainly made for ease of notation. In fact, with the exception of the results in Chapter 6 – where the identification technique, at present, has only been worked out for all-pole models – all results on the power cepstrum and weighted power cepstral distance in this thesis can easily be extended to models with both poles and zeros. Subsection 2.6.1 will give details on how this is done.

We define for every state-space model an observability matrix $\Gamma^i \in \mathbb{R}^{i\times n}$ as

$$\Gamma^i = \begin{pmatrix}(C)^\intercal & (CA)^\intercal & (CA^2)^\intercal & \cdots & (CA^{i-1})^\intercal\end{pmatrix}^\intercal, \tag{2.15}$$

where the $A \in \mathbb{R}^{n\times n}$ and $C \in \mathbb{R}^{1\times n}$ matrix are the matrices from Equation (2.1), and $\cdot^\intercal$ denotes the transpose.

We can invert the system in Equation (2.5), to $U(z) = H^{-1}(z)Y(z)$, and find

$$H^{-1}(z) = D^{-1} - D^{-1}C(z\mathbb{1} - A + BD^{-1}C)^{-1}BD^{-1} = \frac{a(z)}{b(z)}. \tag{2.16}$$

The inverse system to the one in Equation (2.1) is then given by

$$\begin{cases} x(k+1) = \left(A - BD^{-1}C\right)x(k) + BD^{-1}y(k) \\ u(k) = -D^{-1}Cx(k) + D^{-1}y(k) \end{cases}. \tag{2.17}$$

Later on in this thesis, we will use the observability matrix of the inverse system, $\Gamma^i_{\text{inv}} \in \mathbb{R}^{i\times n}$, defined as

$$\Gamma^i_{\text{inv}} = \begin{pmatrix} -D^{-1}C \\ -D^{-1}C\left(A - BD^{-1}C\right) \\ -D^{-1}C\left(A - BD^{-1}C\right)^2 \\ \cdots \\ -D^{-1}C\left(A - BD^{-1}C\right)^{i-1} \end{pmatrix}. \tag{2.18}$$

We thus understand a signal to have underlying dynamics in the form of a dynamical system, whether we know the system parameters explicitly or (as is most often the case) not.

2.2 The power cepstrum

The power[2] cepstrum of a signal, y, is defined as

$$c_y(k) = \mathcal{F}^{-1}\left\{\log\left|\mathcal{F}\left\{y(k)\right\}\right|^2\right\}, \tag{2.19}$$

with $\mathcal{F}^{-1}$ the inverse Fourier transform. The computational procedure is illustrated in Figure 2.1, where we employ the Fast Fourier Transform (FFT). A detailed discussion of the computational aspects will be given in Subsection 2.4.

[2]We will use the terms power cepstrum and cepstrum interchangeably. The name power cepstrum stems from the fact that it is derived from the power spectral density of a signal. There is a different notion called the complex cepstrum, which will be introduced in Subsection 2.3. We will always explicitly refer to the complex cepstrum when it is relevant.

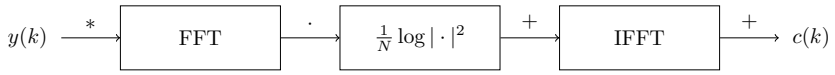


Figure 2.1: The *power cepstrum* of a signal, $y(k)$, can be computed by calculating the Fast Fourier Transform of the signal, $\mathcal{F}$, squaring its magnitude and dividing by N, the length of the signal, then taking the logarithm, and finally applying the Inverse Fast Fourier Transform, $\mathcal{F}^{-1}$. This ensures that the convolution in time domain changes into a multiplication (through the Fourier transform) in frequency domain. The logarithm takes this multiplication to an addition, which remains an addition through the inverse Fourier transform. For a more detailed discussion considering the computational issues, see Section 2.4.

This notion of power cepstrum is used in signal processing, particularly in the subfield of *homomorphic signal processing* [59]. Equation (2.19) turns the convolutional structure of the time domain, through the Fourier transform and the logarithm, into an additive structure. The inverse Fourier transform then turns this back into a transformed version of the time domain. This is known as *quefrency alanysis*, and has given rise to a lot of similar terms, such as liftering instead of filtering and rhamonics instead of harmonics [8].

In light of this, starting from the convolutional structure in time domain from Equation (2.2), it is easy to see that

$$c_y(k) = c_h(k) + c_u(k), \tag{2.20}$$

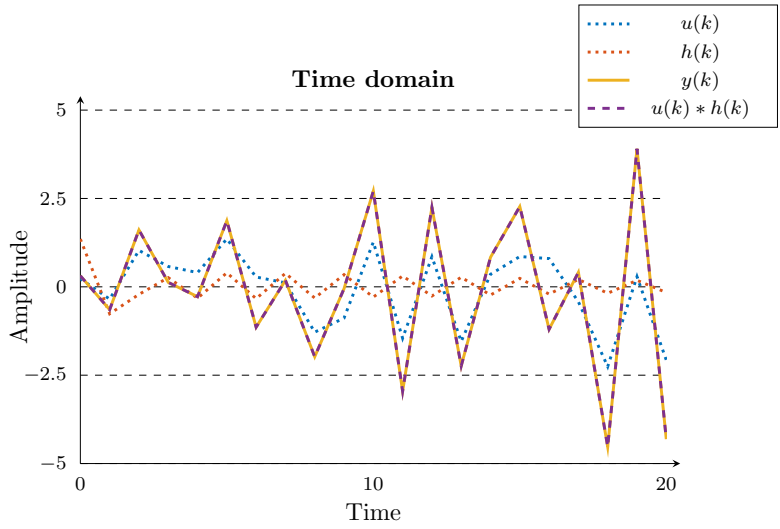
where $c_h(k)$ is the power cepstrum of the impulse response, and $c_u(k)$ is the power cepstrum of the input signal.

We show this additive structure in Figure 2.2, where we plot the input signal, output signal, and impulse response sequence of a system in the time domain, as well as their respective cepstra.

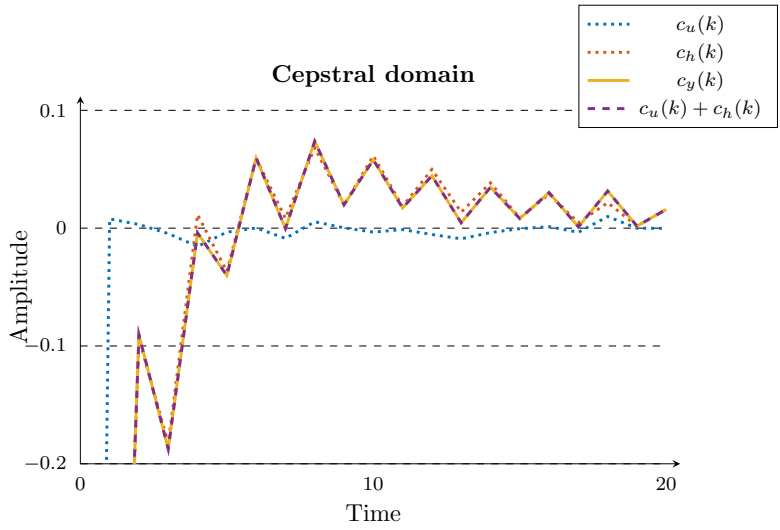
As a last remark, we will note that the power cepstrum of a white noise sequence, $w(k)$ of variance σ^2 is equal to

$$c_w(k) = \begin{cases} \log(\sigma^2) & k = 0, \\ 0 & k \neq 0. \end{cases} \tag{2.21}$$

White noise only contributes to the zero-th cepstrum coefficient, which will most often be omitted in the techniques developed in this thesis.



(a) Numerical illustration for a 10-th order system, with poles at -0.9311, -0.9077, -0.8057, -0.4462, -0.3658, 0.3897, 0.9004, 0.6469, and $0.7025 \pm 0.0705i$, and zeros at -1.1173, -0.9019, -0.7876, -0.3790, -0.1445, 0.5983, $0.7818 \pm 0.1535i$, and $0.8093 \pm 0.1096i$. The impulse response $h(k)$, the random sum-of-sinusoid, corrupted with white noise, input signal $u(k)$, and output signal $y(k)$ are plotted. The convolution $u(k) * h(k)$ is plotted to show that it coincides with $y(k)$, visualizing the convolutional structure of the time domain.



(b) The power cepstra of the signals plotted in Figure 2.2a. Here, the series $c_u(k) + c_h(k)$ is also plotted, to show that the convolution in the time domain changes to an addition in the cepstrum domain, visualizing Equation (2.20). For clarity of visualization, we restrict the amplitude from -0.2 to 0.1. The zero-th cepstrum coefficients, $c_i(0)$, take on large negative values for the signals considered.

Figure 2.2: The convolutional structure in the time domains changes into an additive one in the cepstrum domain.

2.2.1 The model power cepstrum

What is of main interest to us is what we call the model power cepstrum, $c_h(k)$, the power cepstrum of the impulse response. In terms of the power spectral density, we define the model power cepstrum of a system with transfer function $H(z)$ as

$$c_h(k) = \mathcal{F}^{-1}\left\{\log \Phi_h\left(\mathrm{e}^{i\omega}\right)\right\}. \tag{2.22}$$

Referring back to Equation (2.20), we can easily see

$$c_h(k) = c_y(k) - c_u(k). \tag{2.23}$$

This is an important statement, as in practice, we usually can measure inputs and outputs of a system, and therefore have access to $c_u(k)$ and $c_y(k)$, but not the impulse response of the system. We will therefore have to rely on this relation to compute $c_h(k)$.

The novel insight that the model cepstrum, which can be computed directly from input and output signals, is the relevant cepstrum to consider, will allow us to extend the results known for the stochastic case (as explored in-depth in [17]), to the case of non-white input signals[3].

This insight was found independently, in the case of gear and bearing faults under variable speed conditions, in [65], which was published before our paper [45]. The authors of [65] worked on the complex cepstrum, whereas we focus on the power cepstrum, and explore it in the context of the weighted cepstral distance, whereas [65] employed it mainly for filtering and diagnostic purposes.

It is the model power cepstrum that will contain information on the dynamical system between input and output. Indeed, we can directly interpret $c_h(k)$ in terms of model parameters.

2.2.2 Interpretation in poles and zeros

In this subsection, we explicitly interpret the model power cepstrum of a system in terms of poles and zeros. More precisely, the cepstrum will be interpreted as a combination of power sums over the poles and zeros. Where we usually assume stable and inversely stable, all-pole models, we will temporarily relax these assumptions, and give an expression for the model cepstrum of a general minimal, SISO LTI system. We follow the derivation from [59].

[3]Note that for a white noise input, by virtue of Equation (2.21), we have $c_h(k) = c_y(k)$, $\forall k \neq 0$. This will ensure that these extensions reduce to the known results in the stochastic case.

Starting from a transfer function, $H(z)$, with p poles α_i of magnitude smaller than 1, s poles γ_i of magnitude greater than 1, q zeros β_i of magnitude smaller than 1, and r zeros δ_i of magnitude greater than 1, and a gain g, we can express the power spectrum $\Phi_h(\mathrm{e}^{i\omega})$ as

$$\Phi_h(\mathrm{e}^{i\omega}) = g^2 \left| \frac{\prod_{j=1}^{q} \left(1 - \beta_j \mathrm{e}^{-i\omega}\right) \prod_{j=1}^{r} \left(1 - \delta_j \mathrm{e}^{-i\omega}\right)}{\prod_{j=1}^{p} \left(1 - \alpha_j \mathrm{e}^{-i\omega}\right) \prod_{j=1}^{s} \left(1 - \gamma_j \mathrm{e}^{-i\omega}\right)} \right|^2 . \tag{2.24}$$

We can allow for poles and zeros of magnitude greater than 1, because we evaluate the power spectrum on the unit circle in the complex plane. This also means we must exclude poles or zeros of magnitude exactly equal to one. Taking the logarithm, we write

$$\begin{aligned} \log(\Phi_h(\mathrm{e}^{i\omega})) = & \log(g^2) \\ & + \sum_{j=1}^{q} \left(\log\left(1 - \beta_j \mathrm{e}^{-i\omega}\right) + \log\left(1 - \overline{\beta}_j \mathrm{e}^{i\omega}\right)\right) \\ & + \sum_{j=1}^{r} \left(\log\left(1 - \delta_j \mathrm{e}^{-i\omega}\right) + \log\left(1 - \overline{\delta}_j \mathrm{e}^{i\omega}\right)\right) \\ & - \sum_{j=1}^{p} \left(\log\left(1 - \alpha_j \mathrm{e}^{-i\omega}\right) + \log\left(1 - \overline{\alpha}_j \mathrm{e}^{i\omega}\right)\right) \\ & - \sum_{j=1}^{s} \left(\log\left(1 - \gamma_j \mathrm{e}^{-i\omega}\right) + \log\left(1 - \overline{\gamma}_j \mathrm{e}^{i\omega}\right)\right) . \end{aligned} \tag{2.25}$$

Using the series expansions

$$\log(1 - x\mathrm{e}^{-i\omega}) = -\sum_{n=1}^{\infty} \frac{x^n}{n} \mathrm{e}^{-in\omega} \quad \forall |x| < 1, \tag{2.26}$$

for the stable roots, and

$$\begin{aligned} \log(1 - x\mathrm{e}^{-i\omega}) &= -\log\left(\frac{x^{-1}\mathrm{e}^{i\omega}}{x^{-1}\mathrm{e}^{i\omega} - 1}\right) \\ &= -\log\left(-x^{-1}\mathrm{e}^{i\omega}\right) - \log\left(\frac{1}{1 - x^{-1}\mathrm{e}^{i\omega}}\right) \\ &= -\log\left(-x^{-1}\mathrm{e}^{i\omega}\right) - \sum_{n=1}^{\infty} \frac{x^{-n}}{n} \mathrm{e}^{in\omega} \qquad \forall |x| > 1, \end{aligned} \tag{2.27}$$

for the unstable roots, we find

$$\begin{aligned}
&\log(\Phi_h(\mathrm{e}^{i\omega})) = 2\log(g) \\
&- \sum_{j=1}^{q}\left(\sum_{n=1}^{\infty}\frac{\beta_j^n}{n}\mathrm{e}^{-in\omega} + \sum_{n=1}^{\infty}\frac{\overline{\beta}_j^n}{n}\mathrm{e}^{in\omega}\right) \\
&- \sum_{j=1}^{r}\left(\log\left(-\overline{\delta}_j^{-1}\mathrm{e}^{-i\omega}\right) + \log\left(-\delta_j^{-1}\mathrm{e}^{i\omega}\right) + \sum_{n=1}^{\infty}\frac{\overline{\delta}_j^{-n}}{n}\mathrm{e}^{-in\omega} + \sum_{n=1}^{\infty}\frac{\delta_j^{-n}}{n}\mathrm{e}^{in\omega}\right) \\
&+ \sum_{j=1}^{p}\left(\sum_{n=1}^{\infty}\frac{\alpha_j^n}{n}\mathrm{e}^{-in\omega} + \sum_{n=1}^{\infty}\frac{\overline{\alpha}_j^n}{n}\mathrm{e}^{in\omega}\right) \\
&+ \sum_{j=1}^{s}\left(\log\left(-\overline{\gamma}_j^{-1}\mathrm{e}^{-i\omega}\right) + \log\left(-\gamma_j^{-1}\mathrm{e}^{i\omega}\right) + \sum_{n=1}^{\infty}\frac{\overline{\gamma}_j^{-n}}{n}\mathrm{e}^{-in\omega} + \sum_{n=1}^{\infty}\frac{\gamma_j^{-n}}{n}\mathrm{e}^{in\omega}\right).
\end{aligned} \tag{2.28}$$

Note now that $\log\left(-\overline{\delta}_j^{-1}\mathrm{e}^{-i\omega}\right) + \log\left(-\delta_j^{-1}\mathrm{e}^{i\omega}\right) = \log\left(\overline{\delta}_j^{-1}\mathrm{e}^{-i\omega}\delta_j^{-1}\mathrm{e}^{i\omega}\right) = -\log|\delta_j|^2$, and similarly for the terms $\log\left(-\overline{\gamma}_j^{-1}\mathrm{e}^{-i\omega}\right) + \log\left(-\gamma_j^{-1}\mathrm{e}^{i\omega}\right)$. We add these terms to the logarithm of the gain g, which now becomes

$$g' = 2\log(g) + \sum_{j=1}^{r}\log|\delta_j|^2 - \sum_{j=1}^{s}\log|\gamma_j|^2. \tag{2.29}$$

The power cepstrum coefficients $c_h(k)$ are now the inverse Fourier transform of $\log\Phi_h$, or

$$\sum_{k=-\infty}^{\infty} c_h(k)e^{-ik\theta} = \log\Phi_h(e^{i\theta}). \tag{2.30}$$

Comparing this with Equation (2.28) (and noting that poles and zeros appear in complex conjugate pairs), we readily see that

$$c_h(k) = \sum_{j=1}^{p}\frac{\alpha_j^{|k|}}{|k|} + \sum_{j=1}^{s}\frac{\gamma_j^{-|k|}}{|k|} - \sum_{j=1}^{q}\frac{\beta_j^{|k|}}{|k|} - \sum_{j=1}^{r}\frac{\delta_j^{-|k|}}{|k|} \quad \forall k \neq 0, \tag{2.31}$$

and

$$c_h(0) = g'. \tag{2.32}$$

2.3 The complex cepstrum

Because of the use of the power spectral density in equation (2.22), the power cepstrum does not take into account any information about the phase of the systems (i.e. whether poles and zeros have magnitudes greater or smaller than 1).

As we have seen in Subsection 2.2.2, for any unstable pole γ (i.e. $|\gamma| > 1$), the contribution to the power cepstrum coefficient $c(k)$, $\forall k \neq 0$, will be $\frac{\gamma^{-k}}{k}$.

If we want to explicitly take this phase information into account, we can instead employ the *complex cepstrum*[4] of a model, defined as

$$\hat{c}_h = \mathcal{F}^{-1}(\log H(\mathrm{e}^{i\omega})). \tag{2.33}$$

The complex cepstrum of a signal $y(k)$ is defined as

$$\hat{c}_y = \mathcal{F}^{-1}(\log Y(\mathrm{e}^{i\omega})). \tag{2.34}$$

Consider now again an all-pole, not necessarily stable, system

$$Y(z) = \frac{1}{a(z)c(z)}U(z), \tag{2.35}$$

where $a(z)$ contains the stable poles, $|\alpha_j| < 1$ and $c(z)$ the unstable poles, $|\gamma_j| > 1$.

We can calculate its model complex cepstrum, $\hat{c}_h(k) = \hat{c}_y(k) - \hat{c}_u(k)$ and it is easily shown, with a derivation analogous to the one in Subsection 2.2.2, that

$$\hat{c}_h(k) = \begin{cases} \displaystyle\sum_{j=1}^{q} \frac{\alpha_j^k}{k} & \forall k > 0 \\ \log(g') & k = 0 \\ \displaystyle\sum_{j=1}^{s} \frac{\gamma_j^{-|k|}}{|k|} & \forall k < 0 \end{cases}. \tag{2.36}$$

This equation implies that the positive coefficients of the complex cepstrum are the same as the power cepstrum coefficients of the stable part of the system,

[4]The term complex cepstrum is a bit confusing, in the sense that the complex cepstrum coefficients are real numbers. Rather, the phase information of the transfer function is retained in computing the complex cepstrum, giving rise to its name.

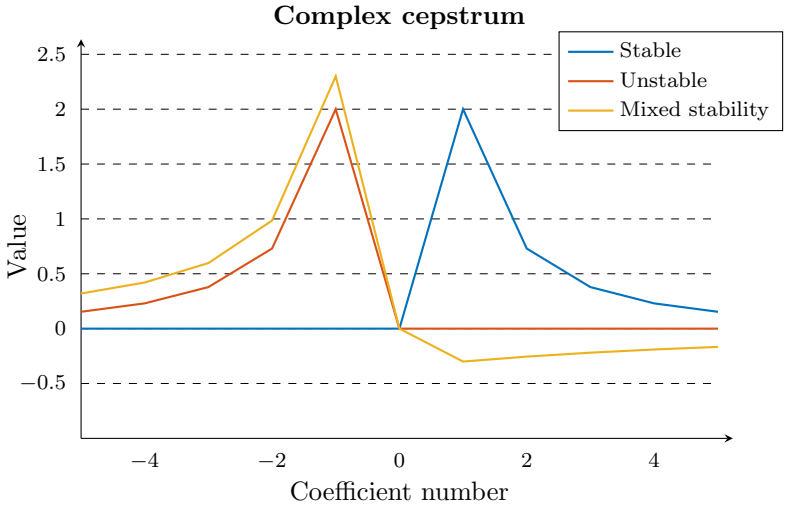


Figure 2.3: The first five positive and negative complex cepstrum coefficients of respectively a stable system (poles at 0.9, 0.7, 0.4, blue line), an unstable system (poles at 1/0.9, 1/0.7, 1/0.4, red line) and a mixed system (poles at 1/0.9, 0.7, 1/0.4, yellow line). The zeroth cepstrum coefficient has been artificially set to 0 in all three cases to improve readability of the graph. This coefficient can take on large (positive or negative) values, but is not important for the discussion here.

while the negative coefficients of the complex cepstrum are the power cepstrum coefficients of the unstable part of the system.

We show an example of the complex cepstrum in Figure 2.3.

2.4 Computational aspects

We expand on some computational aspects related to computing the power and complex cepstrum. Note that, throughout this thesis, the code of most examples, illustrations and applications is available online[5].

[5] `https://github.com/Olauwers`

2.4.1 Power cepstrum

The computation of the power cepstrum is quite straightforward. The power cepstrum of a signal y can be computed as

$$c_y(k) = \text{IFFT}(\log(\Phi_y)), \tag{2.37}$$

with $k = 1, \ldots, N$, and N the length of the FFT. Both the implementation of the IFFT (*Inverse Fast Fourier Transform*) [11] and the logarithm are straightforward in this case and are pre-implemented in systems theory-related packages in many commonly used scientific programming languages like Matlab and Python.

Computing a good estimate of the power spectral density Φ_y is a little more involved. For long enough signals (from about $N = 2^{10}$ and beyond), we can employ the well-known *Fast Fourier Transform*, or FFT, as an approximation of the Fourier transform and implement

$$\Phi_y(k) = \frac{1}{N}|\text{FFT}(y)|^2, \tag{2.38}$$

with N again the length of the FFT and $|\cdot|$ denotes the magnitude. The FFT has a computational complexity of $\mathcal{O}(n \log n)$.

Since the accuracy of the power spectral density estimate is very important to our techniques, it is, however, safer to use more accurate techniques, especially for shorter time series. One such algorithm is *Welch's method* [84].

Welch's method divides the signal in overlapping windows, estimates the power spectral density of each of the windows using the FFT, and averages them out. The result is a less noisy estimate. This technique is again of $\mathcal{O}(n \log n)$, and is what we will use most of the time.

For shorter time series ($N \lessapprox 2^7$), even more accurate estimation techniques exist (e.g., the *Multitaper method* [61]), but these will be computationally more expensive and are therefore to be avoided for longer signals.

The power cepstrum is then computed as in Algorithm 1.

2.4.2 Complex cepstrum

Computing the complex cepstrum is a little more involved, due to the non-uniqueness of the complex logarithm. We will give a short overview of its computation here, a much more detailed account can be found in [59]. We

Algorithm 1 Compute the power cepstrum

```
procedure POWER_CEPSTRUM(y)
    Φ_y ← Welch(y)
    Φ'_y ← log(Φ_y)
    c_y(k) ← IFFT(Φ'_y)
return c_y(k)
```

will again assume an output signal y, with analogous results holding for input signals and transfer functions.

In principle, the complex cepstrum $\hat{c}_y(k)$ can be easily computed, starting from Equation (2.33), as

$$\hat{c}_y(k) = \text{IFFT}(\log(\text{FFT}(y))), \tag{2.39}$$

with FFT and IFFT the Fast Fourier Transform and Inverse Fast Fourier Transform, respectively [11].

However, this expression is ambiguous. Evaluating the complex logarithm does not result in a unique solution. Indeed, the logarithm is the inverse of the exponential, and the complex exponential is a periodic function with period 2π. Usually, this ambiguity is solved by demanding that $-\pi < \arg(Y(z)) < \pi$, which we will call the principal values of the phase, or the *principal branch*, but this results in samples of a discontinuous phase curve. For our purposes, we need samples of the continuous phase curve.

We thus need to *unwrap* the phase by adding or subtracting multiples of 2π to get samples from the continuous phase curve. Luckily, many methods exist to do this, and we use an off-the-shelf one (*numpy.unwrap*) built into the Python NumPy library. Figure 2.4 shows the difference between the principal values of the phase and the unwrapped phase.

The complex cepstrum is then computed as in Algorithm 2.

Algorithm 2 Compute the complex cepstrum

```
procedure COMPLEX_CEPSTRUM(y)
    Y ← FFT(y)
    angles ← unwrap(arg(Y))
    Y' ← log(|Y|) + i * angles
    ĉ_y(k) ← IFFT(Y')
return ĉ_y(k)
```

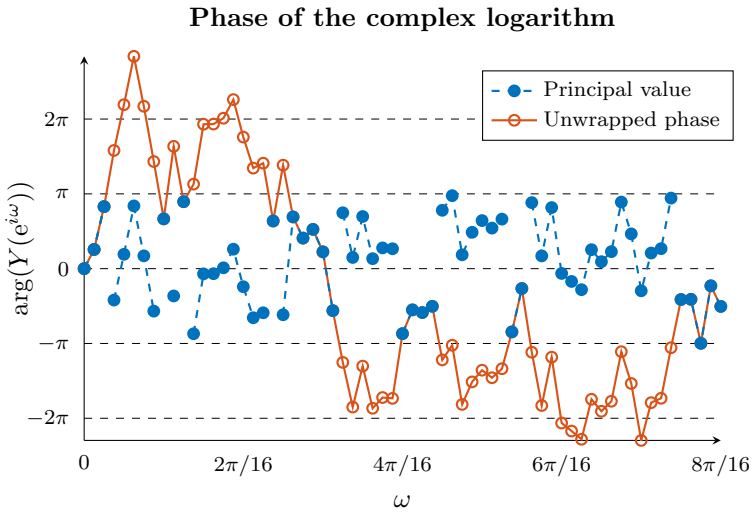


Figure 2.4: Part of the phase of the Fourier transform of an output signal y. The principal values (shown in blue dots) are constrained to be between $-\pi$ and π. To compute the complex cepstrum, we *unwrap* these principal values to obtain samples (shown by the red circles) from the continuous phase curve (an estimate of which is fitted through the samples with a red line). Where the dashed line between blue dots is interrupted, a multiple of 2π was added or subtracted to keep values between $-\pi$ and π.

2.5 The weighted power cepstral distance

We can use the definition of the power cepstrum above to define a distance measure. We will show this distance measure can be interpreted in terms of model parameters.

The weighted power cepstral distance, as first defined in [54], and further explored in [17, 19], between two signals y_1 and y_2, is now defined as

Definition 2. Given two signals, y_1 and y_2 the original (stochastic) weighted cepstral distance was defined as

$$d_c^2(y_1, y_2) = \sum_{k=1}^{\infty} k \left(c_{y_2}(k) - c_{y_1}(k)\right)^2 . \tag{2.40}$$

In the case of stochastic systems, with white noise inputs, we noted that $c_y(k) = c_h(k)$, and we can interpret the weighted cepstral distance in terms of

model parameters.

Theorem 1. *For y_1 (y_2) the output signal of a stochastic, stable, all-pole model H_1 (H_2) of order p (q), with poles α_i (β_i), we have*

$$d_c^2(y_1, y_2) = \log \frac{\prod_{i=1}^{p}\prod_{j=1}^{q} \left|1 - \alpha_i \bar{\beta}_j\right|^2}{\prod_{i,j=1}^{p} (1 - \alpha_i \bar{\alpha}_j) \prod_{i,j=1}^{q} \left(1 - \beta_i \bar{\beta}_j\right)}, \tag{2.41}$$

giving us an interpretation of the stochastic weighted cepstral distance in terms of poles and zeros of the underlying models.

Proof. We have

$$c_{y_1}(k) = \sum_{j=1}^{p} \frac{\alpha_j^{|k|}}{|k|}, \quad c_{y_2}(k) = \sum_{j=1}^{q} \frac{\beta_j^{|k|}}{|k|}, \qquad \forall k \neq 0, \tag{2.42}$$

and $c_{y_i}(0) = g'_{y_i}$, as in Subsection 2.2.2.

We then have,

$$\begin{aligned} d_c^2(y_1, y_2) &= \sum_{k=1}^{\infty} k \left(\sum_{j=1}^{p} \frac{\alpha_j^k}{k} - \sum_{j=1}^{q} \frac{\beta_j^k}{k} \right)^2 \\ &= \sum_{i,j=1}^{p} \sum_{k=1}^{\infty} k \left| \frac{\alpha_i^k \alpha_j^k}{k^2} \right| + \sum_{i,j=1}^{q} \sum_{k=1}^{\infty} k \left| \frac{\beta_i^k \beta_j^k}{k^2} \right| - 2 \sum_{i=1}^{p} \sum_{j=1}^{q} \sum_{k=1}^{\infty} k \left| \frac{\alpha_i^k \beta_j^k}{k^2} \right| \\ &= - \sum_{i,j=1}^{p} \log(1 - \alpha_i \bar{\alpha}_j) - \sum_{i,j=1}^{q} \log\left(1 - \beta_i \bar{\beta}_j\right) + \sum_{i=1}^{p} \sum_{j=1}^{q} \left|1 - \alpha_i \bar{\beta}_j\right|^2 \\ &= \log \frac{\prod_{i=1}^{p}\prod_{j=1}^{q} \left|1 - \alpha_i \bar{\beta}_j\right|^2}{\prod_{i,j=1}^{p} (1 - \alpha_i \bar{\alpha}_j) \prod_{i,j=1}^{q} \left(1 - \beta_i \bar{\beta}_j\right)}, \end{aligned} \tag{2.43}$$

where we used the series expansion from Equation (2.26). □

This stochastic version of the weighted power cepstral distance was proposed by [54] and explored in-depth by [17], where, amongst other results, the above

result and proof were given. In the paper [45], included in this text as Chapter 3, we will use the model power cepstrum to extend this existing definition to allow for deterministic inputs. For completeness, we introduce this extended distance measure here:

Definition 3. Starting from two model power cepstra, c_{h_1} and c_{h_2}, belonging to models with transfer functions H_1 and H_2, we define the (extended) weighted cepstral distance as

$$d_c^2(c_{h_1}, c_{h_2}) = \sum_{k=1}^{\infty} k \left(c_{h_1}(k) - c_{h_2}(k)\right)^2. \tag{2.44}$$

We do not have access to the impulse responses directly, but, as noted above, for given input, u_1 (u_2), and output, y_1 (y_2), signals for the model H_1 (H_2), we can compute this distance as

$$d_c^2((y_1, u_1)(y_2, u_2))^2 = \sum_{k=1}^{\infty} k\Big(\big(c_{y_1}(k) - c_{u_1}(k)\big) - \big(c_{y_2}(k) - c_{u_2}(k)\big)\Big)^2. \tag{2.45}$$

In Chapter 3, we formally introduce this distance measure, and compare it to traditional time series similarity measures in a clustering example on electrical circuits.

2.5.1 The weighted (power) cepstral norm

Associated with the weighted cepstral distance measure, we introduce the weighted cepstral model norm.

Definition 4. Given a model with transfer function H, with c_h the model cepstrum coefficients. The cepstral norm is then defined as

$$||H||_c^2 = \sum_{k=1}^{\infty} k c_h^2(k). \tag{2.46}$$

Consider now two systems, H_1 and H_2, with model power cepstra c_{h_1} and c_{h_2}, for which we can define the cascade $H_{tot} = H_1 H_2^{-1}$. From the definition of the power cepstrum, in Equation (2.22), we can easily see

$$c_{h_{tot}}(k) = c_{h_1}(k) - c_{h_2}(k), \tag{2.47}$$

and therefore

$$||H_{tot}||_c^2 = d_c^2\left(c_{h_1}, c_{h_2}\right). \tag{2.48}$$

Note that, even if we choose the two models to be all-pole models, the cascade H_{tot} is still a model with poles and zeros. The weighted cepstral distance for the stochastic pole-zero case was discussed by [19]. Relevant results (now extended to systems with inputs and outputs) will be discussed in Subsection 2.6.1.

2.5.2 The weight of the zero-th cepstrum coefficient

The weights in the model norm in Equation (2.46) are what makes the weighted cepstral distance work. Indeed, this choice of weights is the main contribution of the original introduction of the distance measure by Martin in [54]. The convergence of the infinite sum was the main motivator for this weighting. As a reminder, the model norm was

$$||H||_c^2 = \sum_{k=0}^{\infty} k c_h(k)^2, \tag{2.49}$$

which means the coefficients are weighted by a factor k.

When weighting the power cepstra, we usually choose to weight the zero-th cepstral coefficient, $c_h(0)$, by 0, effectively omitting it from consideration. This zero-th coefficient contains information on the gain of the underlying system, which decides the scale[6] of the time series, and perhaps a contribution of any unstable poles, as was made explicit in Equation (2.32). Let us introduce a way to weight the cepstral coefficients, so as to turn the semi-norm into a proper norm.

In many applications, we want to compare the signals in a scale-free manner, and omitting the zero-th component is the proper approach. Note, however, that from the model norm point of view, we now do not actually have a proper model norm for stable and minimum-phase LTI systems. Instead, we have a model norm for equivalence classes of such systems, where all systems differing up to a scalar gain belong to the same class. When this is not appropriate for the problem at hand, we can consider a way to include the zero-th cepstral coefficient anyway. To do so, we explore the link between the weighted cepstral norm and the functional theory of Dirichlet space.

An excellent introduction to the theory of Dirichlet space is found in [3]. Following the definition presented there, the Dirichlet space, $\mathcal{D}$, is the Hilbert

[6]Note that this is the reason we do not need to normalize time series before comparing them, when using the weighted cepstral distance: the weighting of the power cepstrum does this automatically for us.

space of analytic functions, f, in the unit disc, $\mathbb{D} = \{z \in \mathbb{C} : |z| < 1\}$, for which the semi-norm

$$||f||^2_{\mathcal{D},*} = \int_{\mathbb{D}} |f'(z)|^2 dA(z), \tag{2.50}$$

is finite, with $dA(z)$ a normalized area measure for the complex plane. It can be shown that, writing f out in its z-transform, $f(z) = \sum_{n=0}^{\infty} a_n z^n$,

$$||f||^2_{\mathcal{D},*} = \sum_{n=1}^{\infty} n|a_n|^2, \tag{2.51}$$

which is reminiscent of the weighted cepstral norm as defined in (2.46)! More precisely, for a system with transfer function H and cepstrum coefficients $c_h(k)$, we have that $||H||^2_c = ||\log(|H|^2)||^2_{\mathcal{D},*} = \sum_{k=1}^{\infty} k|c_h(k)|^2$. The functional theory of Dirichlet spaces is thus intricately linked with stable and minimum-phase (as their poles and zeros lie within the unit disk) LTI systems, and the weighted cepstral model norm.

A detailed connection between the two is outside the scope of this thesis (though would make an excellent topic of future research), and we will highlight just one observation made in [3]. While there are several proposals to extend the semi-norm to a norm, including looking at the limit behavior of the function on the unit disc, for our purposes, an interesting one is

$$||f||^2_{\mathcal{D}} = ||f||^2_{\mathcal{D},*} + |f(0)|^2. \tag{2.52}$$

From the definition of the z-transform, we easily see that $f(0) = a_0$, and for the cepstral norm on a transfer function, we find analogously

$$||H||^2_{c,\prime} = c_h(0)^2 + \sum_{k=1}^{\infty} k(c_h(k))^2 \tag{2.53}$$

$$= c_h(0)^2 + ||H||^2_c \tag{2.54}$$

with $||H||^2_{c,\prime}$ a norm, extended from the weighted cepstral semi-norm $||H||^2_c$ we consider in this thesis. In other words, weighting the zero-th cepstrum coefficient by 1 changes the norm considered in Equation (2.46) from a semi-norm to a proper norm on minimal, stable and inversely stable, SISO LTI models.

In some contexts, such as applications where the scale of the signals is important to take into account, this is a very useful addition to the framework developed in this text.

2.6 Model assumptions

As mentioned above, we usually assume minimal, stable and inversely stable, all-pole models.

When working with two systems with both poles and zeros, however, we can always recast the weighted cepstral distance between them in terms of all-pole models. In Subsection 2.6.1, we show how.

Furthermore, the assumption of stability and inverse stability can be relaxed partially. For maximal-phase systems, containing only unstable poles, we can still calculate the distance in Equation (2.44). Subsection 2.6.2 will show how.

2.6.1 Models with poles and zeros

Consider two pole-zero models, H_1, with p_1 poles $\alpha_{1,i}$ and q_1 zeros $\beta_{1,i}$, and H_2, with p_2 poles $\alpha_{2,i}$ and q_2 zeros $\beta_{2,i}$. Their power cepstra are (see Subsection 2.2.2, Equation (2.31))

$$c_{h_1}(k) = \sum_{j=1}^{p_1} \frac{\alpha_{1,j}^{|k|}}{|k|} - \sum_{j=1}^{q_1} \frac{\beta_{1,j}^{|k|}}{|k|}, \quad c_{h_2}(k) = \sum_{j=1}^{p_2} \frac{\alpha_{2,j}^{|k|}}{|k|} - \sum_{j=1}^{q_2} \frac{\beta_{2,j}^{|k|}}{|k|}, \qquad \forall k \neq 0. \tag{2.55}$$

Considering input and output signal pairs, (y_1, u_1) and (y_2, u_2), coming from H_1 and H_2, respectively, the weighted cepstral distance in Equation (2.44) is

$$\begin{aligned} & d_c^2\left((y_1, u_1), (y_2, u_2)\right) \\ & = \sum_{k=1}^{\infty} k \left(\sum_{j=1}^{p_1} \frac{\alpha_{1,j}^{|k|}}{|k|} - \sum_{j=1}^{q_1} \frac{\beta_{1,j}^{|k|}}{|k|} - \sum_{j=1}^{p_2} \frac{\alpha_{2,j}^{|k|}}{|k|} + \sum_{j=1}^{q_2} \frac{\beta_{2,j}^{|k|}}{|k|} \right)^2 . \end{aligned} \tag{2.56}$$

Grouping variables with the same sign together as

$$\pi_i \in \{\alpha_{1,1}, \ldots, \alpha_{1,p_1}, \beta_{2,1}, \ldots, \beta_{2,q_2}\}, \tag{2.57}$$

$$\zeta_i \in \{\alpha_{2,1}, \ldots, \alpha_{2,p_2}, \beta_{1,1}, \ldots, \beta_{1,q_1}\}, \tag{2.58}$$

we can rewrite this as

$$\begin{aligned} d_c^2\left((y_1,u_1),(y_2,u_2)\right) &= \sum_{k=1}^{\infty} k \left(\sum_{j=1}^{p_1+q_2} \frac{\pi_j^{|k|}}{|k|} - \sum_{j=1}^{p_2+q_1} \frac{\zeta_j^{|k|}}{|k|} \right)^2 \\ &= d_c^2\left(\frac{z^{p_1+q_2}}{\prod_{i=1}^{p_1+q_2}(z-\pi_i)}, \frac{z^{p_2+q_1}}{\prod_{i=1}^{p_2+q_1}(z-\zeta_i)} \right). \end{aligned} \tag{2.59}$$

This is the same as the distance between two all-pole models with poles π_i and ζ_i, respectively. The results from this thesis (with exception of the novel system identification technique in Chapter 6) trivially carry over to the models with both poles and zeros. For clarity and ease of notation, we will thus usually assume all-pole models, without loss of generality.

2.6.2 Stability

Consider a system with transfer function

$$H(z) = \frac{1}{c(z)}, \tag{2.60}$$

with $c(z)$ a polynomial built from s poles γ_i that are unstable, i.e. $|\gamma_i| > 1$.

The power cepstrum of such an unstable system is given (from Equation (2.31) in Subsection 2.2.2) by

$$c_h(k) = \sum_{j=1}^{s} \frac{\gamma_j^{-|k|}}{|k|} \qquad \forall k \neq 0, \tag{2.61}$$

and $c_h(0) = g'$.

The power cepstrum of a system with unstable poles $|\gamma_j| > 1$, $j \in \{1, \ldots, s\}$, consequently, is equal to the power cepstrum of a system with stable poles γ_j^{-1}. This is as expected, as the power cepstrum is based on the power spectral density, which is invariant under inversion of the complex plane over the unit circle (i.e. changing $z \mapsto z^{-1}$).

When the system contains a mixture of stable and unstable poles, the weighted power cepstral distance will not give sensible results. While we can still compute the model power cepstrum coefficients, there will be no difference between a

stable pole and its unstable counterpart. For a stable one-pole model, $H_s = \frac{z}{z-\alpha}$ and its unstable counterpart $H_u = \frac{z}{z-\alpha^{-1}}$, this means that

$$d_c^2(\frac{z}{z-\alpha}, \frac{z}{z-\alpha^{-1}}) = 0, \tag{2.62}$$

which is of course not what we want.

The complex cepstrum, as defined in 2.3, separates the stable and unstable parts of the system, and might provide a solution to this problem. This was not further explored in this thesis. This complex cepstrum will provide a way to identify mixed-phase models in Chapter 6.

2.7 Subspace angles

The weighted power cepstral distance can not only be interpreted in terms of model parameters, as in Section 2.5, but also in terms of what is known as subspace angles. We introduce this geometrical concept here, and will come back to it for the deterministic case in Chapter 4.

2.7.1 Principal angles

Principal angles are generalizations of angles between vectors, and describe angles between given subspaces of a vector space, going back to Akaike [2], Hotelling [30], and Jordan [33]. Figure 2.5 shows a visualisation of these angles, along with a short description of how to obtain them. They are defined recursively as follows.

Denote the principal angles between two subspaces, V and W, by θ_k, $k \in \{1, 2, \ldots, l\}$, with $\theta_1 \leq \theta_2 \leq \ldots \leq \theta_l \leq \frac{\pi}{2}$. With d_V and d_W the dimensions of V and W, respectively, we have that $l = \min(d_V, d_w)$. Now take unit vectors, $v \in V$ and $w \in W$. We then define recursively the principal angles θ_k as

$$\begin{aligned} \cos(\theta_k) = \underset{v \in V, w \in W}{\text{maximize}} \quad & v^\intercal \cdot w \\ \text{subject to} \quad & ||v|| = ||w|| = 1, \\ & v^\intercal \cdot v_i = 0 \quad \forall i < k, \\ & w^\intercal \cdot w_i = 0 \quad \forall i < k. \end{aligned} \tag{2.63}$$

The v_i and w_i found in this way are called the *principal directions* in V and W, respectively.

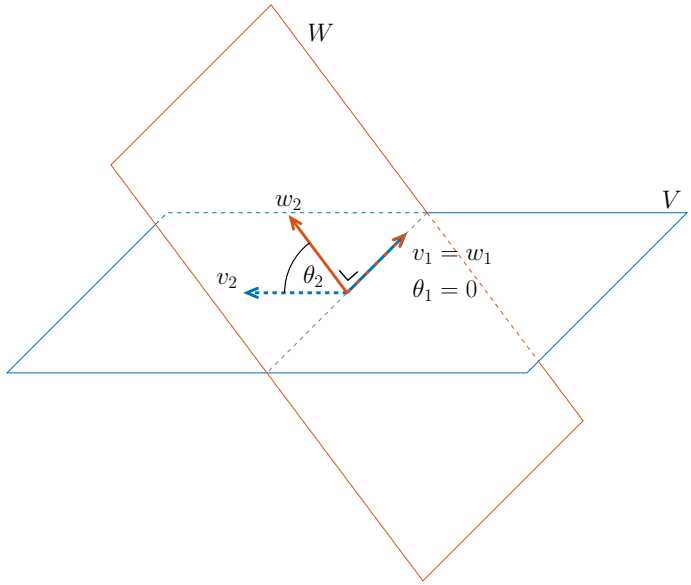


Figure 2.5: Given two subspaces V and W, we define the principal angles between them by first choosing unit vectors, called principal directions, $v_1 \in V$ and $w_1 \in W$, such that $\theta_1 = \angle(v_1, w_1)$, the angle between the two vectors, is minimised (in this case, it is 0). We then iteratively find new principal directions $v_k \in V$ and $w_k \in W$ in V and W, perpendicular to all previous principal directions, such that the angle $\theta_k = \angle(v_k, w_k)$ between them is minimised. These angles θ_k are the principal angles between subspaces V and W and can be calculated in general from an eigenvalue problem.

These principal angles give a measure of (dis)similarity between subspaces. The smaller the angles, the more similar or *closer* the subspaces are. If all principal angles were equal to zero, all principal directions would coincide, and the subspaces would be equal. If all directions were perpendicular to each other, on the other hand, the subspaces would be maximally far apart.

A generalized eigenvalue problem

As shown in [17, 19, 78], the squared cosines of the principal angles and the principal directions between the column spaces of two matrices $A \in \mathbb{R}^{m \times p}$ and $B \in \mathbb{R}^{m \times q}$, of full column rank can be calculated from the symmetric generalized

eigenvalue problem

$$\begin{pmatrix} 0 & A^\intercal B \\ B^\intercal A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} A^\intercal A & 0 \\ 0 & B^\intercal B \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}, \tag{2.64}$$

subject to $x^\intercal A^\intercal A x = 1$ and $y^\intercal B^\intercal B y = 1$. The solutions λ_i, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{p+q}$ of this eigenvalue problem can be related [19] to the cosines of the principal angles θ_i between the column spaces of A and B.

From here, one can show that, when $p \leq q$,

$$\cos^2 \theta_i = \lambda_i \left((A^\intercal A)^{-1} A^\intercal B (B^\intercal B)^{-1} B^\intercal A \right), \tag{2.65}$$

where $\lambda_i(\cdot)$ denotes the i-th eigenvalue of the expression between brackets.

2.7.2 Subspace angles

This notion of principal angles can be employed to define the subspace angles between dynamical models, as follows:

Definition 5. Given two n-th order all-pole models, H_1 and H_2, with observability matrices $\Gamma_1^i, \Gamma_2^i \in \mathbb{R}^{i \times n}$, we define the n subspace angles between these models as the principal angles between the column spaces of their observability matrices

$$\{\theta_k\} = \lim_{i \to \infty} \left[\Gamma_1^i \lhd \Gamma_2^i \right], \tag{2.66}$$

where $\lhd$ denotes the principal angles between the column spaces of the matrix on the left and on the right.

2.7.3 Subspace angles kernel

Based on the subspace angles, we can define a positive-definite kernel:

Definition 6. Given two n-th order all-pole models H_1 and H_2, with n subspace angles $\{\theta_k\} = \lim_{i \to \infty} \left[\Gamma_1^i \lhd \Gamma_2^i \right]$ between them, we can define the subspace angles kernel

$$\kappa(H_1, H_2) = \prod_{i=1}^{n} \cos^2(\theta_i). \tag{2.67}$$

We also have, as proven in [19], that for the stochastic case, this kernel can be related to the weighted cepstral distance, in the following theorem.

Theorem 2. *Given two n-th order all-pole models H_1 and H_2, with n subspace angles $\{\theta_k\} = \lim_{i\to\infty} \left[\Gamma_1^i \triangleleft \Gamma_2^i\right]$ between them, we can relate the weighted cepstral distance between them to the subspace angles kernel as follows:*

$$d_c^2(H_1, H_2) = -\log \kappa(H_1, H_2) = -\log \prod_{i=1}^{n} \cos^2(\theta_i). \tag{2.68}$$

An important lemma (from [17]) in proving this is

Lemma 3. *Consider a minimal SISO model with distinct poles α_l, $l \in \{1, 2, \ldots, n\}$. The column space of the $i \times n$ ($i \geq n$) observability matrix, Γ^i, is only determined by its poles. We show this column space is equal to the column space of the Vandermonde matrix of the poles, Γ_V^i,*

$$\text{range}(\Gamma^i) = \text{range}(\Gamma_V^i) = \text{range} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{i-1} & \alpha_2^{i-1} & \cdots & \alpha_n^{i-1} \end{pmatrix}. \tag{2.69}$$

Proof. Diagonalizing the system matrix from the state space representation in Equation (2.1), through the similarity transformation T, we get $\Lambda = \text{diag}(\alpha_1, \alpha_2, \ldots, \alpha_n) = TAT^{-1}$, and $C_V = CT^{-1}$.

We can then rewrite the observability matrix

$$\begin{aligned} \Gamma^i &= \begin{pmatrix} (C)^\intercal & (CA)^\intercal & (CA^2)^\intercal & \cdots & (CA^{i-1})^\intercal \end{pmatrix}^\intercal \\ &= \begin{pmatrix} (C_V)^\intercal & (C_V\Lambda)^\intercal & (C_V\Lambda^2)^\intercal & \cdots & (C_V\Lambda^{i-1})^\intercal \end{pmatrix}^\intercal T \\ &= \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{i-1} & \alpha_2^{i-1} & \cdots & \alpha_n^{i-1} \end{pmatrix} \begin{pmatrix} c_1' & 0 & \cdots & 0 \\ 0 & c_2' & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & c_n' \end{pmatrix} T, \end{aligned} \tag{2.70}$$

with c_l', $l \in \{1, \ldots, n\}$, the components of C_V. As T and C_V are of full rank, we now have that $\text{range}(\Gamma^i) = \text{range}(\Gamma_V^i)$. □

Furthermore, we can prove

Lemma 4. *Given two n-th order all-pole models H_1 and H_2, with poles α_i and β_i, respectively. We have the following interpretation of the kernel function in terms of model parameters:*

$$\kappa(H_1, H_2) = \frac{\prod_{i,j=1}^{n} (1 - \alpha_i \bar{\alpha}_j) \prod_{i,j=1}^{n} \left(1 - \beta_i \bar{\beta}_j\right)}{\prod_{i=1}^{n} \prod_{j=1}^{n} \left|1 - \alpha_i \bar{\beta}_j\right|^2}. \tag{2.71}$$

Proof. From Equation (2.65), we can calculate the product of the squared cosines of the subspace angles as

$$\kappa\left(H_1, H_2\right) = \prod_{i=1}^{n} \cos^2 \theta_i \tag{2.72}$$

$$= \lim_{i \to \infty} \det \left(\left(\Gamma_1^{i\intercal} \Gamma_1^i \right)^{-1} \Gamma_1^{i\intercal} \Gamma_2^i \left(\Gamma_2^{i\intercal} \Gamma_2^i \right)^{-1} \Gamma_2^{i\intercal} \Gamma_1^i \right) \tag{2.73}$$

$$= \lim_{i \to \infty} \det \left(\left(\Gamma_{V,1}^{i\,H} \Gamma_{V,1}^i \right)^{-1} \Gamma_{V,1}^{i\,H} \Gamma_{V,2}^i \left(\Gamma_{V,2}^{i\,H} \Gamma_{V,2}^i \right)^{-1} \Gamma_{V,2}^{i\,H} \Gamma_{V,1}^i \right) \tag{2.74}$$

$$= \frac{\prod_{i,j=1}^{n} (1 - \alpha_i \bar{\alpha}_j) \prod_{i,j=1}^{n} \left(1 - \beta_i \bar{\beta}_j\right)}{\prod_{i=1}^{n} \prod_{j=1}^{n} \left|1 - \alpha_i \bar{\beta}_j\right|^2}, \tag{2.75}$$

where the $\cdot^H$ denotes the Hermitian conjugate. In going from (2.73) to (2.74), we used Lemma 3. The step from (2.74) to (2.75) follows from the series expansion

$$\sum_{k=1}^{\infty} x^k = \frac{1}{1 - x} \quad \forall |x| < 1. \tag{2.76}$$

□

Together with Theorem 1, this last lemma proves the equivalence in Theorem 2.

2.7.4 Subspace angles kernel from output signals

In [80], it was claimed that this kernel could be calculated on the outputs[7] of the system, and that it was equivalent to the (stochastic) weighted cepstral distance.

It was shown in [17], that this is only true in the case of two systems driven by the same noise realization. (The authors of [80] rectified this in a later paper [81], by assuming there is no noise input, essentially assuming autonomous systems. Here, they also introduced the correct formulation in terms of Hankel matrices of output signals, as we will develop momentarily.)

Following [17], we denote by $Y^{i,j}$ the $i \times j$ Hankel matrix composed of i rows of consecutive elements of the time series y, starting at $y(0)$, where j is the number of columns[8] considered ($i \ll j$). This matrix is thus constructed as follows:

$$Y^{i,j} = \begin{pmatrix} y(0) & y(1) & y(2) & \cdots & y(j-1) \\ y(1) & y(2) & y(3) & \cdots & y(j) \\ y(2) & y(3) & y(4) & \cdots & y(j+1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y(i-1) & y(i) & y(i+1) & \cdots & y(i+j-2) \end{pmatrix}. \tag{2.77}$$

We can define a similar input Hankel matrix $U^{i,j}$ from the input signal $u(k)$. The assumption we made that the signals should be persistently exciting translates here as the Hankel matrices being of full rank i, so that $\text{rank}(Y^{i,j}) = \text{rank}(U^{i,j}) = i$, and the signals are said to be persistently exciting of order at least i.

The correct, most general statement proven on all-pole[9] models in [17], considers two systems of order n, with outputs y_1 and y_2, respectively, and driven by the same white noise input process $w(k)$. They have observability matrices Γ_1^i and Γ_2^i. We can construct the Hankel matrices $Y_1^{i,j}$ and $Y_2^{i,j}$, and we now have

$$\lim_{j\to\infty} \left[(Y_1^{i,j})^\intercal \triangleleft (Y_2^{i,j})^\intercal \right] = \lim_{i\to\infty} \left[\Gamma_1^i \triangleleft \Gamma_2^i \right], 0, \ldots, 0, \tag{2.78}$$

which means we can compute the subspace angles between the models by computing the n largest principal angles between the row spaces of the Hankel

[7] The authors of [80] did not call them outputs, but trajectories of the system, which is the equivalent of the output signal in the behavioral framework explored in [86]. However, they defined the trajectories of a system as the set of output observations y_t of the system, with t the relevant time points to the problem.

[8] Typically, for stationary time series, the limit case where $j \to \infty$ is considered.

[9] We will discuss the principal angles between models with both poles and zeros in Subsection 2.7.5.

matrices of their output signals. (The other principal angles will be equal to zero. In the context of the subspace angles kernel from Definition 6, this does not matter too much, as, for any $\theta = 0$, we have $\cos(\theta) = 1$, which does not change the product in the definition.)

The assumption that the two systems are driven by the same white noise process is very restrictive. In Chapter 4, we will come back to this problem, and extend this computation to deterministic systems, that do not necessarily have to have the same input.

2.7.5 Models with poles and zeros

When working with models with both poles and zeros, the subspace angles are defined in [19] as follows:

Definition 7. Consider invertible pole-zero models, H_1 of order n_1 and H_2 of order n_2, with observability matrices $\Gamma_1^i \in \mathbb{R}^{i \times n_1}$ and $\Gamma_2^i \in \mathbb{R}^{i \times n_2}$. Similarly, we can construct observability matrices, $\Gamma_{1,\text{inv}}^i \in \mathbb{R}^{i \times n_1}$ and $\Gamma_{2,\text{inv}}^i \in \mathbb{R}^{i \times n_2}$, for the inverted models, H_1^{-1} and H_2^{-1}. We define the subspace angles between H_1 and H_2 as the principal angles between the column spaces of their extended observability matrices

$$\{\theta_k\} = [H_1 \lhd H_2] = \lim_{i \to \infty} \left[\begin{pmatrix} \Gamma_1^i & \Gamma_{2,\text{inv}}^i \end{pmatrix} \lhd \begin{pmatrix} \Gamma_2^i & \Gamma_{1,\text{inv}}^i \end{pmatrix} \right], \tag{2.79}$$

where $\lhd$ denotes the principal angles between the column spaces of the matrix on the left and on the right. The observability matrices of the inverse models were defined in Equation (2.18).

The subspace angles between these types of models are discussed extensively in [17], and most results presented here carry over. Notably, the result from Lemma 3 carries over: the column space of the observability matrix of a model with both poles and zeros is only determined by its poles; that of the observability matrix of the inverse system is determined by the zeros.

Chapter 3

A time series distance measure for efficient clustering of input/output signals by their underlying dynamics

This paper was previously published as:
[45] Lauwers, O., and De Moor, B. A time series distance measure for efficient clustering of input/output signals by their underlying dynamics. *IEEE Control Systems Letters 1* (June 2017), 286–291

Contribution: Oliver Lauwers developed the theoretical contributions of the paper, the accompanying code, and performed all experiments. He took the lead in writing the paper.

Abstract

Starting from a dataset with input/output time series generated by multiple deterministic linear dynamical systems, this paper tackles the problem of automatically clustering these time series. We propose an extension to the so-called Martin cepstral distance, that allows to efficiently cluster these time series, and apply it to simulated electrical circuits data.

Traditionally, two ways of handling the problem are used. The first class of methods employs a distance measure on time series (e.g. Euclidean, Dynamic Time Warping) and a clustering technique (e.g. k-means, k-medoids, hierarchical clustering) to find natural groups in the dataset. It is, however, often not clear whether these distance measures effectively take into account the specific temporal correlations in these time series. The second class of methods uses the input/output data to identify a dynamic system using an identification scheme, and then applies a model norm-based distance (e.g. H_2, H_∞) to find out which systems are similar. This, however, can be very time consuming for large amounts of long time series data.

We show that the new distance measure presented in this paper performs as good as when every input/output pair is modelled explicitly, but remains computationally much less complex. The complexity of calculating this distance between two time series of length N is $\mathcal{O}(N \log N)$.

3.1 Introduction

Time series clustering is an important topic in modern research. State-of-the-art clustering methods of other data types are often not suited for this high-dimensional, temporally correlated data structure. Clustering is the task of finding groups with similar elements in a dataset and consists of three components: a similarity measure based on relevant data features, a clustering algorithm and an evaluation criterion. While the latter two components might carry over, defining a good distance measure is a difficult problem, especially if one is interested in the dynamics of the generating dynamical system of the time series.

Representing the time series as convolutional single-input single-output (SISO) linear time invariant (LTI) deterministic dynamical systems further generates problems of its own, as the contributions of the input signal and the impulse response of the system are convolved in the time domain. It is thus not intuitively clear how these two contributions can be separated, for example when one is

interested only in the dynamics of the system and not in the specific input signal.

This problem grows ever more relevant as large scale big data time series problems grow more prevalent in areas like finance, medicine, or the industrial internet of things, where clustering is important in tasks like anomaly detection [53, 76]. A typical industrial problem contains several hundred sensors per machine, tens of machines per plant, and several plants per industrial player, collecting data every few seconds, for months or even years of operation time. This results in datasets of several million time points for thousands of series. Clustering techniques should thus scale well.

In Section 3.2 we look at state-of-the-art clustering methods for time series from two perspectives, starting from a dataset containing input/output time series pairs, generated by different SISO LTI dynamical systems. From a machine learning point of view, we use an automated clustering method with an off-the-shelf time series distance such as the Euclidean distance or Dynamic Time Warping (DTW). From a system identification point of view, we apply norms such as the H_2 or H_∞ norm to compare systems estimated from the data. We find that these techniques either are very fast, but give poor results, or perform well, but are computationally expensive.

Next, in Section 3.3, we look at the Martin cepstral distance [19, 54], which combines insights from systems theory into a distance measure that can be computed on the raw data. This metric was defined for SISO ARMA models (i.e. LTI models that use white noise as an input signal).

The main contribution of this paper is an extension of the cepstral distance measure, that incorporates deterministic input signals, and allows to calculate distances between a broader class of SISO LTI dynamical systems. It thus allows to cluster time series by dynamics, but remains computationally much simpler than explicitly estimating models.

Subsequently, we apply this new distance measure in Section 3.4 to a simulation of electrical circuits, where we generate a dataset consisting of input/output signal pairs, and the problem is to identify which data belong to which generating system. Finally, we conclude the paper and provide some paths for future research in Section 3.5.

3.2 Existing methods

Existing methods to cluster time series employ a clustering technique, together with some distance measure. The author of [47] discerns three types of distance

measures: measures based on raw data, measures based on features of the time series and measures based on models. For the scope of this paper, we will focus on the first and the latter (as the distance measure we propose combines elements of these two broad classes). We present two raw data distance measures, the Euclidean metric and the Dynamic Time Warping metric [36], and two model-based distance measures, connected to the H_2-norm and the H_∞-norm. In the next Section, we will introduce and extend the cepstral distance [19, 54], which combines the efficiency of the raw data distance measures with the insight in generative dynamics of the model norms, and thus has representations both as a raw data distance and as a model-based one.

3.2.1 Raw data distance measures

In what follows we will define u_m to be the input signal of the m-th element of a dataset, y_m is the corresponding output signal and $u_m(k)$ or $y_m(k)$ is the value at timepoint k of respectively the input and output of the m-th element of the input/output dataset. Time series from element m start at $k = 0$ and end at $k = N_m$. The system that generated an output from a given input will be called the generating (dynamical) system.

Euclidean distance

Definition 8. The Euclidean distance, $d_E(\cdot, \cdot)$ treats the time series as a vector, and applies the element-wise Euclidean vector distance between two time series of same length N_m, defined as

$$d_E(y_m, y_n) = \sqrt{\sum_{k=0}^{N_m} (y_m(k) - y_n(k))^2}. \tag{3.1}$$

Advantages

- The Euclidean distance is easy to calculate, allowing for very efficient computation and clustering.
- No system identification step is needed.

Disadvantages

- There is no clear link between this distance measure and the generating system.

- This measure treats the time series as a vector, and ignores the temporal correlations in the data.
- This measure does not allow to compute distances between time series of different lengths.
- This measure does not take the input into account.

Dynamic time warping

Dynamic Time Warping (DTW) [36, 67] is an algorithm that tries to locally align time series, by *warping* them such that the Euclidean distance between the warped time series is minimal. Mathematically, this *warping*, and the measure that is found in this way, can be described as follows.

Given two output signals, y_1 and y_2, of length N_1 and N_2 respectively, a matrix M is constructed, where the (l,m)-th element of M is defined as $M_{(l,m)} = (y_1(l) - y_2(m))^2$. A warping path, $W = w_1, w_2, \ldots, w_k, \ldots, w_K$ is then defined, with each $w_k = \left(M_{(l,m)}\right)_k$ an element of matrix M and $max(N_1, N_2) \leq K < N_1 + N_2 - 1$.

The path is subject to the boundary conditions $w_1 = M_{1,1}$ and $w_K = M_{N_1,N_2}$ (i.e. the path starts in one corner of the matrix and ends in the opposite one), has to be continuous, in such a way that two consecutive elements w_k and w_{k+1} are maximally one column and one row apart, and has to be monotonously increasing in its indices, i.e., that in going from w_k to w_{k+1}, column nor row number can decrease.

Definition 9. We are now interested in the warping path W_{DTW} that minimizes the cost function

$$d_{DTW}(y_1, y_2) = min\left\{\sqrt{\sum_{k=1}^{K} w_k}\right\}. \tag{3.2}$$

The sum over this path is then the DTW distance between the time series.

Though this algorithm is computationally expensive due to the combinatorial nature of the problem, several lower bounds have been devised that can be implemented efficiently. In what follows, we use the lower bound from [36].

Advantages

- The DTW distance takes into account (part of) the local temporal correlations.
- No system identification step is needed.

- Lower bounds on the distance are reasonably efficient.
- This measure allows to calculate distances between time series of different lengths.

Disadvantages

- There is no clear link between this distance measure and the generating system.
- The DTW distance as such is expensive to calculate.
- This measure does not take the input into account.

3.2.2 Model-based distance measures

We use the same notation as in subsection 3.2.1. The generating system of the input/output pair (u_m, y_m) will be denoted by M_m, and its corresponding transfer function will be written $\mathcal{H}_m$. Based on a model norm $||\cdot||$, the distance between two models M_i and M_j is defined as $||\mathcal{H}_i - \mathcal{H}_j||$.

H_2-norm

Definition 10. The H_2-norm, $||\mathcal{H}||_2$, of a discrete-time system M with transfer function $\mathcal{H}$ is defined as

$$||\mathcal{H}||_2 = \sqrt{\frac{1}{2\pi}\int_{-\pi}^{\pi} Tr\left\{\mathcal{H}^H(\mathrm{e}^{i\omega})\mathcal{H}(\mathrm{e}^{i\omega})\right\} d\omega}, \tag{3.3}$$

where $Tr\{\}$ denotes the trace, the superscript $\cdot^H$ denotes the Hermitian conjugate and i denotes the imaginary unit.

The H_2-norm can be seen as the root-mean-square of the system response to a normalized white noise input. It is thus a measure of the power, or steady-state variance of this response. The H_2-norm will be infinite for unstable systems.

Advantages

- The H_2-norm provides a physically interpretable way to characterize underlying dynamics of time series.
- This norm allows to calculate distances between time series of different lengths.
- This norm takes the input data into account.

- A system identification procedure is needed, which is both difficult to automate and often computationally expensive (at least more expensive than the raw data measures).

H_∞-norm

Definition 11. The H_∞-norm, $||\mathcal{H}||_\infty$, of a discrete-time system M with transfer function $\mathcal{H}$ is calculated as

$$||\mathcal{H}||_\infty = \max_{\omega \in [0,\pi[} |\mathcal{H}(\mathrm{e}^{i\omega})|. \tag{3.4}$$

This norm thus measures the maximal gain of the frequency response and is called the *gain* of the system. It becomes infinite for systems with poles on the unit circle.

Advantages

- The H_∞-norm provides a physically interpretable way to characterize underlying dynamics of time series.
- This norm allows to calculate distances between time series of different lengths.
- This norm takes the input data into account.

Disadvantages

- A system identification procedure is needed, which is both difficult to automate and often computationally expensive (at least more expensive than the raw data measures).

3.3 Cepstral distance

In this section we take a closer look at an insightful distance measure on ARMA models, which can be interpreted both as a raw data distance measure and as a model norm: the Martin cepstral norm [19, 54]. We first give a very concise review of the cepstral norm in the stochastic case, then proceed with an extension that allows us to incorporate information about the deterministic input signal.

3.3.1 Original cepstral norm

Based on the power spectral density, Φ_y, of a signal y, we can define its power cepstrum, c_y as

$$c_y = \mathcal{F}^{-1}(\log(\Phi_y)), \tag{3.5}$$

where $\mathcal{F}^{-1}$ denotes the inverse Fourier transform. This produces a series of coefficients, $c_y(k)$, with integer $k \in [0, N]$, where N denotes the length of time series y.

Definition 12. The cepstral norm, $||\mathcal{H}||_C$, of model M with transfer function $\mathcal{H}$, and output y is defined as

$$||\mathcal{H}||_C = \sum_{k=0}^{N} k \left(c_y(k)\right)^2. \tag{3.6}$$

For ARMA models it was proven that there are multiple methods to calculate this norm: it can be derived from the subspace angles of the output Hankel matrices of the generating system [19] or from the mutual information of the output space of a system, and from a combination of poles and zeros of the transfer function of the model [54] (equivalent to the one we will derive in subsection 3.3.2). Moreover, equation (3.6) allows us to calculate the norm straight from raw data (see also [59, Chapter 10]), without the need to identify the underlying systems. We can thus connect the cepstral norm to a raw data distance measure in the following sense:

Definition 13. The cepstral distance, $d_C(y_i, y_j)$, between two time series, y_i and y_j, is defined as

$$d_C(y_i, y_j) = \sum_{k=0}^{\max\{N_i, N_j\}} k \left(c_{y_i}(k) - c_{y_j}(k)\right)^2, \tag{3.7}$$

where $\max\{N_i, N_j\} - \min\{N_i, N_j\}$ zeros are added at the end of the cepstrum of length $\min\{N_i, N_j\}$.

Advantages

- The cepstral distance has an interpretation in terms of the generating models of the time series.
- The cepstral distance is easy to calculate, allowing for very efficient computation and clustering.
- No system identification step is needed.

- This measure allows to calculate distances between time series of different lengths.

Disadvantages

- This distance measure can only take information coming from a stochastic input into account.

3.3.2 Extended cepstral distance

The cepstrum, defined in the previous section, finds its roots in homomorphic signal processing [59, Chapter 10]. In this type of processing, the original time series data, which often involves complex multiplicative operations like convolutions, is mapped, through a non-linear mapping, to a different domain, that allows for linear filtering. The cepstrum, as in equation (3.5), is a good example. The convolution in the time domain changes into a multiplication by calculating the power spectral density:

$$\Phi_y = |\mathcal{H}|^2 \Phi_u. \tag{3.8}$$

Applying a logarithmic transformation then turns the multiplication in frequency domain into an addition, and we get

$$\log(\Phi_y) = \log(|\mathcal{H}|^2) + \log(\Phi_u). \tag{3.9}$$

Finally, the inverse Fourier transform takes the problem back to (a transformed version of) the time domain. Equation (3.5) is thus effectively a method to transform the convolution into an addition. Defining the cepstrum coefficients of the input signal u as $c_u(k)$, and the contribution to the cepstrum coefficients of the transfer function $\mathcal{H}$ as $c_h(k)$, we can write

$$c_y(k) = c_u(k) + c_h(k). \tag{3.10}$$

This allows us to take the output, and separate the contributions from the input signal (which was the main disadvantage left in the cepstral distance, see subsection 3.3.1) and the impulse responses of the system. Based on input/output signal pairs, we now have a measure of the underlying generating system dynamics by looking at $c_h(k) = c_y(k) - c_u(k)$.

Definition 14. The extended cepstral distance, $d_{C_e}((y_i, u_i), (y_j, u_j))$, between two input/output pairs of time series, (y_i, u_i) and (y_j, u_j), with respective

transfer functions $\mathcal{H}_i$ and $\mathcal{H}_j$, is defined as

$$d_{C_e}((y_i, u_i), (y_j, u_j)) = \sum_{k=0}^{\min\{N_i, N_j\}} k \left(c_{h_i}(k) - c_{h_j}(k)\right)^2. \tag{3.11}$$

By interpreting the transfer function as a quotient of two polynomials, with zeros and poles as their respective roots, we can find an interpretation of these cepstrum coefficients. Following the reasoning in [59, Chapter 10], we find (for stable, minimum-phase systems):

$$c_h(k) = \sum_{j=1}^{p} \frac{\alpha_j^{|k|}}{|k|} - \sum_{j=1}^{q} \frac{\beta_j^{|k|}}{|k|} \qquad \forall k \neq 0, \tag{3.12}$$

where the α's denote the p poles, and the β's denote the q zeros. This expression (which is analogous to the one in the stochastic case) sets us on the path to rederive the framework presented in [19] for the deterministic case. This, however, requires an extension of the notion of subspace angles, which is out of the scope of this paper.

We propose this extended cepstral distance as a way to efficiently cluster input/output data by generating dynamics.

Advantages

- The extended cepstral distance is linked to the generating model of the time series.
- The extended cepstral distance is easy to calculate, allowing for very efficient computation and clustering.
- No system identification step is needed.
- This measure allows to calculate distances between time series of different length.
- This measure takes the input into account.

Disadvantages

- The interpretation of the measure in terms of system parameters and properties is not immediately clear, thus the theoretical framework of the original cepstral distance does not carry over trivially.

3.3.3 Computational overview

A pseudo-code overview of the algorithm is shown in Algorithm 3. Calculating the extended cepstral distance amounts to estimating the power spectral density of both input and output by Welch's method [84] (employing the FFT[1], which is of $\mathcal{O}(n \log n)$, with n the length of the windows considered in Welch's method), taking the logarithm of the resulting vector, and then applying an inverse Fourier transform (employing the IFFT, running in $\mathcal{O}(N \log N)$ time, with N the length of the time series) on them. In the end, we then apply a weighted Euclidean distance on the results.

Note that, for very short time series (less than 2^7 time points), we use the multitaper method [61] to achieve better results. This method is a bit slower than Welch's method, but because we only apply it in the case of short time series, this does not matter too much for the analysis here. The complexity of calculating the extended cepstral distance between two time series is then $\mathcal{O}(N \log N)$, with N the length of the time series. That of the Euclidean distance is $\mathcal{O}(N)$, standard DTW is $\mathcal{O}(N^2)$, while the lower bound in [36] is also $\mathcal{O}(N \log N)$. The complexity of explicitly identifying a system and calculating model norms depends very much on the identification method used and the model size. Many of these techniques employ, at their core, a SVD decomposition, of order $\mathcal{O}(M^2 N)$, with M here being the model order. Often this core is then repeated in non-convex optimization problems, solved iteratively. Furthermore, hyperparameters (e.g. model order) need to be estimated, further increasing the computational cost. For example, a simple grid search repeats this whole process for every point in the grid.

3.4 Application on electrical circuits

A minimal working example of the simulations performed in this Section is available on GitHub.[4]

[1]Note that, for longer time series (i.e. 2^{10} and beyond), the Fast Fourier Transform [11] provides a clean enough output to work on. We could thus speed up the algorithm even further for longer series.

[3]For very short time series (i.e. fewer than 2^7 points), we use the multitaper method [61] to achieve better results.

[4]https://github.com/Olauwers/Extended-Cepstral-Distance

Algorithm 3 Algorithm for the extended cepstral distance

input : Two input/output signal pairs, (y_1, u_1) of length N_1, and (y_2, u_2) of length N_2

output : The extended cepstral distance $d_{C_e}((y_1, u_1), (y_2, u_2))$ between these two pairs, as defined in Subsection 3.3.2

for $i \leftarrow 1$ **to** 2 **do**

$\Phi_{u_i} \xleftarrow{\text{Welch's Method}^3} u_i$

$c_{u_i} \leftarrow \text{ifft}\left(\log\left(\Phi_{u_i}\right)\right)$

$\Phi_{y_i} \xleftarrow{\text{Welch's Method}} y_i$

$c_{y_i} \leftarrow \text{ifft}\left(\log\left(\Phi_{y_i}\right)\right)$

// c_{u_i} and c_{y_i} are vectors of length N_i

end

$w = [0, 1, \ldots, \max\{N_1, N_2\} - 1]$

add $(\max\{N_1, N_2\} - \min\{N_1, N_2\})$ 0's to the cepstra of the signal pair of length $\min\{N_1, N_2\}$

$d_{C_e}((y_1, u_1), (y_2, u_2)) \leftarrow w * \left((c_{y_1} - c_{u_1})^\intercal - (c_{y_2} - c_{u_2})^\intercal\right)^2$

3.4.1 Simulation set-up

To test the proposed techniques, we simulate data coming from electrical circuits. We start out by modelling two circuits with the same topology, but different values for the R, L, and C components. The topology was taken from a course on linear physical systems analysis [13]. The network topology and the values of the components are shown in Figure 3.1. The input of the system is the current i_u, the output is the voltage over L_2, e_y. State-space models of order 3 are then written down for these networks.

We provide both systems with 200 different input signals (100 outputs of LTI models of order 15, 50 multisine waves corrupted by Gaussian white noise with standard deviation of 0.1 and 50 white noise signals), and measure the outputs. This generates a dataset of 400 input/output signal pairs (200 inputs times 2 models). The question at hand is whether we can use only this input/output data, to determine which pairs were generated by the same system, i.e. cluster the dataset in two groups, defined by the generative dynamics.

We will do this using the distance measures from section 3.2 and subsection 3.3.1, keeping in mind that we use the lower bound (DTW LB) from [36] as an efficient lower bound to DTW. We compare to the technique developed in subsections 3.3.2 and 3.3.3, where we respectively gave a theoretical and computational overview of the distance measure.

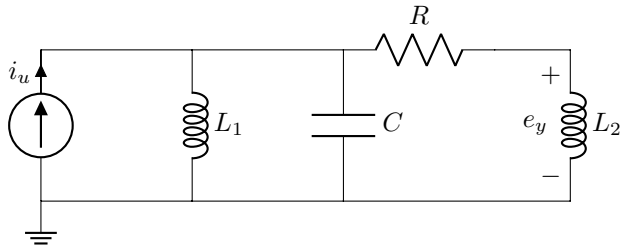


Figure 3.1: Electric circuit that was used for the experiments. Two sets, S_1 and S_2, of values were chosen for the components, namely $S_1 = \{R = 100\Omega, L_1 = 60\text{H}, L_2 = 20\text{H}, C = 50\text{F}\}$ and $S_2 = \{R = 100\Omega, L_1 = 160\text{H}, L_2 = 200\text{H}, C = 75\text{F}\}$. These two electrical circuits were used to perform the simulations in Section 3.4.

The performance of these simulations will be measured by the Adjusted Rand Index (ARI) [32, 64], which is a similarity measure between partitions. The ARI compares two partitions, S_1 and S_2, by calculating the ratio of pairs that have the same partitioning status (i.e. belonging to the same partition or not) in both S_1 and S_2 to the total amount of data pairs, then adjusting the resulting ratio by subtracting the expected value, to account for guessing (i.e. a partitioning that is the result of random guessing is assigned an ARI of 0). An ARI of 1 corresponds to perfectly similar partitions.

We compare the partitions generated by a hierarchical clustering method, cut-off at two clusters, using distance matrices generated by the different distance measures of section 3.2 and section 3.3 versus the ground truth (i.e. the time series was generated by the system with parameters S_1 or with parameters S_2, as in Figure 3.1).

3.4.2 Results

The results for the set-up in the previous subsection are shown in Figure 3.2, which shows the average and standard deviation for the ARI of the simulation results, and Figure 3.3, which shows the average and standard deviation for the execution time of the simulations.

The extended cepstral distance gives the best results, managing to cluster the simulated signal pairs almost perfectly every time. This is to be expected, as it was tailored specifically to take into account the dynamics of the underlying

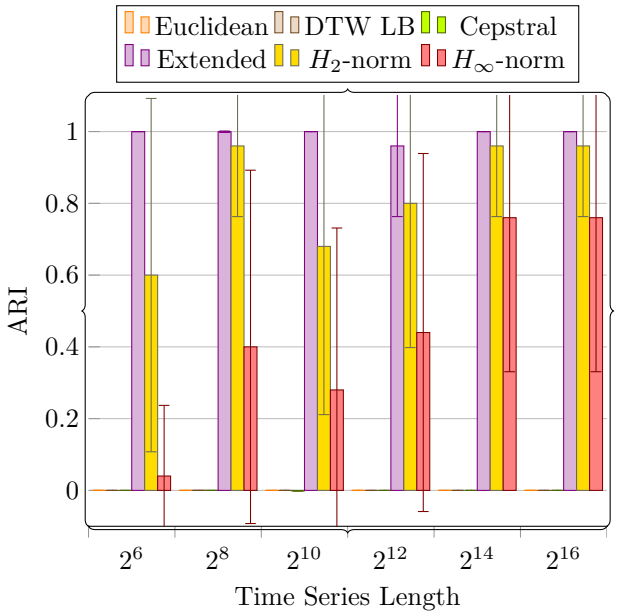


Figure 3.2: Performance of the different clustering algorithms, as measured by the ARI. For each time series length, shown on the x-axis, the average ARI over 100 experiments of finding 2 clusters in 400 time series is depicted as the height of the bar. The error bars show the standard deviation for the performance on these 100 experiments. Note that the Euclidean, DTW LB and cepstral distance have an ARI of 0, i.e., they amount to random guessing. The extended cepstral distance performs best for all series lengths. The model based distances were given a wrong model order, but still give good performance for longer time series.

model[5], and nothing but those dynamics. The reasons why it performs better than the other measures will be explained in what follows, and we again use the distinction between raw data and model-based distances measures from Section 3.2.

Raw Data Distance Measures

The reason the other raw data distance measures do not perform well on the problem at hand, is because they are not able to separate the input signal from

[5]We redid the experiments for generating systems of higher order, and the extended cepstral distance still performed best. Results were omitted.

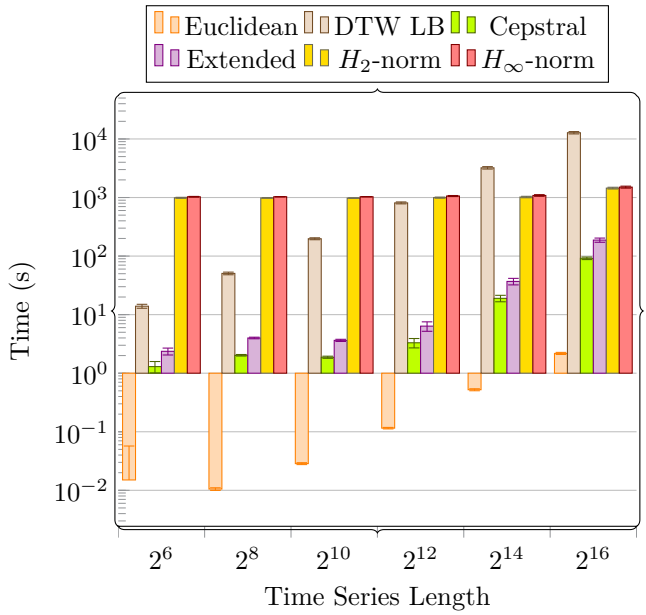


Figure 3.3: Execution time of the different clustering algorithms, measured in seconds. For each time series length, shown on the x-axis, the average time over 100 experiments of finding 2 clusters in 400 time series is depicted as the height of the bar. The error bars show the standard deviation for the execution time on these 100 experiments. Note that the y-axis has logarithmic scale. The extended cepstral distance remains several orders of magnitudes faster than the model-based distances. Note that DTW LB quickly becomes the computationally most expensive technique. The Euclidean distance is always fastest.

the output signal. The dynamics of the output are dominated by the input (the models generating the inputs are of higher order than the models describing the electrical circuits). If we only use white noise inputs the original cepstral distance performs better (see the left hand side in Figure 3.4).[6] Euclidean and DTW distances still do not deliver good results when detecting the difference in dynamics. Explicitly adding in the distance between the inputs did not improve results, which were omitted.[7]

[6]The original and extended cepstral distance are equivalent in this case, as the cepstrum of white noise is only non-zero in its zeroth component, which is omitted in the sum in equations (3.7) and (3.11), which then coincide.

[7]The code is still available on GitHub. The reader is encouraged to try out any adaptation to the distance measures presented.

There is no hope to achieve better results by taking the input signal into account in the Euclidean distance or the DTW distance, as these distances look at the shape of the signal – which does not solve the problem at hand –, but at its generative dynamics. DTW is better at this job [36], but has a big disadvantage: it takes a lot of time to compute, especially for long time series, where it even surpasses the model-based distance measures in computation time (see Figure 3.3).

The extended cepstral distance is thus preferred to cluster input/output signals by dynamics of their generating models.

Model-based Distance Measures

The model-based distance measures show better results than the raw data distance measures, and this again is to be expected. The model-based measures manage to peel out the information on the system that generated the input/output pair. However, a priori we have no information on the order of the underlying system, so we arbitrarily have to set a model order. In this case, we estimated transfer functions of order 5. If we share the correct model order (3) with the identification algorithm, performance of the model norms increases (Figure 3.4).

Schemes exist to estimate model orders, and more effort can be put in correctly identifying the underlying model. However, the model norm techniques are already several orders of magnitude slower than the extended cepstrum distance measure (Figure 3.3). For problems concerning large numbers of long input/output-pairs, as can be found in realistic problems in process industry (see, for example, [53], where more than 250 sensors make a measurement every 5 minutes for 6 months), this becomes highly impractical.

The extended cepstral distance is thus preferred over explicitly identifying systems, because of both being easier to automate, and taking less time to compute.

3.4.3 Non-linear loads

We performed additional tests on robustness against non-liner loads in the circuits, replacing constant inductors with saturating ones. Results depend heavily on saturation levels and signal length, but the extended cepstral distance still outperforms the others. We therefore do not show results here, but provide an interactive tutorial on the accompanying GitHub, where the reader

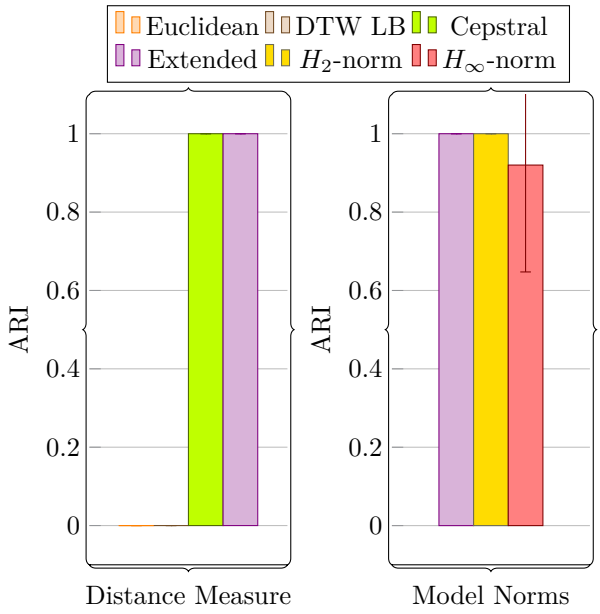


Figure 3.4: On the left, the performance is shown of the different raw data distance measures, as measured by the Adjusted Rand Index (ARI), in the case of white noise as an input, and time series of length 2^{10}. Here, the average over 100 experiments with 400 output signals is shown. Note that the original cepstral distance now shows the same performance as the extended one. On the right, results of an experiment where we provided the system identification step with the correct orders of the models are shown. Here, we calculated an average over 100 experiments with 40 output signals, to reduce computation time. Again, we simulated time series of length 2^{10}. The model-based distances now show better performance.

is encouraged to experiment with different values for the parameters of the non-linearity.

3.5 Conclusion and further research

We present a distance measure that is as insightful as a model norm-based distance, yet remains computationally much simpler than explicitly estimating models. It allows to meaningfully cluster large input/output signal pair datasets based exclusively on the dynamics of the generating systems. We have tested it

on a simulation of electrical circuits, where we started from two circuits with a current as input and a voltage difference over an inductor as output. We provided both circuits with 200 different inputs, resulting in 400 input/output pairs. The proposed measure performs as well as model-based distances on estimates of the generative systems, but is much easier to calculate. Other distance measures (Euclidean, DTW) perform much worse.

We furthermore show that, in the stochastic input case, the extended distance proposed in this paper reduces to the original cepstrum distance, which was proven ([19, 54]) to be equivalent to a model norm. This gives hope that the extended distance could also be linked to a model norm.

The results indicate the extended cepstral distance measure does a good job of capturing the dynamics of input/output pairs. An application to a real-life dataset is needed to validate the effectiveness in practice, but for the simulated problem at hand, the distance measure succeeded in perfectly distinguishing different dynamics based on raw data alone.

Further research should look at further extensions of the distance measure to more general classes of systems, such as nD systems and MIMO systems. It is not intuitively clear how the cepstrum should be defined in these cases, which necessitates further effort.

3.A Non-linear loads

This Appendix did not appear in the original paper. It is an addendum, going into more detail on the results on non-linear loads, which were omitted from the original paper due to page limits.

While the interpretation of the (extended) weighted cepstral distance assumes the underlying systems to be linear, it is interesting to look at how these results hold up in the case of non-linear systems. These experiments are shown in a Matlab Livescript on the accompanying GitHub-page, and we repeat some results here.

We start again from the circuit in Figure 3.1. Up till now, we have assumed all components to exhibit ideal, linear behaviour. Let us now investigate what happens if we replace the linear inductor by a more realistic, non-linear, saturating model of the inductor component, as in Figure 3.5. This inductor operates at a constant, maximum value if the absolute value of the current is lower than b. As soon as this limit is reached, the inductor starts saturating. It is saturated completely at a current of absolute value a, at which point the inductance becomes equal to 1, and remains that way.

The circuit in Figure 3.1 is governed by the following (continuous time) differential equations (see [13] for a derivation),

$$\dot{x} = \begin{bmatrix} 0 & \frac{1}{L_2} & 0 \\ -\frac{1}{C} & 0 & -\frac{1}{C} \\ 0 & \frac{1}{L_1} & -\frac{R}{L_1} \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{1}{C} \\ 0 \end{bmatrix} u, \tag{3.13}$$

with

$$x = \begin{bmatrix} i_{L_2} \\ e_C \\ i_{L_1} \end{bmatrix}, \tag{3.14}$$

where i_{L_1} is the current across L_1, i_{L_2} the current across L_2, and e_C the voltage over C. The output can then be written as

$$e_y = e_C - R i_{L_1}. \tag{3.15}$$

We now repeat the experiments above for two circuits with the same topology, but different values for the different components. Consider again the two

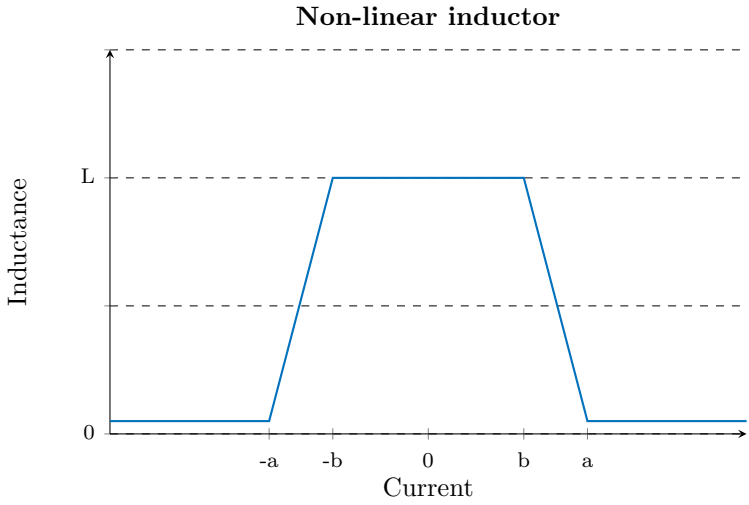


Figure 3.5: Inductance profile of the non-linear, saturating inductor considered in this Appendix. The inductor has a maximum value for the inductance, L, which it reaches for currents between $-b$ and b. At $\pm b$, the inductor starts saturating, and it is saturated completely at $\pm a$. When the saturation is hit, the inductance goes to 1.

systems, S_1 and S_2, with component values $S_1 = \{R = 100\Omega, L_1 = 60\text{H}, L_2 = 20\text{H}, C = 50\text{F}\}$ and $S_2 = \{R = 100\Omega, L_1 = 160\text{H}, L_2 = 200\text{H}, C = 75\text{F}\}$, respectively. The inductors now all have a saturation limit, as in Figure 3.5, and start saturating at a current of $b = 3\text{A}$, and are completely saturated at $a = 5\text{A}$.

In the linear case considered before, we could define this system within Matlab, discretize it, and run a linear simulation. In the non-linear case of the saturating inductor considered here, L_1 and L_2 become dependent on i_{L_1} and i_{L_2}, respectively. We get a set of differential equations, and have to resort to numerical integration methods.

This is computationally less efficient than in the linear case, and we therefore reduce the size of the input dataset. We create 20 signals coming from a linear system, 10 multisine waves corrupted with white noise, and 10 white noise signals. We feed these inputs to the two systems, and simulate their outputs. We then compute distance matrices as before, and cluster the signals using the distance measures described.

As the computation takes quite a while now, and results depend heavily on the

	ARI		ARI
d_e	-0.0012	d_{C_e}	1
d_{DTW}	0	$d_{\|\|\cdot\|\|_2}$	0
d_C	0.0573	$d_{\|\|\cdot\|\|_\infty}$	0

Table 3.1: ARI for the non-linear clustering experiments between two circuits, conforming to the topology of Figure 3.1, with parameters $S_1 = \{R = 100\Omega, L_1 = 60\text{H}, L_2 = 20\text{H}, C = 50\text{F}\}$ and $S_2 = \{R = 100\Omega, L_1 = 160\text{H}, L_2 = 200\text{H}, C = 75\text{F}\}$, respectively. The inductors start saturating at 3A and are completely saturated at 5A. We generated 40 input signals, simulated both systems with all these signals, and clustered the resulting input-output signals. These ARI's are for an experiment with the random number generator set at `rng('default')`.

exact parameters of the non-linearity anyway, we report results for the ARI of just one run. The reader is however, once more, encouraged to try out different set-ups for themselves. For these results, the random number generator was initialized at its default position (using the command `rng('default')`) at the start of the Matlab Livescript. These results are summarized in Table 3.1.

The results are quite striking, the extended cepstral distance is the only distance considered that still works! While we will not always get a perfect ARI equal to 1 for the extended cepstral distance when using different seeds for the random number generator, the general remark that it is the only distance measure giving reasonable results remains.

Shape-based and the original cepstral distance measure failed in the linear case already, when dealing with different inputs, and we now see that the norm-based distances fail in the non-linear case as well. This is because we are now trying to estimate a linear model from non-linear dynamics. Perhaps a change to non-linear model classes could help the norm-based results, but that is outside the scope of this text.

Chapter 4

Subspace angles between input/output systems

In Chapter 3, we introduced an extension to the stochastic weighted cepstral distance as it was introduced in [54] and interpreted in terms of subspace angles in [19], by employing the model power cepstrum in Definition 3.

The stochastic version, which only took into account output signals, was shown to be equivalent to a combination of model parameters in Theorem 1 in Section 2.5, and related to the product over the square cosines of the subspace angles in Theorem 2 in Subsection 2.7.3.

Furthermore, in the stochastic case, Equation (2.78) allowed to calculate the subspace angles, starting from output data, when considering two systems driven by the same white noise realization. This is a very restrictive limitation, which will often be violated in practice.

In this Chapter, we extend these interpretations to the case of systems with inputs and outputs. We will be able to find similar expressions both for the interpretation in terms of model parameters (Section 4.1), as for the subspace angles (Section 4.2).

We will discuss the computational aspects of calculating the subspace angles, and show a new result, where it is now possible to compute the subspace angles for systems with different persistently exciting inputs, from the data, provided the inputs and outputs are available, in Section 4.3.

These equivalences are summarized in Figure 4.1.

Section 4.4 will talk about systems with both poles and zeros, and unstable systems.

The equivalences above will be shown to hold numerically in Section 4.5. A discussion on computational aspects and robustness of the weighted cepstral distance is given in Section 4.6.

4.1 Deterministic weighted cepstral distance

Let us repeat the definition of the deterministic weighted cepstral distance, and extend the interpretation in terms of model parameters we had for the stochastic case.

In Chapter 3, we introduced the deterministic weighted cepstral distance (we also included it in Chapter 2, in Definition 3):

$$d_c^2\left(\left(y_1,u_1\right),\left(y_2,u_2\right)\right)=\sum_{k=1}^{\infty}k\left(c_{h_2}(k)-c_{h_1}(k)\right)^2. \tag{4.1}$$

Note that this extension reduces to the classical definition for

Autonomous systems Here, there is no input to the system, and we define the model cepstrum as $c_h(k)=c_y(k)$.

Stochastic systems Here, we have for the power cepstrum coefficients of a white noise signal, c_w, that $c_w(k)=0$, $\forall k\neq 0$. Therefore, $c_y(k)=c_h(k)$, $\forall k\neq 0$).

We interpret the weighted cepstral distance in terms of model parameters in the following theorem.

Theorem 5. *Given two input/output pairs (u_1,y_1) and (u_2,y_2), belonging to two stable all-pole models H_1 of order p and H_2 of order q, with poles α_i and β_i, respectively, it holds that*

$$d_c^2\left(\left(y_1,u_1\right),\left(y_2,u_2\right)\right)=\log\frac{\prod_{i=1}^{p}\prod_{j=1}^{q}\left|1-\alpha_i\bar{\beta}_j\right|^2}{\prod_{i,j=1}^{p}\left(1-\alpha_i\bar{\alpha}_j\right)\prod_{i,j=1}^{q}\left(1-\beta_i\bar{\beta}_j\right)}. \tag{4.2}$$

Given two n-th order all-pole systems, H_1 and H_2, with input/output pairs (u_1, y_1) and (u_2, y_2) respectively, we can calculate the kernel from Definition 6

$$\kappa\left((u_1, y_1), (u_2, y_2)\right) = \prod_{i=1}^{n} \cos^2(\theta_k),$$

with θ_k the subspace angles between the systems. We show a novel way to relate these angels to input and output data:

$$\lim_{i\to\infty} \lim_{j\to\infty} \left[Y_1 \Pi_{U_1}^{\perp} \triangleleft Y_2 \Pi_{U_2}^{\perp}\right] = \{\theta_k\},$$

with $\Pi_{U_i}^{\perp}$ the orthogonal projector on the orthogonal complement of the row space of U_i, in Theorem 7 in Subsection 4.3.1.
We will prove, in Lemma 5 in Subsection 4.1, that

$$\kappa\left((u_1, y_1), (u_2, y_2)\right) = \frac{\prod_{i,j=1}^{n} (1 - \alpha_i \bar{\alpha}_j) \prod_{i,j=1}^{n} \left(1 - \beta_i \bar{\beta}_j\right)}{\prod_{i=1}^{n} \prod_{j=1}^{n} \left|1 - \alpha_i \bar{\beta}_j\right|^2},$$

with α_i and β_i the poles of the generating dynamical systems, and, in Theorem 6 in Subsection 4.2.2, that

$$\kappa\left((u_1, y_1), (u_2, y_2)\right) = e^{-d_c^2((y_1,u_1),(y_2,u_2))},$$

with $d_c^2\left((y_1, u_1), (y_2, u_2)\right)$ the deterministic weighted cepstral distance.

Figure 4.1: We will study the extended weighted cepstral distance from Equation (3.11), and extend the equivalences that were proven for the stochastic case in this Chapter. We will extend the interpretation of the distance in terms of poles and zeros in Theorem 5. A kernel function on subspace angles is defined in Subsection 4.2.2. Using the equivalences above, we relate the weighted cepstral distance and subspace angles kernel in Theorem 6. A novel way to relate these angles to input and output signals is given in Theorem 7, and a computational way of finding them is given in 4.3.2.

Proof. As the derivation of the expression for the model cepstrum in Equation (2.31) only depends on the power spectral density of the system, Φ_h, the proof is completely analogous to that of Theorem 1.

In the deterministic case, we start with

$$\begin{aligned} c_{y_1}(k) - c_{u_1}(k) = c_{h_1}(k) &= \sum_{j=1}^{p} \frac{\alpha_j^{|k|}}{|k|}, \\ c_{y_2}(k) - c_{u_2}(k) = c_{h_2}(k) &= \sum_{j=1}^{q} \frac{\beta_j^{|k|}}{|k|}, \qquad \forall k \neq 0. \end{aligned} \tag{4.3}$$

We then again have for the weighted cepstral distance in Equation (4.1)

$$\begin{aligned} d_c^2\left((y_1,u_1),(y_2,u_2)\right) &= \sum_{k=1}^{\infty} k \left(\sum_{j=1}^{p} \frac{\alpha_j^k}{k} - \sum_{j=1}^{q} \frac{\beta_j^k}{k} \right)^2 \\ &= \sum_{j=1}^{p}\sum_{k=1}^{\infty} k \left(\frac{\alpha_j^k}{k}\right)^2 + \sum_{j=1}^{q}\sum_{k=1}^{\infty} k \left(\frac{\beta_j^k}{k}\right)^2 - 2\sum_{j=1}^{p}\sum_{l=1}^{q}\sum_{k=1}^{\infty} k \left(\frac{\alpha_j^k}{k}\frac{\beta_l^k}{k}\right) \\ &= \log \frac{\prod_{i=1}^{p}\prod_{j=1}^{q} \left|1 - \alpha_i \bar{\beta}_j\right|^2}{\prod_{i,j=1}^{p} \left(1 - \alpha_i \bar{\alpha}_j\right) \prod_{i,j=1}^{q} \left(1 - \beta_i \bar{\beta}_j\right)}, \end{aligned} \tag{4.4}$$

where in the last equality we used the series expansion from Equation (2.26). □

4.2 Subspace angles in the deterministic case

In the deterministic case, we can still define the subspace angles between two models in terms of their observability matrices. For the subspace angles, θ_k, of two n-th order minimal, stable, all-pole, SISO LTI input-output models, with observability matrices $\Gamma_1^i, \Gamma_2^i \in \mathbb{R}^{i \times n}$, respectively, we still have:

$$\{\theta_k\} = \lim_{i \to \infty} \left[\Gamma_1^i \lhd \Gamma_2^i\right]. \tag{4.5}$$

We can again build a positive-definite kernel from these subspace angles, as we did for the stochastic case in Subsection 2.7.2.

4.2.1 Kernel functions

A positive-definite kernel function is a type of similarity function that is used in many machine learning applications.

Formally, for an input space $\mathcal{X}$, a kernel κ is a function

$$\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}. \tag{4.6}$$

The input space $\mathcal{X}$ is not restricted to vector spaces.

If κ fulfills Mercer's condition [56], i.e. for all square integrable functions $g(x)$, it holds that,

$$\iint g(x_1)\kappa(x_1, x_2)g(x_2)dx_1dx_2 \geq 0, \tag{4.7}$$

we can rewrite the kernel in terms of a feature map $\phi : \mathcal{X} \mapsto \mathcal{V}$, mapping an object, $x \in \mathcal{X}$, to a set of features of this object, $\phi(x)$.

Given two elements of the input space $\mathcal{X}$, x_1 and x_2, and a proper inproduct $\langle \cdot, \cdot \rangle$ on the feature space $\mathcal{V}$, we can then write the kernel

$$\kappa(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle. \tag{4.8}$$

This allows us then to work in the feature space $\mathcal{V}$ directly, instead of in the input space $\mathcal{X}$, which can in many cases considerably simplify computations. In kernel methods literature, this is known as the *kernel trick*.

In the discrete case, this kernel then becomes a matrix, and Mercer's condition assures this matrix is positive-semidefinite.

We will not get into all the details here, but refer to [29] for an overview of the use of kernel methods in machine learning, and to [73] for their use in support vector machines. Results on the use of kernel methods for dynamical models were discussed in [81]. For the present discussion, it suffices to state the importance of these kernels in quantifying similarity in modern machine learning algorithms.

4.2.2 The subspace angles kernel

In Subsection 2.7.3, we introduced a kernel based on subspace angles, for minimal, stable, all-pole, SISO LTI, n-th order systems H_1 and H_2

$$\kappa(H_1, H_2) = \prod_{k=1}^{n} \cos^2(\theta_k), \tag{4.9}$$

with $\{\theta_k\} = \lim_{i\to\infty} \left[\Gamma_1^i \triangleleft \Gamma_2^i\right]$ the n subspace angles between the systems. It was shown in [88] that this kernel is indeed positive-definite.

The subspace angles kernel readily allows for systems with inputs and outputs. Theorem 2 in Subsection 2.7.3, linking the weighted cepstral distance to the subspace angles kernel can then be extended to the input-output case:

Theorem 6. *Given two n-th order minimal, stable, all-pole, input-output, SISO LTI models H_1 and H_2, with n subspace angles $\{\theta_k\} = \lim_{i\to\infty} \left[\Gamma_1^i \triangleleft \Gamma_2^i\right]$ between them, the weighted cepstral distance between them is related to the subspace angles kernel as follows:*

$$d_c^2(H_1, H_2) = -\log \kappa(H_1, H_2) = -\log \prod_{k=1}^{n} \cos^2(\theta_k). \tag{4.10}$$

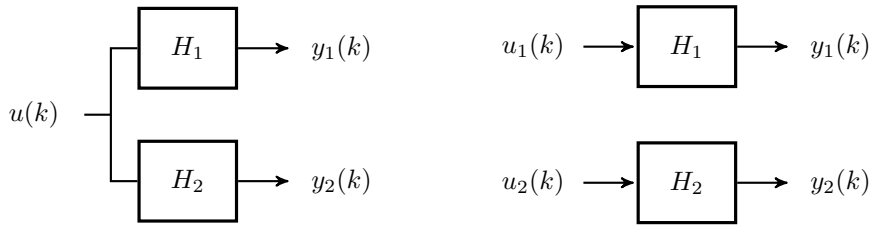
Proof. Lemma 3 related the column space of the observability matrix to that of a Vandermonde matrix of the poles. Proving this Lemma required only the observability and state space matrices, which have not changed by including different inputs. The proof for the input-output case is then analogous to the one for stochastic systems. □

4.3 Computing the subspace angles

While we have shown we can calculate the subspace angles kernel by computing the weighted cepstral distance, we will show an explicit way of computing the subspace angles from input-output data alone, without knowing the underlying models.

Let us start by repeating the definition of the Hankel matrix of a signal. We denote by $Y^{i,j}$ the $i \times j$ Hankel matrix composed of i rows of consecutive elements of the time series y, starting at $y(0)$, where j is the number of columns considered ($i \ll j$). This matrix is thus constructed as follows:

$$Y^{i,j} = \begin{pmatrix} y(0) & y(1) & y(2) & \cdots & y(j-1) \\ y(1) & y(2) & y(3) & \cdots & y(j) \\ y(2) & y(3) & y(4) & \cdots & y(j+1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y(i-1) & y(i) & y(i+1) & \cdots & y(i+j-2) \end{pmatrix}. \tag{4.11}$$



(a) Input-output behaviour in the stochastic case.

(b) Input-output behaviour in the deterministic case.

Figure 4.2: The difference in input and output behaviour between the stochastic and the deterministic case. In the stochastic case, in 4.2a, the two systems are driven by the same input sequence. The deterministic case allows for different inputs for the two systems, as in 4.2b.

For two stochastic systems with output Hankel matrices $Y_1^{i,j}$ and $Y_2^{i,j}$, and observability matrices Γ_1^i and Γ_2^i, driven by the same white noise realization, we had (Equation (2.78))

$$\lim_{j\to\infty}\left[(Y_1^{i,j})^\intercal \lhd (Y_2^{i,j})^\intercal\right] = \lim_{i\to\infty}\left[\Gamma_1^i \lhd \Gamma_2^i\right], 0, \ldots, 0. \tag{4.12}$$

The constraint that both systems have to be driven by the same white noise realization is very limiting, and will not often be fulfilled in practice.

In this Section, we introduce a novel way to compute the subspace angles between models driven by different input signals, from input and output Hankel matrices. The difference between the stochastic and the deterministic case is made clear in Figure 4.2.

4.3.1 Subspace angles from input and output signals

The subspace angles are the principal angles between the observability matrices of two systems, as in Equation (4.5). Let us then first propose a way to compute the column space of the observability matrix in terms of input and output data. To do so, we need to define projection matrices.

We define the orthogonal projector on the row space of a matrix $U \in \mathbb{R}^{i\times j}$ as

$$\Pi_U = U^\intercal(UU^\intercal)U, \tag{4.13}$$

which is a $j \times j$ symmetric idempotent matrix of rank i. Similarly, the orthogonal projector on the orthogonal complement of the row space of U is defined as

$$\Pi_U^\perp = \mathbb{1} - \Pi_U, \tag{4.14}$$

which is again a $j \times j$ symmetric idempotent matrix, now of rank $j - i$.

Using the orthogonal projector matrix on the orthogonal complement of the row space of the input Hankel matrices, we can compute the subspace angles as in the following Lemma.

Theorem 7. *Consider two n-th order, minimal, stable, all-pole, LTI SISO models, H_1 and H_2, with observability matrices Γ_1^i and Γ_2^i, respectively. For H_1 (H_2) we have a persistently exciting input $u_1(k)$ ($u_2(k)$), which gives an output $y_1(k)$ ($y_2(k)$); these signals are associated with Hankel matrices $U_1^{i,j}$ ($U_2^{i,j}$) and $Y_1^{i,j}$ ($Y_2^{i,j}$). We then have for the n subspace angles, θ_k, between these models,*

$$\begin{aligned} &\lim_{i\to\infty}\lim_{j\to\infty}\left[\left(Y_1^{i,j}\Pi_{U_1^{i,j}}^{\perp}\right) \lhd \left(Y_2^{i,j}\Pi_{U_2^{i,j})}^{\perp}\right)\right] \\ &= \lim_{i\to\infty}\left[\Gamma_1^i \lhd \Gamma_2^i\right] \\ &= \{\theta_k\}\,. \end{aligned} \tag{4.15}$$

Proof. Starting from the state space equations in Equation (2.1), we can describe the relation between Hankel matrices (equation (2.77)) of inputs and outputs of a system, H, by the formula (see [78])

$$\begin{pmatrix} U^{i,j} \\ Y^{i,j} \end{pmatrix} = \begin{pmatrix} \mathbb{1}_i & 0 \\ M^i & \Gamma^i \end{pmatrix} \begin{pmatrix} U^{i,j} \\ X^j \end{pmatrix}, \tag{4.16}$$

where $X^j = \begin{pmatrix} x_0 & x_1 & \cdots & x_{j-1} \end{pmatrix}$, the state sequence of the system, and M^i is a lower triangular Toeplitz matrix

$$M^i = \begin{pmatrix} D & 0 & 0 & \cdots & 0 \\ CB & D & 0 & \cdots & 0 \\ CAB & CB & D & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{i-2}B & CA^{i-3}B & CA^{i-4}B & \cdots & D \end{pmatrix}, \tag{4.17}$$

with A, B, C and D, the state space matrices. For an extensive discussion of the subspace framework, see [78].

The last block row of Equation (4.16) gives us

$$Y^{i,j} = M^i U^{i,j} + \Gamma^i X^j. \tag{4.18}$$

For ease of notation, we now drop the superscripts

We assumed persistency of excitation, which implies $\text{rank}(U) = \text{rank}(Y) = i$, with i the amount of rows of the Hankel matrices. Furthermore, we assume that the row space of X does not intersect with the row space of U, so that

$$\text{rank} \begin{pmatrix} U \\ X \end{pmatrix} = i + n, \tag{4.19}$$

with n the order of the system. This condition is generally satisfied.

With the orthogonal projector, $\Pi_U^\perp$ on the row space of the input Hankel matrix, U, as defined in Equation (4.14), we can find from Equation (4.18),

$$Y\Pi_U^\perp = \Gamma X \Pi_U^\perp. \tag{4.20}$$

It then follows that the column space of the observability matrix Γ is equal to the column space of the output Hankel matrix projected on the orthogonal complement of the row space of the input Hankel matrix:

$$\text{range}(Y^{i,j}\Pi_{U^{i,j}}^\perp) = \text{range}(\Gamma^i). \tag{4.21}$$

Consider now two n-th order, minimal, stable, all-pole, LTI SISO models, H_1 and H_2, with observability matrices Γ_1^i and Γ_2^i, respectively. For H_1 (H_2) we have a persistently exciting input $u_1(k)$ ($u_2(k)$), which gives an output $y_1(k)$ ($y_2(k)$); these signals are associated with Hankel matrices $U_1^{i,j}$ ($U_2^{i,j}$) and $Y_1^{i,j}$ ($Y_2^{i,j}$).

We then have

$$\begin{aligned} \text{range}(Y_1^{i,j}\Pi_{U_1^{i,j}}^\perp) &= \text{range}(\Gamma_1^i), \\ \text{range}(Y_2^{i,j}\Pi_{U_2^{i,j}}^\perp) &= \text{range}(\Gamma_2^i). \end{aligned} \tag{4.22}$$

As the subspace angles θ_k were defined, in Definition 5, as

$$\theta_k = \lim_{i\to\infty} \left[\Gamma_1^i \lhd \Gamma_2^i\right], \tag{4.23}$$

we now find an expression of these subspace angles in terms of input and output Hankel matrices, with more columns than rows (i.e. $i << j$),

$$\begin{aligned} &\lim_{i\to\infty}\lim_{j\to\infty}\left[\left(Y_1^{i,j}\Pi_{U_1^{i,j}}^\perp\right) \lhd \left(Y_2^{i,j}\Pi_{U_2^{i,j}}^\perp)\right)\right] \\ &= \lim_{i\to\infty}\left[\Gamma_1^i \lhd \Gamma_2^i\right] \\ &= \{\theta_k\}. \end{aligned} \tag{4.24}$$

□

In the next subsections, we will propose an algorithm to compute these angles.

4.3.2 Computing the column space of the observability matrix from input and output signals: LQ decomposition

We propose an algorithm to compute the column space of the observability matrix of a system, starting from input and output data, based on the LQ-decomposition[1]. We will never need to compute the Q-matrix explicitly. We show that the singular value decomposition (SVD) of part of the L-matrix will give us a representation of the relevant column space.

Consider the LQ-decomposition of the Hankel matrices of the input, $U^{i,j}$, and the output, $Y^{i,j}$ (dropping the superscripts for clarity),

$$\begin{pmatrix} U \\ Y \end{pmatrix} = \begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \end{pmatrix} \begin{pmatrix} Q_1^\intercal \\ Q_2^\intercal \\ Q_3^\intercal \end{pmatrix}, \tag{4.25}$$

where the blocks $L_{11}, L_{22} \in \mathbb{R}^{i\times i}$ are lower triangular, and $L_{21} \in \mathbb{R}^{i\times i}$. The matrix $Q = \begin{pmatrix} Q_1 & Q_2 & Q_3 \end{pmatrix}$ is orthonormal, such that $Q_1^\intercal Q_1 = Q_2^\intercal Q_2 = \mathbb{1}_i$, and $Q_1^\intercal Q_2 = 0$. We have $Q_1, Q_2 \in \mathbb{R}^{j\times i}$, and $Q_3 \in \mathbb{R}^{j\times j-2i}$.

From

$$U = L_{11} Q_1^\intercal, \tag{4.26}$$

where we again use the generic assumption that all matrices involved have rank i, we see that the rows of $Q_1^\intercal$ generate an orthonormal basis for the row space of U.

Similarly

$$Y = L_{21} Q_1^\intercal + L_{22} Q_2^\intercal. \tag{4.27}$$

The first term contains row vectors belonging to the row space of U, while the row vectors of the second term are all perpendicular to the row space of U, or

$$\Pi_U = Q_1 Q_1^\intercal \tag{4.28}$$

$$\Pi_U^\perp = \mathbb{1}_j - Q_1 Q_1^\intercal. \tag{4.29}$$

This means

$$Y \Pi_U^\perp = L_{22} Q_2^\intercal, \tag{4.30}$$

[1] This is the transpose of the QR-decomposition, see [28].

and therefore, from Equation (4.21),

$$\text{range}(Y\Pi_U^{\perp}) = \text{range}(\Gamma^i) = \text{range}(L_{22}). \tag{4.31}$$

Taking the singular value decomposition of L_{22} then gives us

$$L_{22} = WSZ^{\intercal} = \begin{pmatrix} W_1 & W_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} Z_1^{\intercal} \\ Z_2^{\intercal}, \end{pmatrix} \tag{4.32}$$

with $\text{rank}(S) = n$, the order of the underlying system, and $W_1 \in \mathbb{R}^{i \times n}$, $W_2 \in \mathbb{R}^{i \times n}$. We then have that

$$\text{range}(Y^{i,j}\Pi_{U^{i,j}}^{\perp}) = \text{range}(\Gamma^i) = \text{range}(W_1). \tag{4.33}$$

Starting from input and output Hankel matrices of two systems H_1 and H_2 fulfilling the conditions in Theorem 7, this is then an algorithm to compute the column spaces of their observability matrices.

Example 8. We show this computational method with an example. Consider the discrete, stable, all-pole models

$$H_1 = \frac{1}{(z-0.7)(z-0.8)}, \tag{4.34}$$

$$H_2 = \frac{1}{(z-0.1)(z-0.3)}, \tag{4.35}$$

with sampling time $T_s = 10^{-8}$.

We compute their subspace angles first by explicitly constructing the Vandermonde matrix of their poles, as in Lemma 3,

$$\Gamma_{1,V}^i = \begin{pmatrix} 1.0000 & 1.0000 \\ 0.7000 & 0.8000 \\ 0.4900 & 0.6400 \\ 0.3430 & 0.5120 \\ 0.2401 & 0.4096 \\ \vdots & \vdots \end{pmatrix} \qquad \Gamma_{2,V}^i = \begin{pmatrix} 1.0000 & 1.0000 \\ 0.1000 & 0.3000 \\ 0.0100 & 0.0900 \\ 0.0010 & 0.0270 \\ 0.0001 & 0.0081 \\ \vdots & \vdots \end{pmatrix}, \tag{4.36}$$

where the amount of rows, $i = 1000$. We compute the QR-decompositions, $\Gamma_{1,V}^i = Q_1 R_1$ and $\Gamma_{2,V}^i = Q_2 R_2$. The cosines of the subspace angles can now be computed as the singular values of $Q_1^{\intercal} Q_2$ (more efficient methods that do not explicitly calculate the Q's exist, see e.g. [88]; for the didactical purposes of the relatively small example here, this does not matter too much).

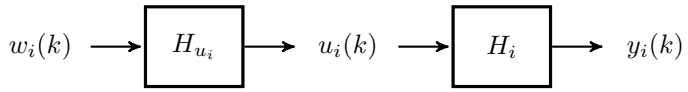


Figure 4.3: We generate signals for system H_i by applying a white noise signal, $w_i(k)$ to a random input system H_{u_i}, generating an input $u_i(k)$. This input is then fed to the system H_i, to generate an output $y_i(k)$.

The squared cosines of the subspace angles calculated in this way are

$$\begin{aligned} \cos^2 \theta_1 &= 0.9847, \\ \cos^2 \theta_2 &= 0.3431. \end{aligned} \tag{4.37}$$

We generate data as in Figure 4.3, where we construct two stable, all-pole input systems of order 2, $H_{u_1} = \frac{1}{z^2+1.24z+0.6808}$, and $H_{u_2} = \frac{1}{z^2+0.3245z+0.04538}$, and apply white noise to them, to get two persistently exciting input signals u_1 and u_2.

Applying u_1 (u_2) to H_1 (H_2) gives us the output signal y_1 (y_2). All signals are of length $N = 2^{10} = 1024$.

Now, we construct Hankel matrices $U_1^{102,923}$, $Y_1^{102,923}$, $U_2^{102,923}$, $Y_2^{102,923}$ from these signals. Following the computational procedure based on the LQ-decomposition above, we get the following estimates for the squared cosines of the subspace angles

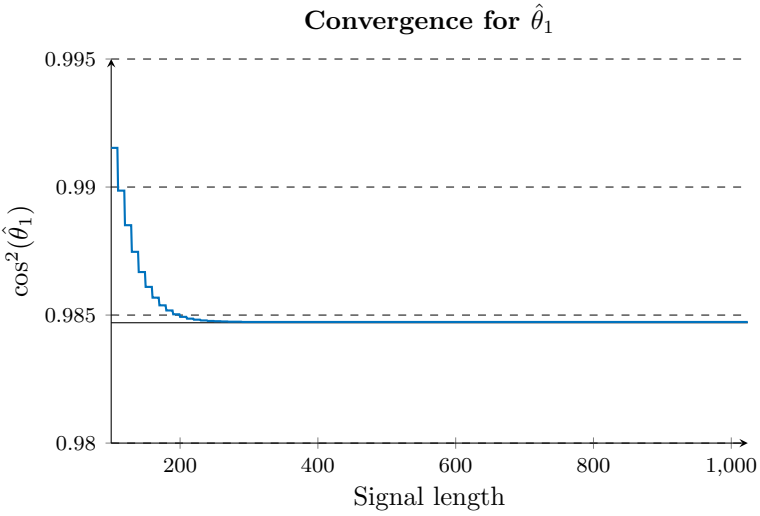
$$\begin{aligned} \cos^2 \hat{\theta}_1 &= 0.9847, \\ \cos^2 \hat{\theta}_2 &= 0.3431, \end{aligned} \tag{4.38}$$

in good accordance with the estimate of the cosines obtained from the Vandermonde matrices of the poles in Equation (4.37).

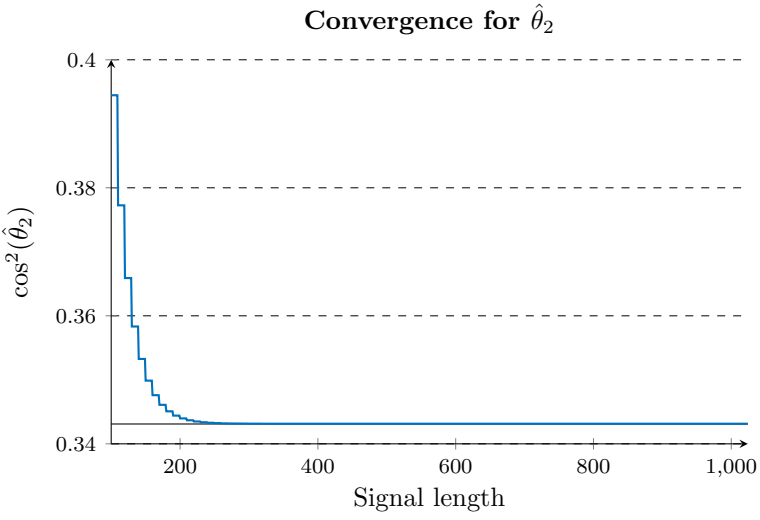
We investigate the convergence of our procedure for different signal lengths, where the amount of rows is taken as the nearest integer below 10% of the chosen signal length, $i = \lfloor \frac{N_l}{10} \rfloor$, and the chosen signal length goes from $N_l = 100$ to $N_l = N$. Results are shown in Figure 4.4.

4.4 Model assumptions

Just like in the stochastic case, we can extend these results to models with poles and zeros, and to completely unstable systems.



(a)



(b)

Figure 4.4: Convergence behaviour for the squared cosines of the subspace angles between the systems in Equation (4.35), using the algorithm presented in Subsection 4.3.2, for different signal lengths, N_l. The amount of rows of the data Hankel matrices was chosen $i = \lfloor \frac{N_l}{10} \rfloor$. The black horizontal lines represent the estimates of the true values of the squared cosines of the angles, as calculated through the Vandermonde matrices. We see that, from signals of length about 300, the estimates have stabilized.

4.4.1 Models with poles and zeros

We have developed a way to interpret the weighted cepstral distance in terms of the subspace angles kernel for deterministic all-pole models. Just as in the stochastic case, we can now extend this to systems with both poles and zeros.

Consider two minimal, stable and inversely stable, LTI SISO pole-zero models, H_1 of order n_1, and H_2 of order n_2, with observability matrices Γ_1^i and Γ_2^i, respectively. The inverse systems have observability matrices $\Gamma_{1,\text{inv}}^i$ and $\Gamma_{2,\text{inv}}^i$. We then have, for the $n_1 + n_2$ subspace angles, θ_k, between H_1 and H_2,

$$\{\theta_k\} = \lim_{i\to\infty} \left[\begin{pmatrix}\Gamma_1^i & \Gamma_{2,\text{inv}}^i\end{pmatrix} \lhd \begin{pmatrix}\Gamma_2^i & \Gamma_{1,\text{inv}}^i\end{pmatrix}\right]. \tag{4.39}$$

We can now find an expression for these subspace angles in terms of input and output Hankel matrices, in the following theorem.

Theorem 9. *Consider two minimal, stable and inversely stable, pole-zero, LTI SISO models, H_1 and H_2, with observability matrices Γ_1^i and Γ_2^i, respectively. The observability matrices of the inverse systems are $\Gamma_{1,inv}^i$ and $\Gamma_{2,inv}^i$. For H_1 (H_2) we have a persistently exciting input $u_1(k)$ ($u_2(k)$), which gives an output $y_1(k)$ ($y_2(k)$); these signals are associated with Hankel matrices $U_1^{i,j}$ ($U_2^{i,j}$) and $Y_1^{i,j}$ ($Y_2^{i,j}$). We can then find the subspace angles, θ_k between these models as combinations of input and output Hankel matrices, as*

$$\begin{aligned}
&\lim_{i\to\infty}\lim_{j\to\infty}\left[\left(Y_1^{i,j}\Pi_{U_1^{i,j}}^{\perp} \quad U_2^{i,j}\Pi_{Y_2^{i,j}}^{\perp}\right) \lhd \left(Y_2^{i,j}\Pi_{U_2^{i,j}}^{\perp} \quad U_1^{i,j}\Pi_{Y_1^{i,j}}^{\perp}\right)\right] \\
&= \lim_{i\to\infty}\lim_{j\to\infty}\left[\begin{pmatrix}\Gamma_1^i & \Gamma_{2,inv}^i\end{pmatrix} \lhd \begin{pmatrix}\Gamma_2^i & \Gamma_{1,inv}^i\end{pmatrix}\right], \\
&= \{\theta_k\}.
\end{aligned} \tag{4.40}$$

Proof. The fact that $\text{range}(Y^{i,j}\Pi_{U^{i,j}}^{\perp}) = \text{range}(\Gamma^i)$ was proven in Theorem 7. We now still have to prove that

$$\text{range}(U\Pi_Y^{\perp}) = \text{range}(\Gamma_{\text{inv}}). \tag{4.41}$$

As the system was assumed to be invertible (i.e. $D \neq 0$), we have that M^i is invertible as well. We find for the inverse system, from Equation (4.18),

$$U^{i,j} = -{M^i}^{-1}\Gamma^i X + {M^i}^{-1} Y, \tag{4.42}$$

where now $-{M^i}^{-1}\Gamma^i = \Gamma_{\text{inv}}^i$, the observability matrix of the inverse system.

We leave out superscripts in what follows, for ease of notation.

We assume that the row space of X does not intersect with the row space of Y, so that

$$\operatorname{rank}\begin{pmatrix} Y \\ X \end{pmatrix} = i + n, \tag{4.43}$$

with n the order of the system. This condition is generally satisfied.

With the orthogonal projector, $\Pi_Y^\perp$ on the orthogonal complement of the row space of the output Hankel matrix, Y, as defined in Equation (4.14), we can find from Equation (4.42),

$$U\Pi_Y^\perp = \Gamma_{\text{inv}} X \Pi_Y^\perp. \tag{4.44}$$

It then follows that the column space of the observability matrix of the inverse system, Γ_{inv}, is equal to the column space of the input Hankel matrix projected on the orthogonal complement of the row space of the output Hankel matrix:

$$\operatorname{range}(U_Y^\perp) = \operatorname{range}(\Gamma_{\text{inv}}). \tag{4.45}$$

Consider two minimal, stable and inversely stable, pole-zero, LTI SISO models, H_1 and H_2, with observability matrices Γ_1^i and Γ_2^i, respectively. The observability matrices of the inverse systems are $\Gamma_{1,\text{inv}}^i$ and $\Gamma_{2,\text{inv}}^i$. For H_1 (H_2) we have a persistently exciting input $u_1(k)$ ($u_2(k)$), which gives an output $y_1(k)$ ($y_2(k)$); these signals are associated with Hankel matrices $U_1^{i,j}$ ($U_2^{i,j}$) and $Y_1^{i,j}$ ($Y_2^{i,j}$). We then have

$$\begin{aligned} \operatorname{range}(U_1^{i,j}\Pi_{Y_1^{i,j}}^\perp) &= \operatorname{range}(\Gamma_{1,\text{inv}}^i), \\ \operatorname{range}(U_2^{i,j}\Pi_{Y_2^{i,j}}^\perp) &= \operatorname{range}(\Gamma_{2,\text{inv}}^i). \end{aligned} \tag{4.46}$$

Together with Equation (4.22), this then gives the relation between input and output Hankel matrices and subspace angles, θ_k,

$$\begin{aligned} &\lim_{i\to\infty}\lim_{j\to\infty}\left[\left(Y_1^{i,j}\Pi_{U_1^{i,j}}^\perp \quad U_2^{i,j}\Pi_{Y_2^{i,j}}^\perp\right) \lhd \left(Y_2^{i,j}\Pi_{U_2^{i,j}}^\perp \quad U_1^{i,j}\Pi_{Y_1^{i,j}}^\perp\right)\right] \\ &= \lim_{i\to\infty}\left[\left(\Gamma_1^i \quad \Gamma_{2,\text{inv}}^i\right) \lhd \left(\Gamma_2^i \quad \Gamma_{1,\text{inv}}^i\right)\right], \\ &= \{\theta_k\}. \end{aligned} \tag{4.47}$$

□

A completely analogous computation to the one in Subsection 4.3.2, now starting from the LQ-decomposition of the matrix,

$$\begin{pmatrix} Y \\ U \end{pmatrix}, \tag{4.48}$$

gives us a way to compute range(Γ_{inv}).

4.4.2 Stability

Like we did for the interpretation in terms of poles and zeros for the stochastic case in Subsection 2.6.2, we can relax the assumptions of stability and inverse stability we made for the interpretation in subspace angles considered in this system.

For two systems with observability matrices Γ_1^i and Γ_2^i, we can write the column spaces of these matrices as Vandermonde-matrices $\Gamma_{1,V}^i$ and $\Gamma_{2,V}^i$, as in Lemma 3. The squared cosines of the subspace angles are then calculated as squared cosines between column spaces of these matrices.

Consider them the inversion

$$z \mapsto z^{-1} \tag{4.49}$$

of the complex plane over the unit circle .

This is a conformal map for these column vectors, and their angles remain the same under this mapping, as we can easily see from the following derivation.

Consider two columns of a Vandermonde matrix like the one in Lemma 3,

$$c = \begin{pmatrix} 1 & \gamma & \gamma^2 & \cdots & \gamma^{i-1} \end{pmatrix}^\intercal \tag{4.50}$$

$$d = \begin{pmatrix} 1 & \delta & \delta^2 & \cdots & \delta^{i-1} \end{pmatrix}^\intercal, \tag{4.51}$$

with $|\gamma| > 1$ and $|\delta| > 1$ two poles of magnitude greater than 1.

The cosine of the angle between these two vectors is

$$\cos^2 [c \sphericalangle d] = \frac{|c^\intercal d|^2}{|c^\intercal c||d^\intercal d|} = \lim_{i\to\infty} \frac{\left(\sum_{k=0}^{i-1} (\gamma\delta)^k\right)^2}{\sum_{k=0}^{i-1} \gamma^{2k} \sum_{k=0}^{i-1} \delta^{2k}}. \tag{4.52}$$

For $|x| \neq 1$, we have for the finite geometric sum

$$\sum_{k=0}^{i-1} x^k = \frac{1-x^i}{1-x}, \tag{4.53}$$

which we can use to rewrite the equation above

$$\cos^2[c \sphericalangle d] = \lim_{i\to\infty} \frac{\left(1-(\gamma\delta)^i\right)^2}{(1-\gamma\delta)^2} \left(\frac{1-\gamma^{2i}}{1-\gamma^2}\frac{1-\delta^{2i}}{1-\delta^2}\right)^{-1} \tag{4.54}$$

$$= \lim_{i\to\infty} \frac{(1-\gamma^2)(1-\delta^2)}{(1-\gamma\delta)^2} \frac{1-2(\gamma\delta)^i+(\gamma\delta)^{2i}}{1-\gamma^{2i}-\delta^{2i}+(\delta\gamma)^{2i}}. \tag{4.55}$$

Dividing the numerator and denominator by $(\gamma\delta)^{2i}$, we get

$$\cos^2[c \sphericalangle d] = \frac{(1-\gamma^2)(1-\delta^2)}{(1-\gamma\delta)^2} \lim_{i\to\infty} \frac{\frac{1}{(\gamma\delta)^{2i}} - \frac{2}{(\gamma\delta)^i} + 1}{\frac{1}{(\gamma\delta)^{2i}} - \frac{1}{(\gamma)^{2i}} - \frac{1}{(\delta)^{2i}} + 1}. \tag{4.56}$$

Now taking the limit of $i \to \infty$, all terms involving a power of i or $2i$ vanish, and we are left with

$$\cos^2[c \sphericalangle d] = \frac{(1-\gamma^2)(1-\delta^2)}{(1-\gamma\delta)^2}. \tag{4.57}$$

Dividing numerator and denominator by $-\gamma^2\delta^2$, we get

$$\cos^2[c \sphericalangle d] = \frac{(1-\delta^{-2})(1-\gamma^{-2})}{(1-\gamma^{-1}\delta^{-1})^2}. \tag{4.58}$$

Let us now consider the angles between these rows after the conformal map in Equation (4.49). We then have $|\gamma^{-1}| < 1$ and $|\delta^{-1}| < 1$, and

$$c_{\text{inv}} = \begin{pmatrix} 1 & \gamma^{-1} & \gamma^{-2} & \cdots & \gamma^{-(i-1)} \end{pmatrix}^\intercal, \tag{4.59}$$

$$d_{\text{inv}} = \begin{pmatrix} 1 & \delta^{-1} & \delta^{-2} & \cdots & \delta^{-(i-1)} \end{pmatrix}^\intercal, \tag{4.60}$$

The squared cosine of the angle between these vectors is

$$\cos^2[c_{\text{inv}} \sphericalangle d_{\text{inv}}] = \frac{|c_{\text{inv}}^\intercal d_{\text{inv}}|^2}{|c_{\text{inv}}^\intercal c_{\text{inv}}||d_{\text{inv}}^\intercal d_{\text{inv}}|} = \lim_{i\to\infty} \frac{\left(\sum_{k=0}^{i-1}(\gamma^{-1}\delta^{-1})^k\right)^2}{\sum_{k=0}^{i-1}\gamma^{-2k}\sum_{k=0}^{i-1}\delta^{-2k}}. \tag{4.61}$$

We can carry out the limit by noting that for $|x| < 1$, the infinite geometric sum converges as

$$\sum_{k=0}^{\infty} = \frac{1}{1-x}. \tag{4.62}$$

As both $|\gamma^{-1}| < 1$ and $|\delta^{-1}| < 1$, we get

$$\cos^2\left[c_{\text{inv}} \triangleleft d_{\text{inv}}\right] = \frac{(1-\delta^{-2})(1-\gamma^{-2})}{(1-\gamma^{-1}\delta^{-1})^2} = \cos^2\left[c \triangleleft d\right], \tag{4.63}$$

showing that the squared cosines of the columns of the Vandermonde matrices remain the same under an inversion of the plane as in Equation (4.49).

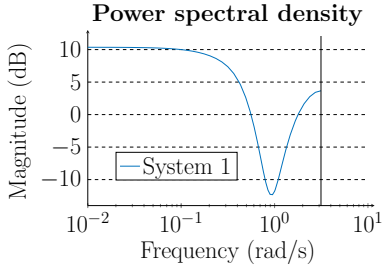
The relation between the weighted cepstral distance and the positive-definite kernel then carries over to the case of unstable systems. Subspace angles between systems with poles of mixed stability are at the moment not defined.

4.5 Numerical illustration of the relation between the weighted cepstral distance and the subspace angles kernel

To corroborate the equivalences and relations from this Chapter, summarized in Figure 4.1, numerically, we start with the two stable and inversely stable pole-zero models:

System 1:

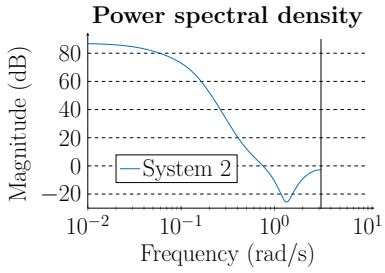
$$H_1 = \frac{z^5 - 2.9z^4 + 3.71z^3 - 2.609z^2 + 0.982z - 0.1525}{z^5 - 2.8z^4 + 3.13z^3 - 1.738z^2 + 0.4752z - 0.0504}, \tag{4.64}$$



Poles	Zeros
$\alpha_{1,1} = 0.6 + 0.2i$	$\beta_{1,1} = 0.5 + 0.6i$
$\alpha_{1,2} = 0.6 - 0.2i$	$\beta_{1,2} = 0.5 - 0.6i$
$\alpha_{1,3} = 0.7$	$\beta_{1,3} = 0.7 + 0.1i$
$\alpha_{1,4} = 0.6$	$\beta_{1,4} = 0.7 - 0.1i$
$\alpha_{1,5} = 0.3$	$\beta_{1,5} = 0.5$

System 2:

$$H_2 = \frac{z^5 - 2.55z^4 + 3.17z^3 - 2.549z^2 + 1.282z - 0.2958}{z^5 - 3.45z^4 + 4.537z^3 - 2.777z^2 + 0.754z - 0.06427}, \tag{4.65}$$



Poles	Zeros
$\alpha_{2,1} = 0.9 + 0.1i$	$\beta_{2,1} = 0.2 + 0.8i$
$\alpha_{2,2} = 0.9 - 0.1i$	$\beta_{2,2} = 0.2 - 0.8i$
$\alpha_{2,3} = 0.95$	$\beta_{2,3} = 0.7 + 0.3i$
$\alpha_{2,4} = 0.55$	$\beta_{2,4} = 0.7 - 0.3i$
$\alpha_{2,5} = 0.15$	$\beta_{2,5} = 0.75$

Grouping together $\pi = \{\alpha_{1,i}, \beta_{2,i}\}$ and $\zeta = \{\alpha_{2,i}, \beta_{1,i}\}$, we can now explicitly build Vandermonde-matrices, like the ones from Lemma 3, for π and ζ separately, which we shall denote by Γ_π and Γ_ζ, respectively. We compute 1000 rows of these matrices. The negative logarithm of the subspace angles kernel between them is

$$d_c^2(H_1, H_2) = -\log \prod_{k=1}^{10} \cos^2 \left[\Gamma_\pi \lhd \Gamma_\zeta\right] = 6.5352. \tag{4.66}$$

Alternatively, using Equation (2.59), and grouping variables as in Equations (2.57) and (2.58) we compute

$$d_c^2(H_1, H_2) = \log \frac{\prod_{i=1}^{10}\prod_{j=1}^{10} \left|1 - \pi_i \bar{\zeta}_j\right|^2}{\prod_{i,j=1}^{10} \left(1 - \pi_i \bar{\pi}_j\right) \prod_{i,j=1}^{10} \left(1 - \zeta_i \bar{\zeta}_j\right)} = 6.5352, \tag{4.67}$$

giving the exact same result as the subspace angles kernel in Equation (4.66).

Following the results from Subsection 4.3.2, we can now also explicitly calculate subspace angles from input and output data.

We take two random input systems, with 5 poles and zeros each, and simulate their response to white, Gaussian noise. This generates two deterministic input

signals, u_1 and u_2. The simulated output of the system with transfer function H_1, driven by u_1 is y_1, and, analogously, y_2 is the output of system H_2 when applying u_2 as input. All signals are of length $N = 2^{16}$. This set-up is the same as that in figure 4.3.

Next, we generate Hankel matrices with $i = 200$ rows and $j = 65337$ columns. Following the algorithm we find

$$d_c^2(H_1, H_2) = \tag{4.68}$$

$$-\log \prod_{k=1}^{10} \cos^2 \left[\left(Y_1^{i,j} \Pi_{U_1^{i,j}}^{\perp} \quad U_2^{i,j} \Pi_{Y_2^{i,j}}^{\perp} \right) \lhd \left(Y_2^{i,j} \Pi_{U_2^{i,j}}^{\perp} \quad U_1^{i,j} \Pi_{Y_1^{i,j}}^{\perp} \right) \right] = 6.5352,$$

giving again the exact same result.

Computing the cepstra of all the signals, we can compute the weighted cepstral distance directly and find

$$d_c^2 \left((y_1, u_1), (y_2, u_2) \right) = 6.5286, \tag{4.69}$$

in reasonable correspondence to the results in Equations (4.66), (4.67), and (4.68), but not exactly equal. Computing the weighted cepstral distance requires a choice for certain parameters, which can influence results. We investigate the statistical properties of errors induced by certain choices of these parameters in Section 4.6.

4.6 Computational aspects of the weighted power cepstral distance

The computation of the weighted power cepstral distance involves some algorithm and application dependent parameters. The subspace angles and model parameters give us independent ways to compute the weighted cepstral distance. We will use the ground truth obtained in this way to check the dependency of the computation of the weighted cepstral distance on the following parameters:

Length of PSD estimate In Equation (2.19), the quantity $\Phi_y \approx |\mathcal{F}(y)|^2$, was mentioned. Computationally, we use the well-known Welch's method [84], which accepts a parameter (`NFFT`) defining the length of the power spectral density (PSD) estimate. For the results above, a value of $\texttt{NFFT} = 2^{11}$ was chosen.

Sum cutoff The weighted cepstral distance involves an infinite sum. In practice, we are working with finite signals, so we have to cut off this sum somewhere, inducing a parameter (`cutoff`). The PSD estimate of length `NFFT` mentioned above includes both the dc-gain of the PSD, and the positive and negative parts of Φ, so a natural choice could be `cutoff` $=$ `NFFT`$/2$ (as the infinite sum runs over the positive coefficient numbers). However, note that the cepstral distance multiplies the difference between cepstral coefficients by their coefficient number k. For large k, this becomes numerically unstable. For the results above, we took `cutoff` $= 100$.

Order of the input model The order of the input dynamics is not precisely an algorithm parameter we get to choose, but rather an application-dependent property of the problem at hand. However, it is interesting to look at synthetic data with different values for this `inputOrder`, changing the importance of the model dynamics versus the input dynamics, and study the statistical properties of the results. In the experiment above, `inputOrder` $= 5$.

We now repeat the experiment from Section 4.5, for different values of these parameters, and repeat them 10000 times for different random inputs, to show how the results depend on these parameters, and what kind of errors can be expected.

Figure 4.5 shows resulting probability density functions for varying values of the variable `NFFT`, where the experiment was repeated 10000 times for each value of the parameter. When the parameter is chosen too small (in this case, $\leq 2^6$), a bias is introduced in the resulting value for the distance. The parameter `NFFT` decides how many points are present in the Fourier transform, and consequently what frequency resolution we can work with. The bias is a direct result of the power spectral density being too coarse-grained.

For slightly larger values ($\approx 2^7 - 2^{10}$), the bias is gone, and the distribution centers around the correct value of 6.5352. However, the resulting distributions still have a significant probability to overestimate the distance. While we would expect the distance to be underestimated (as we are cutting off the infinite sum of the distance after a finite amount of elements), a significant overestimation could indicate an improper choice of frequency resolution. Increasing `NFFT` further resolves this issue, with a distribution that has a very small probability of overestimating the distance. Note that, from about 2^{11} onwards, increasing the value even further does not change the distribution anymore. A good value in this case would then be `NFFT` $= 2^{11}$.

Figure 4.6 shows the results for different values of `cutoff`. When we cut off

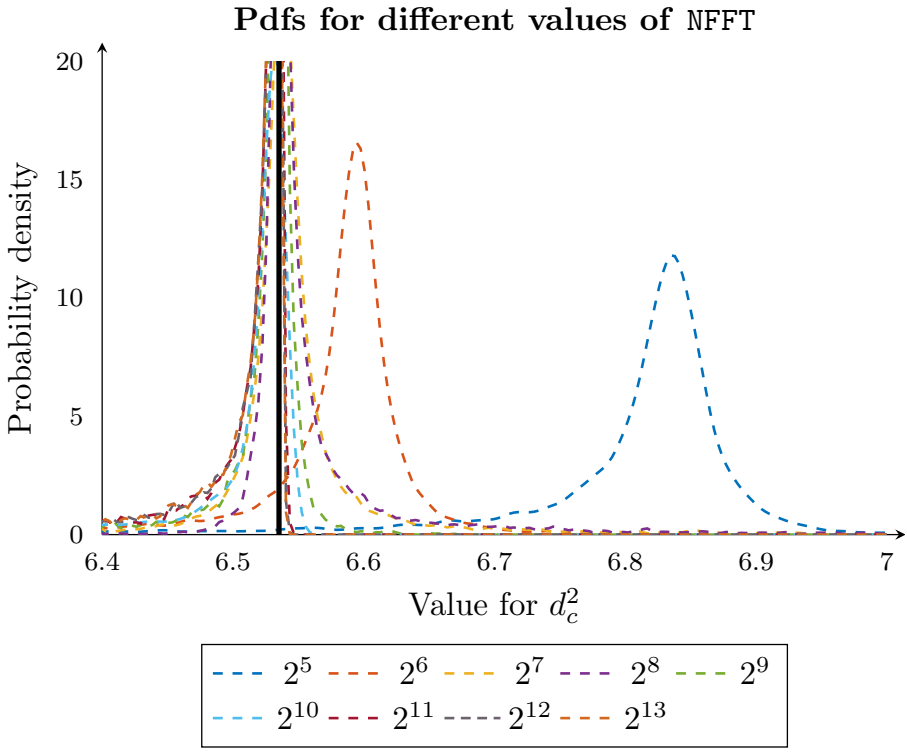


Figure 4.5: Probability density functions of the computed weighted cepstral distance for 10000 repetitions of the experiment in Subsection 4.5, for different values of `NFFT`. The other parameters are set at `cutoff` $= \min(\texttt{NFFT}/2-1, 100)$, `inputOrder` $= 1$. The black vertical line is at 6.5352, the theoretical value calculated via the subspace angles kernel. Too small values for `NFFT` results in a bias, and overestimation of the distance. Starting from about `NFFT` $= 2^{11}$, no further change in the distribution occurs, indicating an adequate choice of `NFFT`.

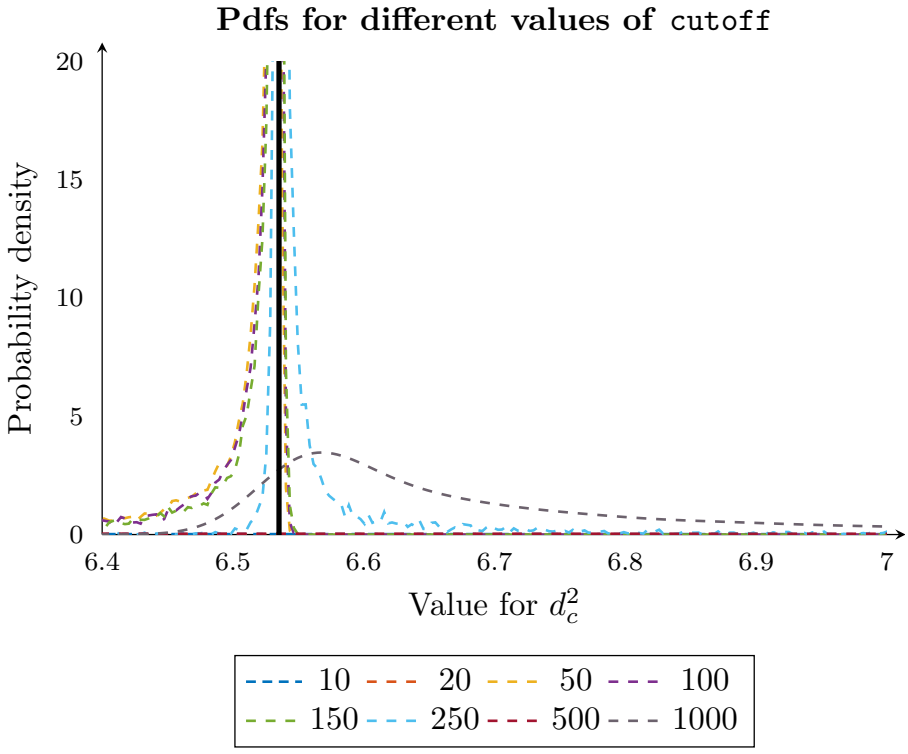


Figure 4.6: Probability density functions of the computed weighted cepstral distance for 10000 repetitions of the experiment in Subsection 4.5, for different values of `cutoff`. The other parameters are set as `NFFT` $= 2^{11}$, `inputOrder` $= 1$. The black vertical line is at 6.5352, the theoretical value calculated via the subspace angles kernel. We need `cutoff` large enough for the infinite sum in the distance calculation to converge, but small enough to avoid numerical errors to dominate. A good choice for this experiment is `cutoff` $= 100$.

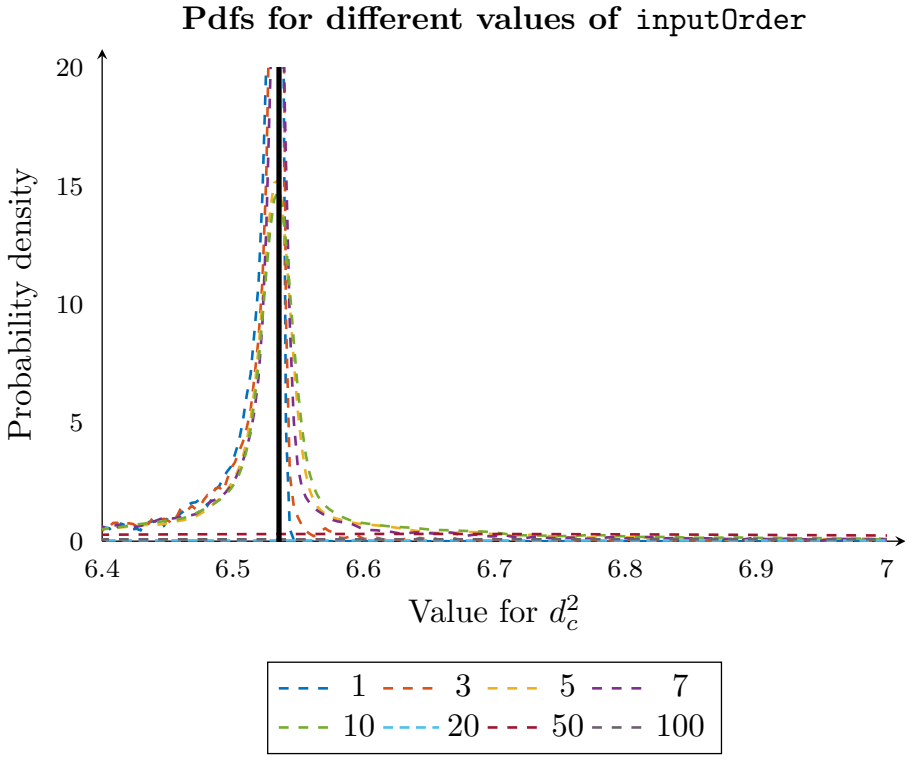


Figure 4.7: Probability density functions of the computed weighted cepstral distance for 10000 repetitions of the experiment in Subsection 4.5, for different values of `inputOrder`. The other parameters are set as $\texttt{NFFT} = 2^{11}$, $\texttt{cutoff} = \min(\texttt{NFFT}/2 - 1, 100)$. The black vertical line is at 6.5352, the theoretical value calculated via the subspace angles kernel. For larger and larger orders of the input process, the deterministic weighted cepstral distance becomes a worse and worse estimate of the subspace angles kernel.

too early (`cutoff` ≤ 20), the probability density function is very flat, and we do not get a meaningful estimate for the distance. Larger values ($\approx 50 - 150$) result in a nice distribution, almost never overestimating the distance (though they still exclude a lot of cepstral coefficients, and will underestimate the value a significant amount of time). Here, we can notice the problems with numerical stability for higher cepstrum coefficients: for larger values of `cutoff` (≥ 250), this noise starts to dominate the sum, resulting in an overestimation of the distance.

Figure 4.7 shows the results for different values of `inputOrder`. The interpretation is straightforward: as the dynamics of the input become relatively more important, the distribution becomes wider. When the system dynamics are completely dominated by those of the input, the computation of the weighted cepstral distance is no longer a meaningful estimate of the kernel. It seems strange that the order of the input has an impact on the results, as we subtract the input cepstrum to eliminate the input. However, small inaccuracies in computing the cepstra will drown out the contribution of the system itself, for high input orders.

4.7 Conclusion

We have shown the equivalences between the weighted cepstral distance and the model parameters of the underlying models on the one hand, and between the distance and the subspace angles kernel on the other.

Moreover, we have presented a novel algorithm to explicitly calculate the subspace angles between systems with different inputs, from their input and output data. This allowed us to compare the different approaches and show that the equivalences in this Chapter hold up numerically.

We then used the value obtained in this way as a ground truth, to investigate the dependency of the weighted cepstral distance on some algorithm and application dependent parameters.

The theoretical interpretation of the weigthed cepstral distance that existed in the stochastic case thus carries over to the case of deterministic systems. We will use this weighted cepstral distance measure to construct a novel clustering algorithm in the next Chapter. We will explicitly use the power cepstra to interpret the results of such a clustering algorithm, but it would be an interesting path of future research to do an analysis of the different equivalent ways to compute similarity presented in this Chapter, and explore their robustness.

Chapter 5

Weighted cepstral clustering

In this Chapter, we examine in detail the workflow of a cepstral clustering application with the weighted cepstral distance (summarized in Figure 5.1).

We have introduced, in Chapter 2, input and output signals, the power cepstrum and the weighted power cepstral distance (with some remarks on the weighting of the zero-th cepstrum coefficient). Chapters 3 and 4 gave us an extension of this distance measure to the deterministic case.

Here, we fill in the missing parts to be able to perform cepstral clustering:

- cepstral k-means in Section 5.1, an extension of the traditional k-means algorithm to make it compatible to the weighted cepstral distance,
- cluster interpretation in Subsection 5.1.3, where we interpret clustering results in terms of unknown dynamical systems generating the time series under consideration,
- windowing in Section 5.2, which is an application-dependent preprocessing step in many time series analysis problems in general.

While discussing the windowing step, we will introduce an open problem in machine learning, namely that of time series subsequence clustering. We will show that the cepstral k-means algorithm introduced in this Chapter provides a solution to this important challenge.

We will show the practical usefulness of the techniques introduced here in a real-life application on wind turbine anomaly detection, in Section 5.3.

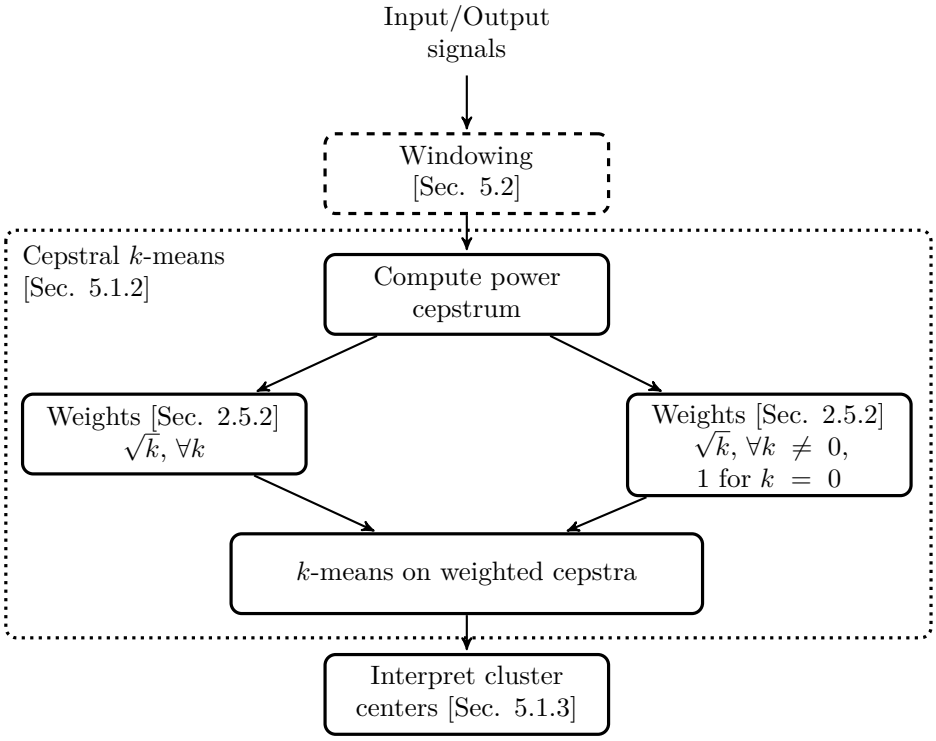


Figure 5.1: General summary of a cepstral clustering approach. We discussed some notation on input and output signals in Section 2.1. Computing the power cepstrum was discussed in Section 2.4, while the choice of weighting for the zero-th cepstrum coefficient was discussed in Subsection 2.5.2. In this Chapter, we will discuss the k-means algorithm in Subsection 5.1.1. The different nodes inside the dotted box constitute the cepstral k-means algorithm, discussed in Subsection 5.1.2. Subsection 5.1.3 will give an interpretation to the clustering results, while Section 5.2 will explain windowing of time series, and some issues related to that.

5.1 A cepstral clustering algorithm

In Chapter 3, we presented a first clustering application using the weighted power cepstral distance, concerning electrical circuits (see Subsection 3.4). There, we employed an off-the-shelf clustering technique, known as hierarchical clustering on distance matrices generated by different distance measures. In this way, we showed the relative merit of the weighted cepstral distance versus that of traditional shape-based techniques, on the specific application of electrical circuits. Some results on cepstral clustering (or the subspace angles kernel above), in some form or another, can be found, in, for example [1, 7, 81, 88].

In this Section, we will introduce a novel cepstral clustering algorithm. It is an extension of the traditional, well-known k-means clustering, a summary of which is provided in Subsection 5.1.1. In the new clustering approach we develop, we work on the weighted cepstra of the time series, instead of on the raw signal data. We will present the extended algorithm, which we dub *cepstral k-means*, in Subsection 5.1.2.

The benefit of this approach is that it allows us to interpret the clustering results in terms of the dynamical systems underlying the problem, as we will show for cluster centers and variance in Subsection 5.1.3.

5.1.1 k-means clustering

The traditional k-means algorithm is a well-known and often-used clustering algorithm, included in most data analysis programming languages, e.g. in Python's *sklearn*-library or Matlab's *Statistics and Machine Learning Toolbox*. It was first developed by Stuart Lloyd in 1957, though he did not publish the algorithm in a journal paper until 1982 [50]. The term k-means was first coined by MacQueen [51] in 1967.

Though faster alternatives exist now (see, for example [21]), the original algorithm, sometimes referred to as *naive k-means*, is easy to understand, and summarized in Algorithm 4.

Starting from a set of signals, a value for the amount of clusters the algorithm must return, and an initial estimate for the cluster centers, the algorithm iterates between two steps:

Cluster assignment: the algorithm assigns each signal in the set to a cluster,

Update cluster centers: the algorithm calculates the centers of the new clusters generated in the previous step.

Algorithm 4 Traditional k-means

input : Y, the set of signals, y, we want to partition in clusters
$n_{\mathcal{C}}$, the amount of clusters we want to partition Y in
$\tilde{\mathcal{Y}} = \{\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_{n_{\mathcal{C}}}\}$, initial estimates for the cluster centers
output : $\mathcal{C} = \{C_1, C_2, \ldots, C_{n_{\mathcal{C}}}\}$, a partitioning of Y
$\tilde{\mathcal{Y}} = \{\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_{n_{\mathcal{C}}}\}$, the cluster centers
for $j \leftarrow 1$ **to** *maximum number of iterations* **or** *partitioning does not change anymore* **do**

$$\forall i,\ C_i \leftarrow \left\{y : ||y - \tilde{y}_i||^2 < ||y - \tilde{y}_l||^2, \quad \forall l \neq i, \quad l \in \{1, 2, \ldots, n_{\mathcal{C}}\}\right\}$$

$$\forall i,\ y_i \leftarrow \frac{1}{N_{C_i}} \sum_{\{m : y_m \in C_i\}} y_m$$

// N_{C_i} is the number of signals in partition C_i

end
return $\mathcal{C}$,$\tilde{\mathcal{Y}}$

The algorithm then iterates until it converges, or a maximum amount of iterations is reached.

Note that, from the mathematical expression of these two steps in Algorithm 4, it is clear that the cluster assignment is done using the Euclidean distance, and the cluster center is defined as the average over the different signals. The k-means algorithm tacitly assumes similarity is to be measured by the Euclidean distance, and it is not suited for arbitrary[1] distance measures.

The k-means clustering algorithm then aims to find a partitioning, $\mathcal{C} = \{C_1, C_2, \ldots, C_{n_{\mathcal{C}}}\}$, which minimizes the objective function

$$\arg\min_{\mathcal{C}} \sum_{i=1}^{n_{\mathcal{C}}} N_{C_i} \sigma^2 (C_i), \tag{5.1}$$

with $\sigma^2(\cdot)$ the variance of the cluster. For a cluster C_i, containing signals y_j, and cluster average $\tilde{y}_i$ we have

$$\sigma^2(C_i) = \sum_{y_m \in C_i} \frac{1}{N_{C_i} ||y_m - \tilde{y}||^2}. \tag{5.2}$$

While the algorithm tries to minimize Equation (5.1), it is not guaranteed this optimum is reached.

[1] A popular alternative that allows a user-defined distance measure, is k-medoids, where the cluster centers are no longer the average of all objects in a cluster, but the object of the cluster that minimizes a distance metric to all others. This algorithm was first proposed in [35]. This means that for the k-medoids algorithm, the center of the cluster will always be an object of the original dataset.

5.1.2 Cepstral k-means clustering

We propose a novel clustering algorithm, which we will call cepstral k-means. This clustering algorithm is an extension of the k-means algorithm described above, by rewriting the weighted cepstral distance between signals as the Euclidean distance between power cepstra weighted by the square root of their coefficient number.

Given two pairs of input-output signals, (y_1, u_1) and (y_2, u_2), we can calculate two model power cepstra $c_{h_1}(k) = c_{y_1}(k) - c_{u_1}(k)$, $\forall k$, and analogous for c_{h_2}. Weighting the cepstral coefficients with the square root of their coefficient number, $\sqrt{k}$, we can define

$$s_{h_1} = \sqrt{k} c_{h_1}(k), \tag{5.3}$$

and analogous for s_{h_2}.

We can now rewrite the weighted cepstral distance in Equation (2.44) as

$$\begin{aligned} d_c^2\left((y_1, u_1), (y_2, u_2)\right) &= \sum_{k=1}^{\infty} k \left(c_{h_2}(k) - c_{h_1}(k)\right)^2 \\ &= \sum_{k=1}^{\infty} \left(\sqrt{k} c_{h_2}(k) - \sqrt{k} c_{h_1}(k)\right)^2 \\ &= d_e^2\left(s_{h_1}, s_{h_2}\right), \end{aligned} \tag{5.4}$$

with d_e the Euclidean distance. The k-means clustering algorithm employs the Euclidean distance, which means we can run k-means on the square root weighted power cepstra $s_{h_i} = \sqrt{k} c_i(k)$.

In practice, we are working with finite signals (of length n), and the infinite sum will be approximated as

$$d_c^2\left((y_1, u_1), (y_2, u_2)\right) \approx \sum_{k=0}^{n} \left(\sqrt{k} c_{h_2}(k) - \sqrt{k} c_{h_1}(k)\right)^2. \tag{5.5}$$

The cepstral k-means algorithm consists of running the k-means algorithm in Algorithm 4 on the weighted model cepstra $\sqrt{k} c_{h_i}(k)$ belonging to the signals in the data set to be clustered.

5.1.3 Interpretation of clustering results

We now investigate the centers or averages, and the variance in the resulting clusters, and give them an interpretation in terms of generating dynamics of the clustered signals.

Cluster average

To interpret the cluster centers, we first repeat a result from [63] on the geometric average of power spectral densities.

Given N estimates of a power spectral density, $\hat{\Phi}_{h,j}$, with $j \in \{1, 2, \ldots, N\}$, we take their geometric average to get an estimator

$$\tilde{\Phi}_h = \sqrt[N]{\prod_{j=1}^{N} \hat{\Phi}_{h,j}}. \tag{5.6}$$

This estimator has been shown [4, 63] to be less biased than the arithmetic average over the different $\hat{\Phi}_{h,j}$, and as such is a less-biased estimator of the average dynamics in the set of power spectral densities.

These results hold for certain assumptions on the frequency domain noise of the estimates. The authors of [63] claim, for Fast Fourier Transforms (FFT) of finite signals, it is justified in practice to treat the noise as being uncorrelated and normally distributed. For a more in-depth treatise of noise on Fourier coefficients in the computational domain, we refer to [70].

Computationally, we sample each power spectral density at a discrete set of frequencies, ω_l^d, $l \in \{1, \ldots, n_\Phi\}$, with n the length of the sampled version of the power spectral density. Their geometric average is then computed as

$$\tilde{\Phi}_h\left(\mathrm{e}^{i\omega_l^d}\right) = \sqrt[N]{\prod_{j=1}^{N} \hat{\Phi}_{h,j}\left(\mathrm{e}^{i\omega_l^d}\right)}. \tag{5.7}$$

This result has an implication for the average model power cepstrum. Starting from N signals, y_j, all coming from the same system with true power spectral density Φ_h, we compute their power cepstra, $\hat{c}_{h,j}(k)$, with $j \in \{1, 2, \ldots, N\}$. The average of these cepstra is then

$$\tilde{c}_h(k) = \frac{1}{N} \sum_{j=1}^{N} \hat{c}_{h,j}(k), \tag{5.8}$$

where each cepstrum has the same number of coefficients.

Taking the Fourier transform of both sides, we find (following the definition of the model cepstrum in Equation (2.22))

$$\begin{aligned}\mathcal{F}\{\tilde{c}_h\} &= \frac{1}{N}\sum_{j=1}^{N}\log\left(\hat{\Phi}_{h,j}\right) = \frac{1}{N}\log\left(\prod_{j=1}^{N}\hat{\Phi}_{h,j}\right) \\ &= \log\left(\sqrt[N]{\prod_{j=1}^{N}\hat{\Phi}_{h,j}}\right) = \log\left(\tilde{\Phi}_h\right),\end{aligned} \tag{5.9}$$

where the last line follows from Equation (5.6).

The average of the power cepstra in Equation (5.8) thus represents the dynamics of the geometric mean of the different estimates of the power spectral density. The average $\tilde{c}_h$ now can be seen to capture the dynamics of $\tilde{\Phi}_h$, the same low-biased geometric average estimator, with the low-biased properties of the result in [63].

Assume a cluster C, one of the clusters resulting from the cepstral k-means algorithm, with the signals y_j, $j \in \{1, \ldots, N_C\}$ assigned to it. Compute for each y_j its power cepstrum $\hat{c}_j$. The cluster average is then

$$\sqrt{k}\tilde{c}_h(k) = \frac{\sqrt{k}}{N_C}\sum_{j=1}^{N_C}\hat{c}_{y_j}(k), \tag{5.10}$$

which can be interpreted following the discussion above as the $\sqrt{k}$-weighted cepstral coefficients representing the geometric average over the power spectral densities of the signals present in the cluster.

These results show that cepstral k-means clustering produces reasonable results. We indeed want a good cepstral clustering algorithm to base its partitioning on the dynamics underlying the time series. The fact that the cluster averages can be interpreted in terms of the average dynamics of the signals is a good start. In Chapter 6, we will come back to this result, and propose an extremely efficient way to estimate a model for these cluster centers.

For now, let us first look at an interpretation for the within-cluster variance of cepstral k-means.

Cluster variance

We set out to obtain clusters that were meaningful and interpretable. Interpreting the cluster centers as low-bias estimators in Equation (5.10) was a good start to achieve this. Indeed, it shows that the clusters are centered around different dynamical systems. If we have signals coming from a machine with several operational modes, these clusters could thus be centered around these modes.

The k-means algorithm tries to minimize the variance within a cluster. We can interpret this variance for the cepstral k-means algorithm in terms of dynamical systems.

Let us assume a cluster C, consisting[2] of N_C weighted power cepstra $\sqrt{k}c_i(k)$, of length n, as we would get from a cepstral k-means clustering procedure. Each weighted power cepstrum is derived from a signal, y_i.

We assume these signals come from the same system $\tilde{H}$, which represents the true dynamics of the problem. We can associate with $\tilde{H}$ a power cepstrum $c_{\tilde{h}}$, which is approximated in a low-biased way by the average cepstrum of the cluster C, denoted by $\tilde{c}_h$, as in Equation (5.10). We thus have $c_{\tilde{h}}(k) \approx \tilde{c}_h(k)$.

We assume each signal y_i to be described by a power cepstrum

$$c_{y_i} = c_{\tilde{h}} + c_{h_i}, \tag{5.11}$$

which translates to z-domain as

$$Y_i(z) = \tilde{H}(z)H_i(z), \tag{5.12}$$

where $H_i(z)$ is a transfer function between the true dynamics of the problem and its concrete realization in the signal y_i. It represents the model we would have to cascade the true model with to get a signal y_i. A block diagram of these models is given in Figure 5.2.

The variance within the cluster is defined as

$$\sigma^2(C) = \sum_{k=0}^{n} \frac{1}{N_C} \sum_{i=1}^{N_C} \left(\sqrt{k}c_{y_i}(k) - \sqrt{k}\tilde{c}(k) \right)^2 \tag{5.13}$$

$$= \sum_{k=0}^{n} \frac{k}{N_C} \sum_{i=1}^{N_C} \left(c_{y_i}(k) - \tilde{c}(k) \right)^2. \tag{5.14}$$

[2]Of course, one could argue the cluster consists of the signals y_i of which we computed the power cepstra. Since there is a one to one correspondence between signals and cepstra in this case, we will refer to the cluster as having signals or power cepstra, according to what is most convenient in the specific situation.

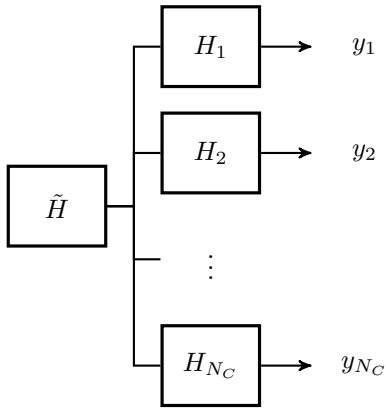


Figure 5.2: A block diagram of the systems giving rise to the signals associated with the power cepstra in Equation (5.11).

From Equation (5.11), we find that $c_{y_i}(k) - \tilde{c}(k) = c_{h_i}(k)$, the power cepstrum of the error transfer function, H_i.

We then have that

$$\sigma^2(C) = \frac{1}{N_C} \sum_{i=1}^{N_C} \sum_{k=0}^{n} k c_{h_i}(k)^2 \tag{5.15}$$

$$= \frac{1}{N_C} \sum_{i=1}^{N_C} ||H_i||_c^2, \tag{5.16}$$

the average of the weighted cepstral norms of the error transfer functions.

The k-means algorithm from Subsection 5.1.1 tries to find a partitioning, $\mathcal{C} = \{C_1, C_2, \ldots, C_p\}$, which minimizes the objective function (as in Equation (5.1))

$$\arg\min_{\mathcal{C}} \sum_{j=1}^{p} N_{C_j} \sigma^2 (C_j) \,. \tag{5.17}$$

In the case of cepstral k-means, we then minimize the objective function

$$\arg\min_{\mathcal{C}} \sum_{j=1}^{p} N_{C_j} \frac{1}{N_{C_j}} \sum_{i=1}^{N_{C_j}} ||H_i||_c^2 \tag{5.18}$$

$$= \arg\min_{\mathcal{C}} \sum_{i=1}^{N} ||H_i||_c^2, \tag{5.19}$$

with $N = \sum_j^p N_{C_j}$.

The cepstral k-means algorithm is thus trying to minimize the sum over the squares of the weighted cepstral norms of the individual error transfer functions it would take to go from the dynamics of the cluster center to the clustered signals.

In light of the previous two sections, we can now expect that, starting from different systems producing signals, the cepstral k-means algorithm will return clusters centered around the true dynamics of each system, and with the output signals of that system as the objects in the cluster.

5.2 Windowing

Windowing time series is a preprocessing step in many time series applications. A windowing approach is defined by two parameters, the window length, l_w, and the step, s. A time series, $y(k)$, $k \in \{1, 2, \ldots, N\}$ is cut up in windows w_i as follows (assume $N - l_w + 1$ an integer multiple of s):

$$\begin{aligned} w_1 &= y(1 : l_w), \\ w_2 &= y(s : s + l_w - 1), \\ &\vdots \\ w_m &= y(N - l_w + 1 : N), \end{aligned} \tag{5.20}$$

where the notation $y(a : b)$ denotes a slice of the signal $y(k)$ containing all timepoints between a and b, including a and b.

These windows are very useful entities in several scenarios, for example:

Anomaly detection: the objective here would be to find which windows w_i belong to normal behaviour of the application at hand, and which belong

to faulty behaviour. An example could be fault detection in wind turbines, which we will discuss in Subsection 5.3.

Operation mode identification: trying to identify different operation modes in an application, and grouping together windows belonging to the same mode of operation. An example of this will follow later on in this thesis, in Chapter 8, on partitioning historical data on international energy markets in groups based on the dynamics of the market.

In the literature, the windows defined above are also known as subsequences, and the problem of clustering them is known as time series subsequence clustering. While this might seem a straightforward method of time series analysis, it exhibits some fundamental issues. Traditional techniques are inadequate to solve this problem, as we discuss in the next Subsection.

5.2.1 Time series subsequence clustering

While there used to be an active body of research employing this technique, a surprising result showed that, in general, this problem is fundamentally flawed. Indeed, in [37], the authors show through examples and some theorems that time series subsequence clustering is meaningless.

It appears that, no matter what dataset we use to execute the procedure on, the results will always be the same: clusters are centered around sine waves, with phases that are maximally far apart in frequency domain. In other words, the results of time series subsequence clustering using traditional techniques are independent of the input!

These results were of course quite shocking, and a good overview of their implications for the field is given in [91], dividing the history of subsequence time series clustering in the preproof, interproof and postproof periods (the proof being the results in [37]).

A classic example will elicit what we mean.

Example 10 (Cylinder-Bell-Funnel data)**.** The Cylinder-Bell-Funnel (CBF) data, which is the canonical example used in the (interproof and postproof) time series subsequence clustering literature, consists of three types of signals, the shapes of which loosely resemble a cylinder, a bell or a funnel. This synthetic dataset was first introduced in [69].

The three classes of signals are generated as

cylinder

$$c(k) = (6 + \eta)\chi_{[a,b]}(k) + \epsilon(k), \tag{5.21}$$

bell

$$b(k) = (6 + \eta)\frac{(k - a)}{(b - a)}\chi_{[a,b]}(k) + \epsilon(k), \tag{5.22}$$

funnel

$$f(k) = (6 + \eta)\frac{(b - k)}{(b - k)}\chi_{[a,b]}(k) + \epsilon(k), \tag{5.23}$$

with $k = 1, \ldots, 128$, a is an integer, uniformly random on the interval $[16, 32]$, $b - a$ is an integer from the uniform distribution on $[32, 96]$, η and ϵ are standard normal, Gaussian noise, and $\chi_{[a,b]}(k)$ is the characteristic function, which equals 1 on the interval $[a, b]$, and 0 outside it.

Realizations of the different signals are shown in Figure 5.3.

We now generate 12 instances of these signals (4 instances per class), and concatenate them to each other in a random order, as in Figure 5.4. We then take sliding windows of this data, with window length $l_w = 128$, and sliding step, $s = 1$.

We establish a ground truth, by assigning each window to the class that is most prevalent within it (i.e., the first 64 windows belong to the class of the first signal in the concatenated series, the next 128 to that of the second, ...). The results of a k-means clustering algorithm (which uses the Euclidean distance) on these windows, alongside the ground truth, is shown in Figure 5.5. These results look quite random, and do not meaningfully represent the clusters from the ground truth.

Figure 5.6 shows the centers of the different clusters. Note that these are indeed converging to sine waves, maximally differing in phase. This indicates that the phase component of the dynamics behind the time series are given too much importance in clustering the subsequences.

In the original paper by [37], experiments seem to indicate these surprising results hold, regardless of the distance measure used. However, the fundamental reasons why this method fails, remained unclear, and in the interproof period, many attempts have been made to gain more insight in this regard.

An interesting contribution is that of [25], who do a frequency analysis of time series subsequence clustering. They prove that the power spectra of cluster centers obtained with traditional techniques will concentrate on a small number of frequencies, and therefore closely resemble sine waves. This is a consequence

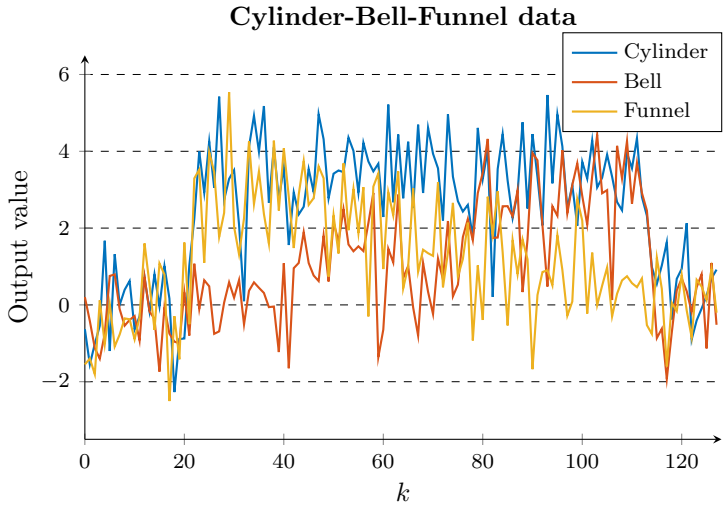


Figure 5.3: Examples of the cylinder (Equation (5.21)), bell (Equation (5.22)) and funnel (Equation (5.23)) signals, first introduced in [69].

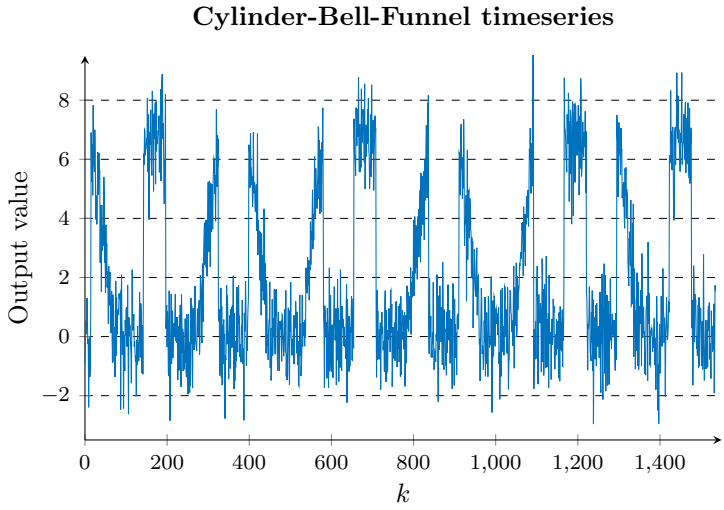


Figure 5.4: A concatenated series of 12 random instances of CBF-signals. We do subsequence clustering on this dataset.

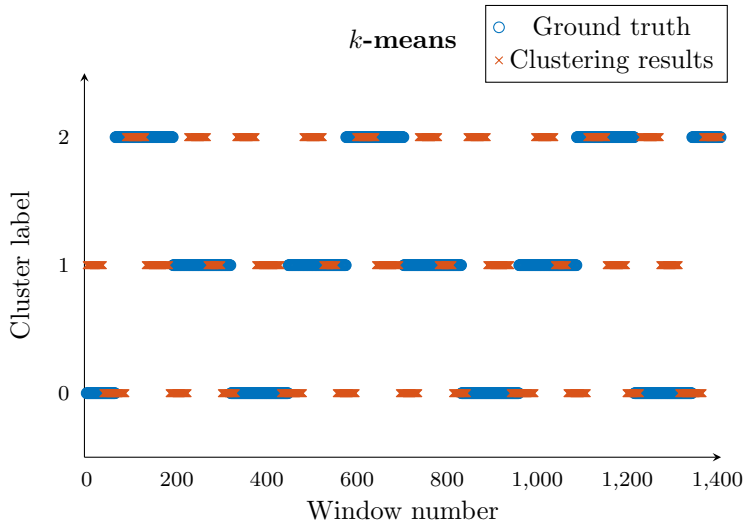


Figure 5.5: Traditional k-means clustering on windows from the time series in Figure 5.4, alongside the ground truth. We see the results from the clustering algorithm are quite random, and certainly do not coincide with the ground truth at all.

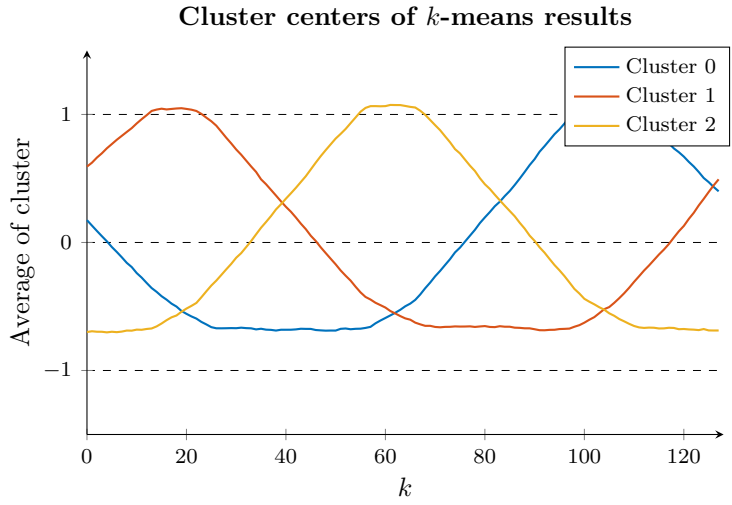


Figure 5.6: Cluster centers for the k-means algorithm, as in Figure 5.5. Note that the cluster centers converge to sine waves, maximally differing in phase.

of the fact that the frequency responses belonging to consecutive windows will have very similar magnitudes, but different phases [15].

The weighted cepstral distance developed here could potentially solve this problem, as it hinges on a transformation of the *power* spectra of the individual windows or subsequences, and therefore does not take the phase information into account. Returning to the CBF dataset, results indeed seem quite promising!

Example 11. (Cylinder-Bell-Funnel data, continued)

As in the previous example, we create a dataset with instances of the CBF signals. Comparing Equations (5.22) and (5.23), for the bell and funnel signals, respectively, it is easily seen that the funnel and bell represent the same dynamics with the role of a and b inverted. This is also immediately clear from the visual representation in Figure 5.3. The weighted cepstral distance does not discern between these signals, as they are the same from a dynamical point of view.

Therefore, in this example, we will work with a dataset consisting of only cylinder and bell data.[3] We generate 6 instances of both classes, concatenate and generate windows, exactly as above. Now, however, we calculate for each window w_i the weighted power cepstrum

$$w_i^c = \sqrt{k} c_i(k), \tag{5.24}$$

with c_i the power cepstrum of window w_i, and run k-means on the w_i^c. This is the cepstral k-means clustering algorithm from Section 5.1.

Results are shown in Figure 5.7. We see that the cepstral k-means almost perfectly extracts the two clusters, except at the boundaries of the clusters, where the problem is ill-defined in any case.

We show the average over the signals belonging to the different clusters in Figure 5.8. As can be observed, these are no longer the maximally spaced sine waves from the previous example, that were shown in [37] to be fingerprints of a meaningless, input-independent subsequence clustering algorithm. As we are now no longer running k-means on the data windows themselves, but rather on the weighted cepstral coefficients of these windows, we plot the cluster centers (which represent the averages over all the coefficient series belonging to the cluster, as in Equation (5.10)) in Figure 5.9. Here, again, we do not get maximally spaced sine waves. Rather, we have fundamentally solved the problem related to the different phases of the data windows dominating the clustering results.

[3]Running traditional k-means on this new set again results in two sine waves, maximally apart in phase, and results are similar to the example with the three classes present. Because of this similarity, results for traditional k-means for the two-class set are omitted.

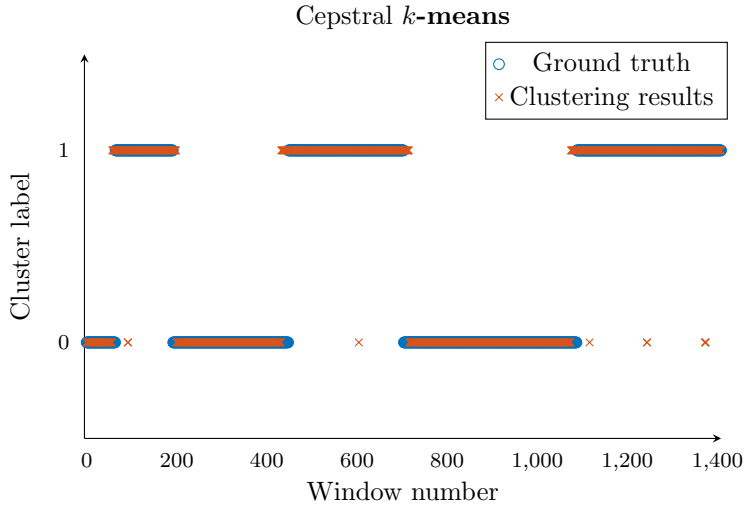


Figure 5.7: Cepstral k-means clustering results for a time series consisting of concatenated cylinder and bell signals. This algorithm almost perfectly returns the correct clusters.

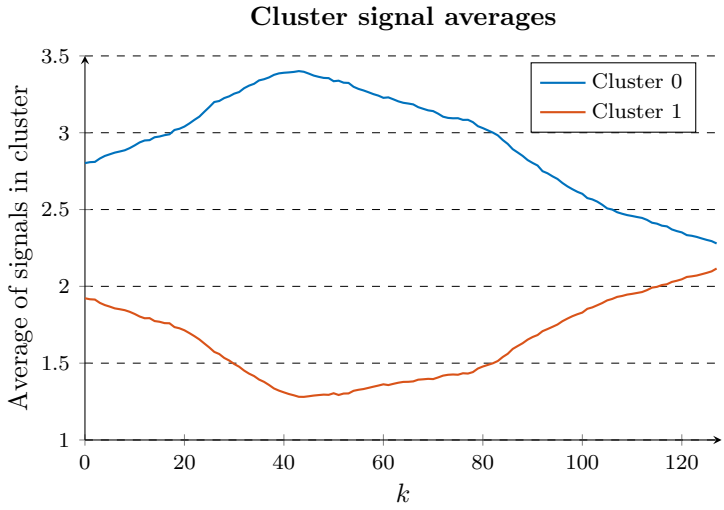


Figure 5.8: Average of the signals present in the two clusters from Figure 5.7. In contrast with the traditional k-means method, these averages are no longer maximally spaced sine waves.

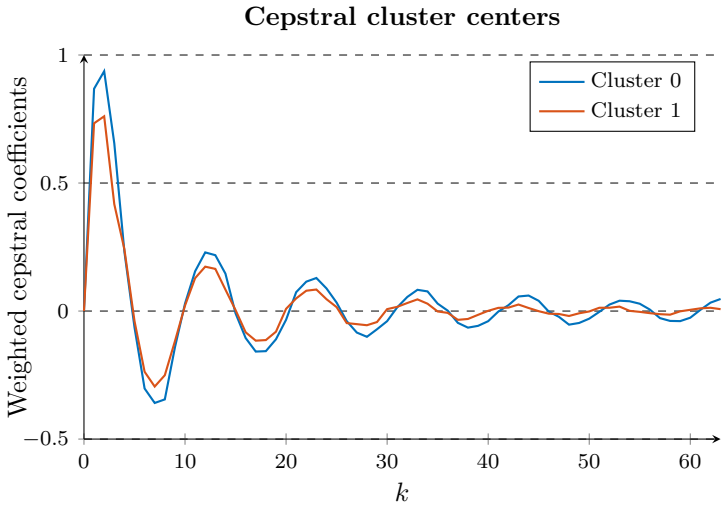


Figure 5.9: Cluster centers of the cepstral k-means clustering in Figure 5.7. These are the averages over all w_i^c from Equation (5.24).

While the weighted cepstral distance is not perfectly suited for the CBF dataset, as it is not able to discern between the bell and funnel signals, it is clear that it is not troubled by the usual problems with time series subsequence clustering.

The weighted cepstral distance thus allows us to get a solution to a problem unachievable by traditional techniques. It is yet another example of how the choice of distance measure is of paramount importance in time series analysis. Moreover, it shows how the dynamical background of the weighted cepstral distance, and the theoretical framework that comes along with it, can help us understand – and therefore overcome – limitations and applicability of unsupervised methods.

On a side note: perhaps a proper extension of the semi-norm to a norm, as discussed in Subsection 2.5.2, could help alleviate the problems we had with discerning between bell and funnel data in the examples in the previous section. The fact that we are working with a semi-norm can help us understand that we were not able to discern between bell and funnel windows. However, we did not manage to find a different weighting scheme that solves this particular problem.

5.3 Unsupervised anomaly detection in wind turbines

Let us now try out the cepstral k-means algorithm on a real-life problem, with open source data coming from a test set-up.

An interesting challenge in wind turbine operation is anomaly detection. Unscheduled maintenance due to unexpected failures are particularly expensive, and automatic detection of faults improves reliability and reduces maintenance costs.

In this Section, we investigate cepstral k-means as a method to discover faulty dynamics in a completely unsupervised way. Normal and faulty operating behaviours will have different underlying dynamics. The interpretation of cluster centers and variances in terms of underlying models, from Subsection 5.1.3, would suggest it should be able to discern between the two.

In [65], on gear and bearing fault diagnostics, cepstral techniques for harmonics filtering and fault diagnostics in rotating devices are discussed, by looking at what we would call the model complex ceptrum. Similar techniques were used in [52], to edit signals coming from wind turbines to identify modes. These results encourage us that the use of the weighted cepstral distance could generate impressive results in these types of applications.

5.3.1 Data

The data under consideration is the training dataset from EDF's *Hack the Wind* challenge [20], consisting of data coming from a *Supervisory Control And Data Acquisition* (SCADA) system installed on several wind turbines. This SCADA data is already being collected by the wind turbine, so no extra system installation is required for the fault detection step. The data contains information on five different components, namely the gearbox, the generator, generator bearings, the transformer, and the hydraulic group. In this application, we focus on generator faults, as it is one of the more costly components to inspect, repair and replace. The data set consists of multiple turbines, but for the purposes of this application, we focus on turbine 6, as it contains the most samples of generator faults.

For this generator, we have averaged (over ten minutes) temperature data available for each of the three phases in the stator winding within the generator T_1, T_2, and T_3. We combine these signals into one by taking the root mean

square:

$$T_{tot} = \sqrt{\frac{T_1^2 + T_2^2 + T_3^2}{3}}. \tag{5.25}$$

The root mean square was taken, rather than e.g. the maximum, as the (im)balance between the temperatures of the different phases can be important. We consider this as the output signal of the system, and take the average wind speed, v_w, as an input. Both T_{tot} and v_w are measured every ten minutes, starting on 1st of January 2016, up to the 31st of October 2017. We generate sliding windows of this data, consisting of a week of data each, with a sliding step of one day. This results in 1008 data points per window (7 days of 24 hours, with 6 measurements per hour).

Some timestamped error logs are available, containing information about faults that occured, and what action was taken to mitigate. We should thus expect the faulty behavior to precede the entry in the error log. The aim will be to find all errors pertaining to the generator, without incorporating any of the error logs in the algorithm.

To run our cepstral k-means algorithm, we compute the power cepstra of the average wind speed, $c_{v_w,i}$, and that of the total temperature, $c_{T_{tot},i}$, for each window, i. We can then calculate the model cepstrum, c_i, of window i, as

$$c_i(k) = c_{T_{tot},i}(k) - c_{v_w,i}(k), \tag{5.26}$$

which we can then weight by the square root of k, and get $\sqrt{k}c_i(k)$. For each window, we compute 64 weighted cepstrum coefficients, which will be our cutoff. We then run k-means on these weighted cepstral windows, as explained in 5.1.2.

5.3.2 Clustering results

Running the cepstral k-means algorithm, instructing it to find 2 clusters, we get the results shown in Figure 5.10. It is immediately clear we get one cluster with healthy behavior, and one with faulty behavior, containing all the generator faults. Indeed, every entry of the 0-th cluster precedes an error log (where the first two error logs are close together in time, and follow one period of anomalous behavior).

These results depend crucially on the inclusion of the average wind speed as an input. When we do not take this into account, and work solely on the total temperature, we get the clustering results from Figure 5.11, where only the first anomaly is found. This is an instance where the use of the model cepstrum instead of the output cepstrum shows its merit.

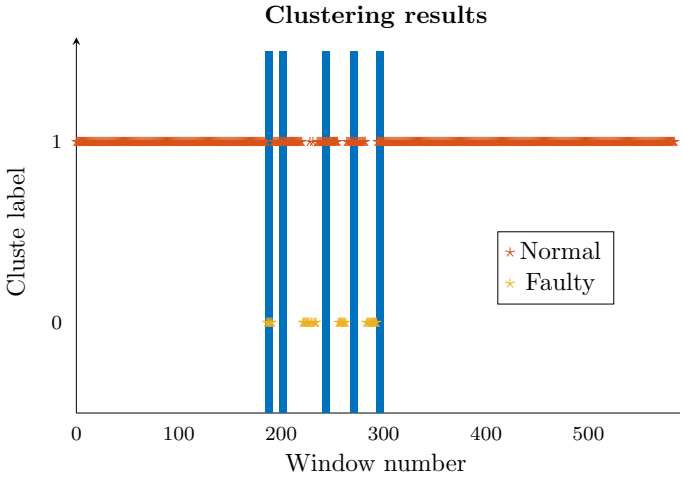


Figure 5.10: Cepstral k-means clustering results. For each window number, the assigned cluster label is marked. The blue rectangles span all windows containing the error log. There are seven such windows per error. Each error log is preceded by some windows corresponding to cluster 0, with the two first errors close together in time, and following one period of anomalous behavior.

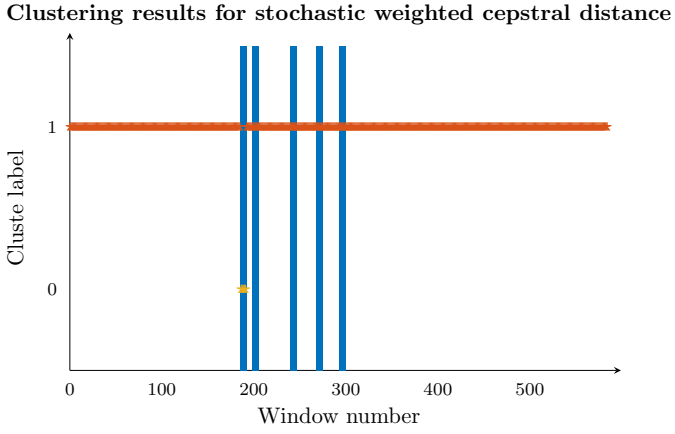


Figure 5.11: The same experiment as in Figure 5.10, but now working on only the temperature data, doing k-means clustering on $\sqrt{k}c_{T_{tot},i}(k)$. The clustering algorithm now misses most of the anomalous behavior.

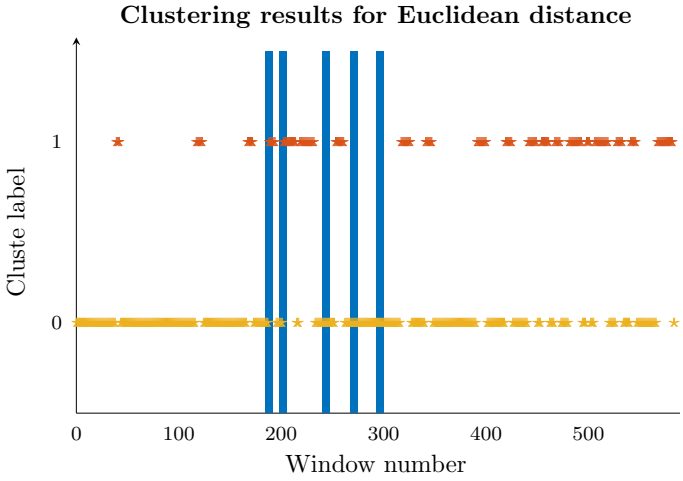


Figure 5.12: The same experiment as in Figure 5.10, but now working with the Euclidean distance on the total temperature data. The clustering algorithm does not separate the normal and faulty behavior.

To see the added value of cepstral k-means, we show the results of standard k-means clustering (using the Euclidean distance on the total temperature data), in Figure 5.12. It is clear we did not retrieve the faulty behavior in this way.

Note that this was done completely unsupervised, and the algorithm had no prior knowledge about what the faults look like! The model power cepstral distance shows to be a natural similarity measure to look at these types of problems, and cepstral k-means provides a way to usefully apply it in a clustering problem.

The set-up as presented here is just to show the power of the weighted cepstral distance in this type of anomaly detection. Of course, in practice, it might be more sensible to use the training labels as extra information for a classification algorithm (that could itself be based on the model cepstra). The fact that we can automatically retrieve the faults from the data without these labels, however, is noteworthy in itself, which is why we preferred to show results for the unsupervised approach in this Chapter.

Chapter 6

Cepstral identification of autoregressive systems

In this chapter, we develop a system identification method that estimates the difference equation or pole polynomial $a(z)$ of a model, starting from its model power cepstrum $c_h(k)$ (which itself can be derived from input and output cepstra, $c_u(k)$ and $c_y(k)$).

This system identification algorithm will

- only need the first n cepstral coefficients to obtain an n-th order model,
- consist of exact expressions for $a(z)$ in function of the $c_h(k)$, and as such operates in constant time with regard to the length of the signals/cepstra; the computational complexity will thus be $\mathcal{O}(n)$, with n the order of the model,
- return stable all-pole models for any model power cepstrum, but can be generalized to systems of mixed stability.

This estimation technique is instrumental in interpreting cluster centers and variance, resulting from a cepstral k-means approach.

6.1 Estimating dynamics present in a cluster

In Section 5.1, we interpreted resulting cluster centers and cluster variance of the cepstral k-means algorithm. The cluster centers are the averages over the weighted power cepstra of all the signals in a cluster, and were shown to be related to the geometric average over the power spectral densities of these signals. The cluster variance proved to be a sum over the squared norms of the error models needed to go from the average dynamics to the individual signals.

While this is an interesting result in its own right, we can go one step further, and explicitly derive these average dynamics (and the dynamics of the error models). To do so, we need a way to estimate dynamic systems, starting from power cepstra. In practice, we will have to work with large amounts of signals (e.g. sensors in chemical plants, smart meters for every household in a country, ...), so we want to be able to estimate these models in an as computationally cheap and automated way as possible.

This seems a tall order, which we will deliver by using well-known results on symmetric polynomials, known as Newton's identities. Section 6.2 will introduce necessary concepts and results from the field of symmetric polynomials, which we then translate to a cepstral pole polynomial identification technique in Theorem 15 and Algorithm 5, in Section 6.3.

Section 6.4 will give some remarks on how to use this algorithm when dealing with non-stable systems, and section 6.5 will give numerical examples and illustrations corroborating these results:

- a numerical illustration on a system with three known poles will show the algorithm indeed manages to estimate the pole polynomial (Subsection 6.5.1),
- an analysis of the convergence behaviour and noise robustness of the proposed technique (Subsection 6.5.2),
- a clustering example with signals coming from two systems with poles that are close together (Subsection 6.5.3), corroborating the results on cluster centers from Section 5.1,
- a historic and classic example of AR modeling by Yule, on sunspot numbers, comparing results to that of the traditional least squares approach; the cepstral approach from this Chapter requires less data preprocessing (i.e. detrending) than the least squares one (Subsection 6.6.1),
- a contemporary example on structural health monitoring using AR coefficients, where we show the method presented in this Chapter to

provide more distinctive coefficients for distinguishing between normal and faulty behaviour, compared to the traditional least squares approach (Subsection 6.6.2).

6.2 Symmetric polynomials

Symmetric polynomials are polynomials $P(\alpha_1, \ldots, \alpha_n)$ in n variables (in this case, the n poles of a model, α_i), such that, for any permutation, σ, of the subscripts $\{1, \ldots, n\}$, it holds that

$$P\left(\alpha_{\sigma(1)}, \ldots, \alpha_{\sigma(n)}\right) = P(\alpha_1, \ldots, \alpha_n). \tag{6.1}$$

Special types of symmetric polynomials, namely the elementary symmetric and power sum polynomials, over the poles of an AR model, can be linked to the concepts on dynamical systems and the power cepstrum introduced in Chapter 2.

Well-known relations between these two types of polynomials will provide us with a way to link the pole polynomial and cepstrum coefficients of an AR model.

6.2.1 Elementary symmetric polynomials

The *elementary symmetric polynomials* over a set of variables $(\alpha_1, \ldots, \alpha_n)$ are defined as

$$e_l(\alpha_1, \ldots, \alpha_n) = \sum_{1 \le j_1 < j_2 < \ldots < j_l \le n} \alpha_{j_1} \alpha_{j_2} \cdots \alpha_{j_l}, \tag{6.2}$$

where $l \in \mathbb{N}$. This implies $e_l(\alpha_1, \ldots, \alpha_n) = 0, \ \forall l > n$.

Example 12. For $n = 3$:

$$\begin{aligned} e_0(\alpha_1, \alpha_2, \alpha_3) &= 1, \\ e_1(\alpha_1, \alpha_2, \alpha_3) &= \alpha_1 + \alpha_2 + \alpha_3, \\ e_2(\alpha_1, \alpha_2, \alpha_3) &= \alpha_1\alpha_2 + \alpha_1\alpha_3 + \alpha_2\alpha_3, \\ e_3(\alpha_1, \alpha_2, \alpha_3) &= \alpha_1\alpha_2\alpha_3. \end{aligned} \tag{6.3}$$

These polynomials are important, as they are elementary building blocks for all symmetric polynomials. Indeed, the fundamental theorem of symmetric

polynomials states that any symmetric polynomial has a unique representation in terms of e_l's [6].

More specifically, the linear factorization of a monic univariate polynomial can be expanded in terms of elementary symmetric polynomials, as follows

$$\prod_{i=1}^{n}(z-\alpha_i)=\sum_{l=0}^{n}(-1)^l e_l\left(\alpha_1,\ldots,\alpha_n\right)z^{n-l}. \tag{6.4}$$

This is known as Vieta's theorem [6].

Interpreting the variables $(\alpha_1,\ldots,\alpha_n)$ as the poles of an n-th order AR model, the coefficients, a_i of its pole polynomial, as in Equation (6.4), are now nothing more than the elementary symmetric polynomials of its poles, up to their sign, or

$$a_l=(-1)^l e_l\left(\alpha_1,\ldots,\alpha_n\right). \tag{6.5}$$

6.2.2 Power sum polynomials

A special type of symmetric polynomials are the *power sum polynomials*. Over a set of variables $(\alpha_1,\ldots,\alpha_n)$, these are defined as

$$p_l\left(\alpha_1,\ldots,\alpha_n\right)=\sum_{i=1}^{n}\alpha_i^l, \tag{6.6}$$

with $l\in\mathbb{N}$.

Example 13. For $n=3$:

$$\begin{aligned}
p_0\left(\alpha_1,\alpha_2,\alpha_3\right)&=3,\\
p_1\left(\alpha_1,\alpha_2,\alpha_3\right)&=\alpha_1+\alpha_2+\alpha_3,\\
p_2\left(\alpha_1,\alpha_2,\alpha_3\right)&=\alpha_1^2+\alpha_2^2+\alpha_3^2,\\
p_3\left(\alpha_1,\alpha_2,\alpha_3\right)&=\alpha_1^3+\alpha_2^3+\alpha_3^3.
\end{aligned} \tag{6.7}$$

From Equation (2.31), we interpret the power sum polynomials over the poles of a system in terms of its power cepstrum coefficients. We find

$$p_l\left(\alpha_1,\ldots,\alpha_n\right)=\sum_{i=1}^{n}\alpha_i^l=lc_h(l), \tag{6.8}$$

again momentarily interpreting the α_i as the poles of an n-th order AR-model with cepstrum coefficients $c_h(k)$, and $l\in\mathbb{N}$.

6.2.3 Newton's identities

The elementary symmetric and power sum polynomials can be related to each other. These relations are known as *Newton's identities* [9] and are defined recursively.

Definition 14. Newton's Identities relate the power sum polynomials p_l and elementary symmetric polynomials e_l over a set of variables $(\alpha_1, \ldots, \alpha_n)$, as

$$\begin{aligned} & l e_l \left(\alpha_1, \ldots, \alpha_n\right) \\ & \quad = \sum_{i=1}^{l} (-1)^{i-1} e_{l-i} \left(\alpha_1, \ldots, \alpha_n\right) p_i \left(\alpha_1, \ldots, \alpha_n\right). \end{aligned} \tag{6.9}$$

We can solve these recursive relations to equate the elementary symmetric polynomials to combinations of power sum polynomials (dropping the variables $(\alpha_1, \ldots, \alpha_n)$ to improve readability),

$$\begin{aligned} e_1 &= p_1, \\ e_2 &= \frac{1}{2}\left(p_1^2 - p_2\right), \\ e_3 &= \frac{1}{6}\left(p_1^3 - 3p_1 p_2 + 2p_3\right), \\ &\vdots \\ e_l &= \frac{(-1)^l}{l!} B_l\left(-p_1, -1!p_2, -2!p_3, \ldots, -(l-1)!p_l\right), \end{aligned} \tag{6.10}$$

with the B_l the complete exponential Bell polynomials [57], a type of polynomials that arise in combinatorics. They encode information on how a set can be partitioned.

For example, $B_3(x_1, x_2, x_3) = x_1^3 + 3x_1 x_2 + x_3$, as there is 1 way to partition a set of 3 variables as 3 groups of 1 element each, 3 ways to partition 3 in 1 group of 1 element and 1 group of 2 elements, and one way to partition 3 as 1 group of 3 elements.

We can now use these results to efficiently identify AR models from their cepstral coefficients.

6.3 Cepstral identification of autoregressive systems

We are now ready to identify the pole polynomial of an AR model from its model cepstrum coefficients, which can be calculated directly from the input/output-data.

Theorem 15 (Pole polynomial identification)**.** *Given an n-th order stable AR model, with pole polynomial $a(z)$, and power cepstrum coefficients $c_h(k)$, we have*

$$a(z) = \sum_{l=0}^{n} \frac{1}{l!} B_l \left(-1!c_h(1), -2!c_h(2), \ldots, -l!c_h(l)\right) z^{n-l}, \tag{6.11}$$

with the B_l the complete exponential Bell polynomials.

Proof. We start from the monic pole polynomial

$$a(z) = \prod_{i=1}^{n} \left(z - \alpha_i\right), \tag{6.12}$$

with $\alpha_i,\ i \in 1, \ldots, n$, the poles of the AR model. Expanding in terms of elementary symmetric polynomials, as in Vieta's theorem in Equation (6.4), we get

$$a(z) = \sum_{l=0}^{n} (-1)^l e_l \left(\alpha_1, \ldots, \alpha_n\right) z^{n-l}. \tag{6.13}$$

Using Equation (6.10), we can rewrite this expression in terms of Bell polynomials over power sums, as

$$a(z) = \sum_{l=0}^{n} \frac{1}{l!} B_l \left(-p_1, -1!p_2, -2!p_3, \ldots, -(l-1)!p_l\right) z^{n-l}. \tag{6.14}$$

Finally, the relation between power sums and cepstrum coefficients in Equation (6.8) gives us the result from the theorem,

$$a(z) = \sum_{l=0}^{n} \frac{1}{l!} B_l \left(-1!c_h(1), -2!c_h(2), \ldots, -l!c_h(l)\right) z^{n-l}. \tag{6.15}$$

□

Algorithm 5 Cepstral AR system identification

function CEPSTRAL_SYSID(c_h,$order$)
    Initialize $a(order) = 1$, with a a vector which will contain the coefficients of the differential equation.

    For $l = 1 : order$ **do**
        $a(order - l) \leftarrow \frac{1}{l!} B_l \left(-1!c_h(1), -2!c_h(2), \ldots, -l!c_h(l)\right)$
    End

    return a

Using this theorem, we now have a system identification algorithm, as described in Algorithm 5, fulfilling the objectives set out at the start of this Chapter.

The Bell polynomials can be explicitly written out in terms of their arguments as

$$B_n(x_1, \ldots, x_n) = \tag{6.16}$$
$$\sum_{k=1}^{n} \sum_{j_i} \frac{n!}{j_1! \cdots j_{n-k+1}!} \left(\frac{x_1}{1!}\right)^{j_1} \cdots \left(\frac{x_{n-k+1}}{(n-k+1)!}\right)^{j_{n-k+1}},$$

where the second sum is over all sequences $j_1, j_2, \ldots, j_{n-k+1}$ such that $j_1 + j_2 + \ldots + j_{n-k+1} = k$ and $j_1 + 2j_2 + \ldots + (n-k+1)j_{n-k+1} = n$.

With this representation, Equation (6.11) is similar to the results of [71], who arrived at explicit expressions linking LPC model parameters and cepstral coefficients in a different way. This builds on a body of literature in speech processing, where recursive relations for this problem have been known for quite some time. In [31], the authors mention that these recursions should not be used to calculate model parameters from cepstral coefficients, however, as this might result in unstable systems. Employing the power cepstrum, rather than the complex or real cepstrum, alleviates this problem.

The power spectral density (Equation (2.11)) does not take into account the phase information of the transfer function. A system $H(z)$ and $H(z^{-1})$ will have the same power spectrum. In the expansion of the power cepstrum coefficients in terms of poles (Equation (2.24)), we will always get a contribution as if the pole were stable, and a fortiori the resulting system will be stable. In Section 6.4, we will talk in more detail about these stability issues.

Example 16 (Second order system)**.** We illustrate the results with a very simple example of a system with two poles,

$$H(z) = \frac{1}{(z-\alpha_1)(z-\alpha_2)}. \tag{6.17}$$

This example will be easy to follow, and illustrates the concepts introduced before. The same approach holds for systems of any order.

We find, from Equation (2.24), the cepstrum coefficients

$$\begin{cases} c(1) = \alpha_1 + \alpha_2 \\ \\ c(2) = \dfrac{\alpha_1^2}{2} + \dfrac{\alpha_2^2}{2} \end{cases}, \tag{6.18}$$

which we can rewrite in terms of elementary symmetric polynomials as

$$\begin{cases} e_1 = \alpha_1 + \alpha_2 = c(1) \\ \\ e_2 = \alpha_1\alpha_2 = \dfrac{c(1)^2}{2} - c(2) \end{cases}. \tag{6.19}$$

The expressions on the right hand side of this equation are the Bell polynomials from Equation (6.11).

From Equation (6.4), we now find the pole polynomial

$$a(z) = (-1)^0 e_0 z^2 + (-1)^1 e_1 z + (-1)^2 e_2 \tag{6.20}$$

$$= z^2 - c(1)z + \frac{c(1)^2}{2} - c(2), \tag{6.21}$$

which is the result from Theorem 15.

In general, we are now done. As a bonus, for the second order system here, we can find exact solutions for the poles of the system as well,

$$\begin{cases} \alpha_1 = \dfrac{c(1)}{2} + \dfrac{1}{2}\sqrt{4c(2) - c(1)^2} \\ \\ \alpha_2 = \dfrac{c(1)}{2} - \dfrac{1}{2}\sqrt{4c(2) - c(1)^2} \end{cases}. \tag{6.22}$$

Note that these solutions consist of either complex conjugate poles (when $4c(2) - c(1)^2 < 0$) or two real poles (when $4c(2) - c(1)^2 > 0$), as they should for real signals.

For high-order models, we will in general not be able to find such exact solutions for the poles.

6.4 Stability

Up to now, we have always assumed stable models. Here, we extend the cepstral identification framework to non-stable systems.

Consider again the system from Equation (2.35)

$$Y(z) = \frac{1}{b(z)c(z)}U(z), \tag{6.23}$$

where $b(z)$ contains the stable poles, $|\beta_j| < 1$ and $c(z)$ the unstable poles, $|\gamma_j| > 1$.

The question we want to answer now is whether we can, based on input-output signal pairs only, determine whether the system is stable or not, and identify its pole polynomial $a(z) = b(z)c(z)$.

From Equation (2.36), we see that the complex cepstrum stores the power cepstrum coefficients belonging to the stable poles in positive values of k and the coefficients belonging to the unstable poles in negative values of k. We can then run Algorithm 5 on the positive and negative sequences separately.

This will give us $b(z)$ and $c(z^{-1})$ from the system in Equation (2.35). We then only have to substitute $z \to z^{-1}$ for the unstable part, multiply the two polynomials, and we find the full pole polynomial

$$a(z) = b(z)c(z). \tag{6.24}$$

6.5 Numerical examples

In this section, we illustrate the theoretical results obtained in this Chapter and provide several numerical examples to explore cepstral system identification in practice.

The first numerical illustration in Subsection 6.5.1, serves as a proof of concept, while the second example, with synthetic data, investigates convergence behaviour and noise robustness, in Subsection 6.5.2. We use the same example to show that the mean of a cepstral cluster is a good estimate of the average dynamics, in Subsection 6.5.3.

We work only on the system outputs, and thus assume that the output cepstra correspond to the relevant model cepstra, $c_y(k) = c_h(k)$.

6.5.1 Numerical illustration

As a first practical acquaintance with cepstral system identification, we consider a simple third order autoregressive system

$$H(z) = \frac{z^3}{z^3 - 0.5z^2 + 0.25z - 0.125}, \tag{6.25}$$

which has three poles, namely $z_1 = 0.5$, $z_2 = 0.5i$, and $z_3 = -0.5i$.

In order to identify this system, we simulate the response, $y(k)$, on a Gaussian white noise input signal[1], $u(k)$, of length $N = 5 \times 10^5$. Subsequently, the power cepstrum, $c_h(k)$, yields, via Algorithm 5, a third order model[2]

$$\hat{H}(z) = \frac{z^3}{z^3 - 0.497z^2 + 0.249z - 0.126}. \tag{6.26}$$

The poles and the transfer function of the identified model $\hat{H}(z)$ closely resemble those of the original system $H(z)$, as is also visualized in Figure 6.1.

At the moment, we assume that the order of the model is a user-input to the algorithm. Given the fact that the identification algorithm is computationally very cheap once the cepstrum is known, in practice, we could use a simple grid search to find an appropriate value for the model order n.

6.5.2 Synthetic data

Let us now turn to synthetic data, where we know exactly what generative models the underlying data comes from, to examine the properties of the proposed techniques.

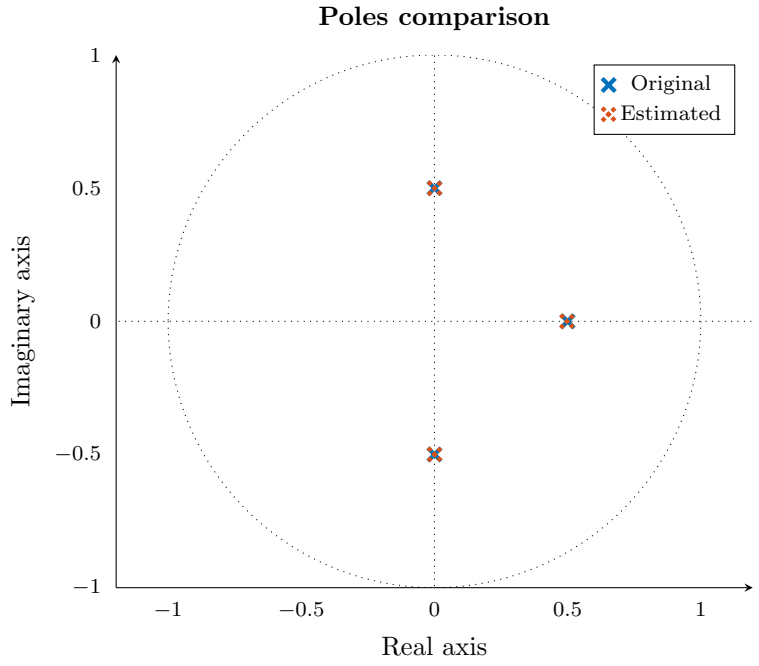
We consider again the third order system, $H(z)$, in Equation (6.25) and numerically examine the convergence behaviour and noise robustness of cepstral system identification. Moreover, we compare these results with those of the well-known least-squares regression approach.

Convergence behaviour

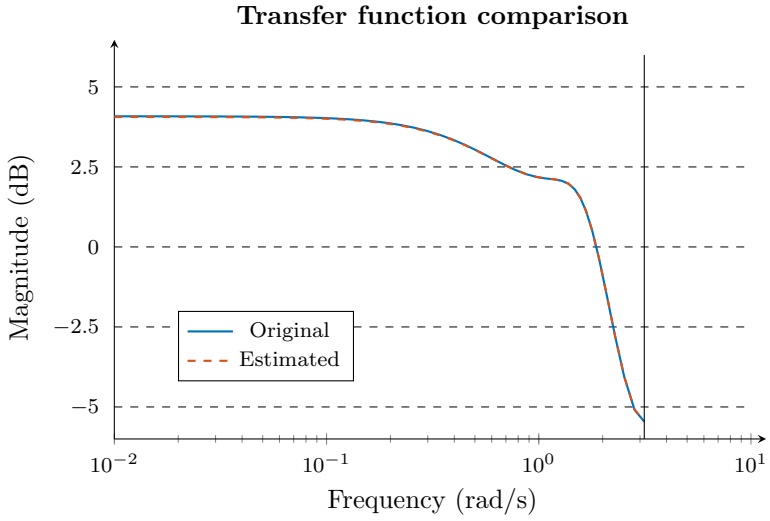
A first experiment compares the $\mathcal{H}_2$-error of the identified models for different data lengths N. The $\mathcal{H}_2$-norm of a stable system $H(z)$ is the root-mean-square

[1] In the case of white noise inputs, we had $c_y(k) = c_h(k)$, $\forall k \neq 0$. We therefore work only on the output signal.

[2] The computed cepstral coefficients change slightly, depending on the exact noise realization. Hence, the exact model parameters may vary somewhat for different random seeds.



(a) Poles



(b) Transfer functions

Figure 6.1: The model identified via the above described cepstral system identification procedure closely resembles the original system $H(z)$. The original poles are $z_1 = 0.5$, $z_2 = 0.5i$, and $z_3 = -0.5i$.

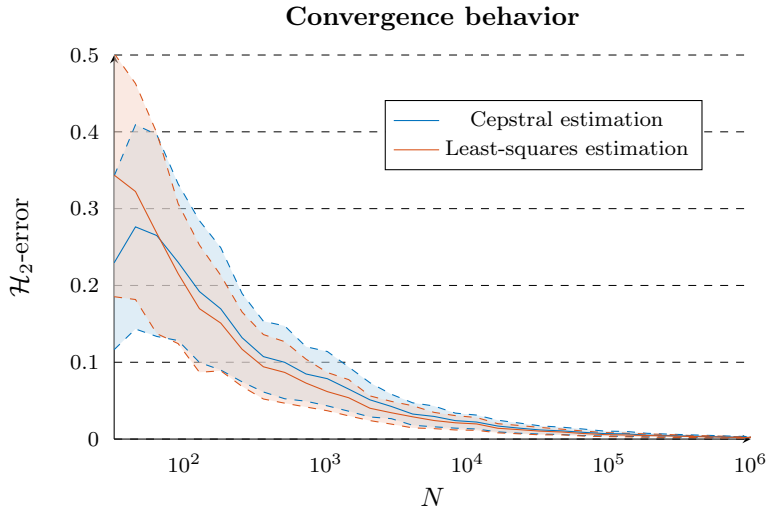


Figure 6.2: Both the cepstral and the least-squares procedure are applied 100 times to different input-output data of the same length coming from $H(z)$ in Equation (6.25) with white Gaussian input. The solid line shows how the mean of the $\mathcal{H}_2$-error decreases for longer data lengths, while the dashed lines represent the standard deviation. The cepstral system identification procedure seems to cope better with shorter time series than its least-squares counterpart.

of the impulse response of the system. Moreover, it also corresponds to the area underneath the transfer function $H(z)$. Therefore, the $\mathcal{H}_2$-error, which is the $\mathcal{H}_2$-norm of the error system $e(z) = H(z) - \hat{H}(z)$, represents the mismatch between the identified model $\hat{H}(z)$ and the original system $H(z)$.

Both the cepstral and the least-squares approach are applied 100 times to different responses of the same data length for the model in Equation (6.25). Figure 6.2 shows how the mean (over the 100 experiments) of the $\mathcal{H}_2$-error decreases, as expected, for longer data lengths. In this convergence experiment, the cepstral approach seems to cope better with shorter time series than its least-squares counterpart.

Noise robustness

In Figure 6.3, we investigate the influence of noise on the identification results. The response $y(k)$ simulated by the system in Equation (6.25) is contaminated with additive white Gaussian noise of different magnitudes, σ: $\tilde{y}(k) = y(k) +$

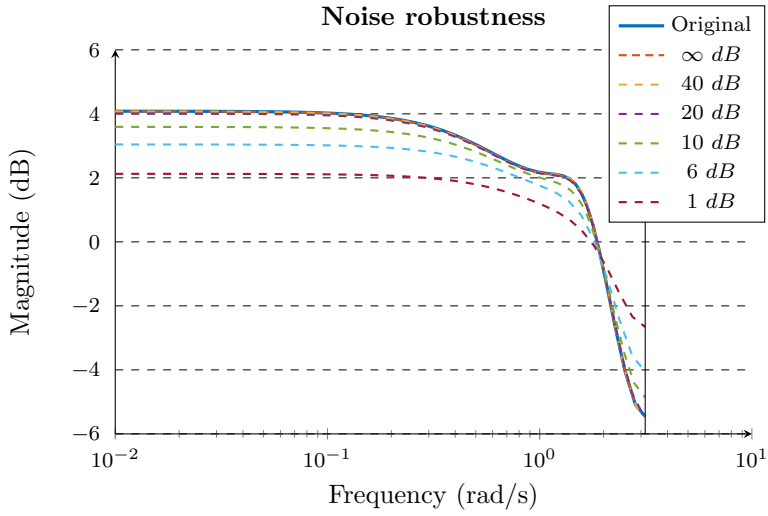


Figure 6.3: The figure above visualizes the influence of the signal-to-noise ratio (SNR) on the identification results. For high SNRs, the identified transfer function behaves almost exactly as the original one. Down to 10 dB, the results remain quite satisfactory. However, when the SNR drops below 10 dB, the cepstral identification procedure fails to identify the system.

$\sigma w(k)$. For high signal-to-noise ratios (SNRs), the identified transfer function behaves almost exactly as the original one. Down to 10 dB, the results remain quite satisfactory. However, when the SNR drops below 10 dB, the cepstral identification procedure fails to identify the system. An identification via the least-squares approach yields a similar noise robustness.

6.5.3 Mean of cepstral cluster

We now turn our attention to the interpretation of the average cepstrum in Equation (5.9), in the context of cepstral clustering.

Consider two systems,

$$H_1(z) = H(z) = \frac{1}{z^3 - 0.5z^2 + 0.25z - 0.125}, \tag{6.27}$$

as in Equation (6.25), and

$$H_2(z) = \frac{1}{z^3 - 0.4z^2 + 0.3025z - 0.121}, \tag{6.28}$$

with poles $z_1 = 0.4$, $z_2 = 0.55i$, and $z_3 = -0.55i$.

We simulate each system a hundred times with random standard white noise input signals. Denote the output signals of $H_1(z)$ as $y_{1,i}$, for $i \in \{1, 2, \ldots, 100\}$, with cepstra $c_{1,i}$, and those of $H_2(z)$ as $y_{2,i}$, for $i \in \{1, 2, \ldots, 100\}$, with cepstra $c_{2,i}$. Two such output signals, one for each system, are shown in Figure 6.4. The poles of these systems were purposely chosen close together, so the output signals are similar and the clustering challenge is non-trivial.

Performing cepstral k-means clustering (see Section 5.1) on this dataset, for two clusters, exactly separates the signals coming from each system. The first cluster, C_1, contains all cepstra $c_{1,i}$, while the second cluster, C_2 contains all cepstra $c_{2,i}$. Figure 6.5 shows the resulting labels.

Given a cepstrum, we can perform the cepstral system identification technique developed in this paper. We will compute and show the power spectral density of the resulting models to illustrate the claims made in this Chapter.

Figure 6.6 shows power spectral densities obtained in this way for various elements of $c_{1,i}$. In Figure 6.7, we show, for cluster C_1, the true power spectral density (i.e. the power spectral density associated with the system H_1), the geometric mean of all power spectral densities (i.e. we first identified the power spectral density for each signal, by using the system identification technique from this paper to estimate the difference equation, then computed the frequency response; the geometric average over all spectral densities obtained in this way is shown), and the power spectral density associated with the cluster center (i.e. the system identification technique applied to the cluster center, which itself is the average over all the cepstra present in the cluster). We compare this with the power spectrum of the euclidean average over all the signals in the cluster.

It is clear that the cluster center provides a better estimate for the average dynamics of the cluster than the euclidean average.

We can thus interpret the cluster centers obtained by cepstral clustering as good estimators of the average dynamics present in the cluster. Instead of having to find a model for each realization of the system and taking the geometric mean, it suffices to model the cluster center.

6.6 Applications on real-life data

In this section, we repeat the historical identification of Yule's sunspot numbers and use the cepstrum to identify autoregressive models that monitor structural health.

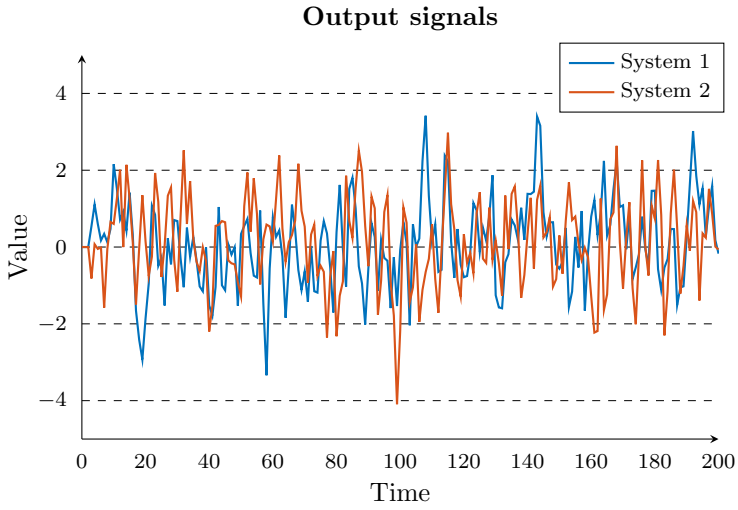


Figure 6.4: Two output signals, which are realizations of systems $H_1(z)$ in Equation (6.25) and $H_2(z)$ in Equation (6.28), respectively, with Gaussian white noise as inputs.

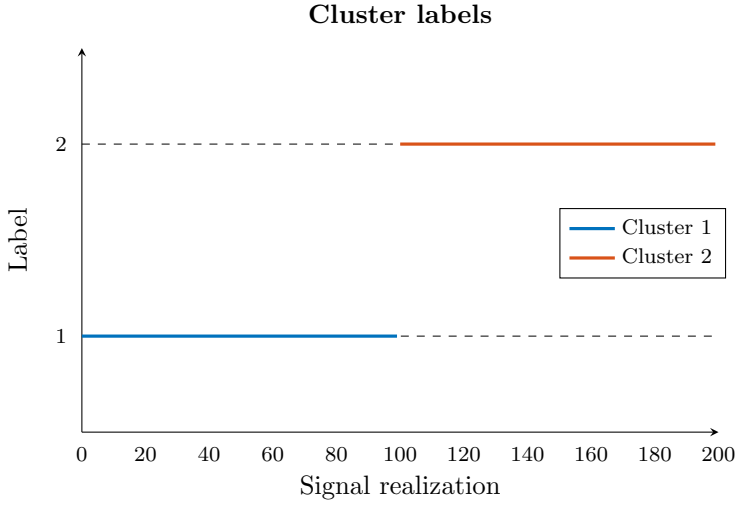


Figure 6.5: Results of cepstral k-means. The first 100 signals (0-99) are outputs of $H_1(z)$ in Equation (6.25), the last 100 belong to system $H_2(z)$ in Equation (6.28). Cepstral clustering separates them perfectly in two clusters, each representing one of the two systems. Examples of signals belonging to both clusters are shown in Figure 6.4.

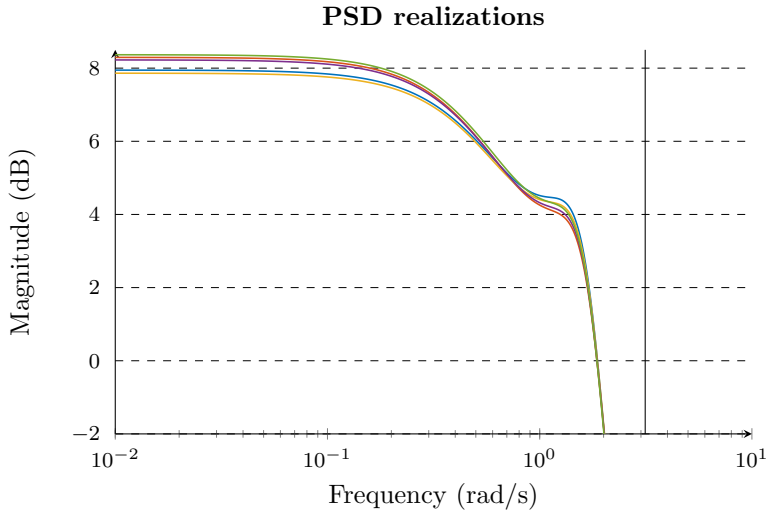


Figure 6.6: Power spectral density estimates of some of the signals in $y_{1,i}$, the output signals of system 1. There is some variance in these estimates.

6.6.1 Yule's historic sunspot numbers

Already in 1927, Yule [90] proposed the structure that we know today as an autoregressive model. He introduced this statistical technique to describe the historical series of recorded yearly sunspot numbers, a time series of visual sunspot counts initiated in 1848 by the Swiss astronomer Johann Rudolf Wolf [85]. Yule, however, called this series Wolfer's sunspot numbers, in honor of Wolf's successor at the Zürich observatory, Alfred Wolfer. These sunspot numbers are an important indicator for solar activity and the only direct information available to retrace the long-term evolution of the solar cycle, but also play a role in more recent discussions, like climate change, space flights, or telecommunication [16, 85].

Although the quality of the original data is questionable [85], especially the oldest observations, we repeat this historical experiment on the same data set as Yule (including the initial detrending[3] - see Figure 6.8), with yearly sunspot numbers between 1749 and 1924. As expected, both the cepstral and the least-squares identification approach closely match Yule's historical model. Figure 6.9 compares the different approaches.

[3]The detrending removes the best straight-line fit linear trend from the data. The trend is shown in Figure 6.8.

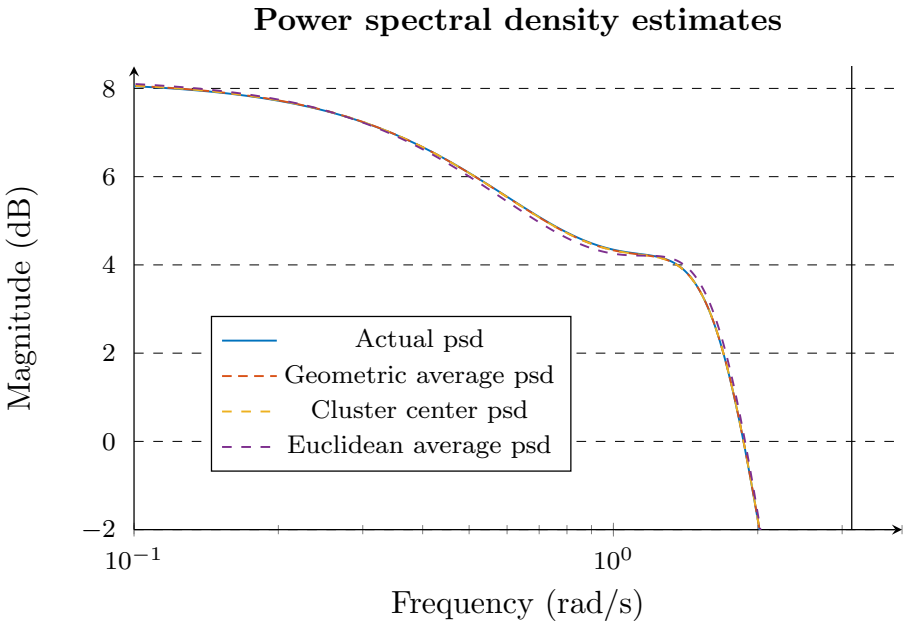


Figure 6.7: Power spectral density (PSD) estimates for various quantities, compared to the actual PSD. All of these results are for the output signals of system $H_1(z)$, in Equation (6.25). The geometric average PSD is the geometric average over the PSD's belonging to all the output signals of system $H_1(z)$ (for this plot, we identified a model for each signal, using the cepstral system identification proposed in this Chapter, computed their power spectral densities, and took the geometric average). The cluster center PSD is the one belonging to the cepstral cluster center. Note that the latter is equal to the geometric average, illustrating Equation (5.9), and they are both good estimators of the actual power spectral density. The Euclidean average PSD is the power spectral density belonging to the euclidean average over all signals in cluster 1. Note that this deviates from the actual PSD, showing the merit of cepstral clustering over traditional approaches, for the purpose of finding average dynamics.

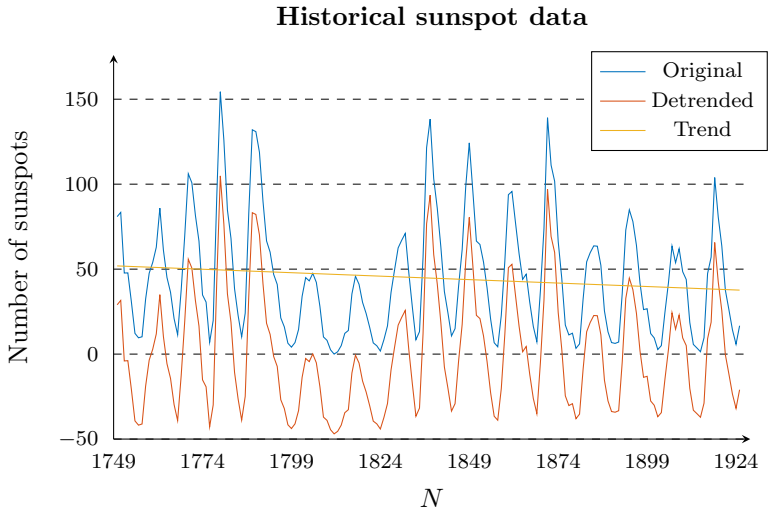


Figure 6.8: This figure visualizes Yule's historic sunspot numbers before and after detrending.

Furthermore, we notice that the cepstrum seems to cope very well with a trend in the data. Figure 6.10 shows how the identification results change when we work with the original data (without the intial detrending). The least-squares approach completely fails, while the cepstrum results in a satisfactory model that captures most of the system dynamics. We suspect that this stems from the fact that the power cepstrum stores information on the scale of the problem in its zero-th coefficient, which is not used in the identification technique, and conclude that this approach seems to be less sensitive to trends than the least-squares approach.

6.6.2 Structural health monitoring via autoregressive models

Many civil and mechanical systems require a close structural health monitoring (SHM) system, both from an economic and a safety perspective [72]. Damage identification based upon changes in dynamic response to vibrations is one of the few methods to monitor the structural health in a non-destructive way. One non-destructive approach is the use of autoregressive models to identify damages in given structures. Both the residuals [10, 24] and the coefficients [72] of the autoregressive models fitted on the motion measurements of a vibrating structure serve as important features to detect any structural damage.

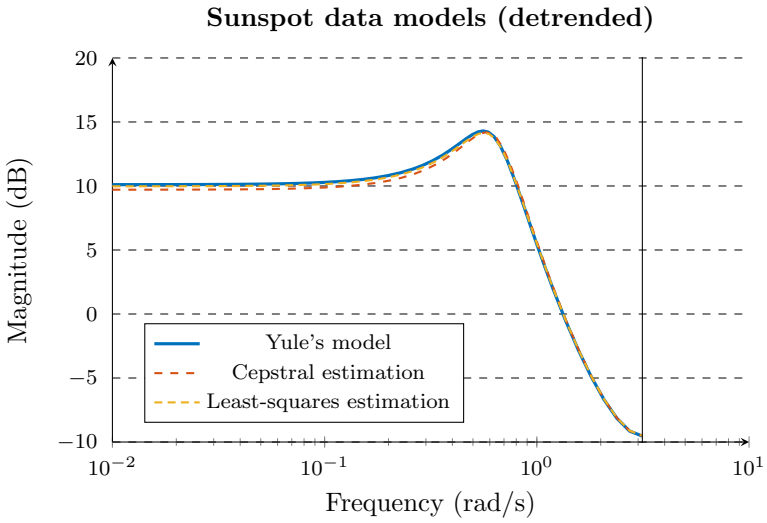


Figure 6.9: The identification of Yule's historic sunspot numbers via cepstral system identification and a least-squares approach. Both methods yield a close match to Yule's model.

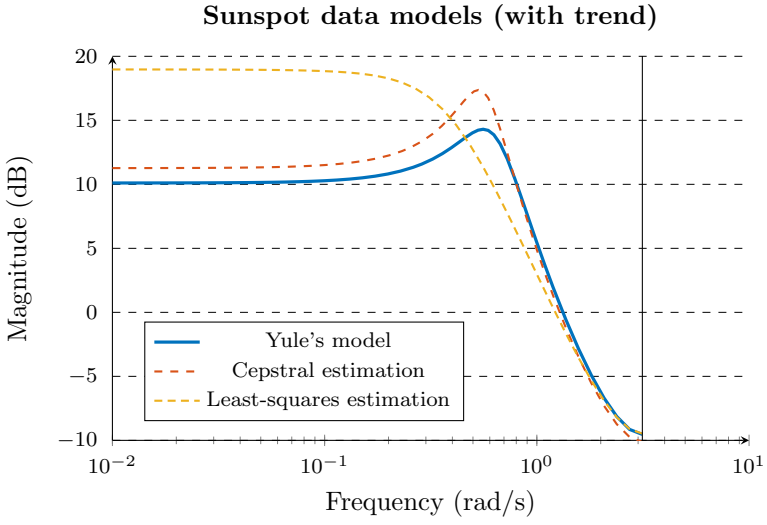


Figure 6.10: We reran the estimation experiment, but now without any data preprocessing. The data used here is the data as it was made available by Wolf and Wolfer. When we do not detrend the sunspot number data before identification, the least-squares approach fails completely to identify the system. However, the cepstrum results in a satisfactory result that captures most of the system dynamics.

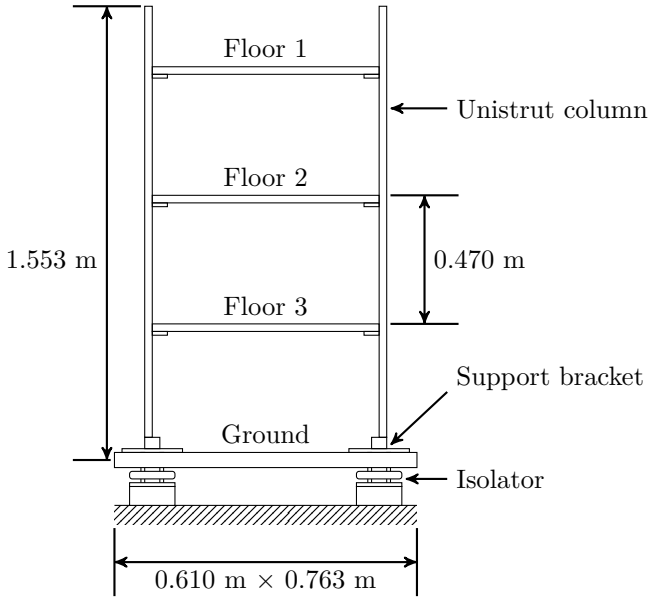


Figure 6.11: The three-story test setup of which we monitor its structural health, consists of an aluminium frame and four aluminium floor plates, connected through bolted connections. In undamaged state, all bolted connections are tightened properly. However, to simulate structural damage, we loosen or remove some bolts. At its bottom, a shaker vibrates the structure and several accelerometers measure the resulting movements of the different aluminium parts.

We apply this autoregressive approach on the measured response of a vibrating three-story frame structure (sketched in Figure 6.11) and use the coefficients of the fitted models to identify any structural damage. The structure consists of an aluminum frame with floor plates, connected through bolted connections. In undamaged state, all bolted connections are tightened properly. By loosening or removing some bolts, structural damage is simulated. The frame is then excited by a shaker at its bottom. Several accelerometers measure the movements of the different aluminium parts[4].

We follow the approach as described in the paper of Sohn et al. [72], where similar experiments are made on a bridge structure. We identify the autoregressive

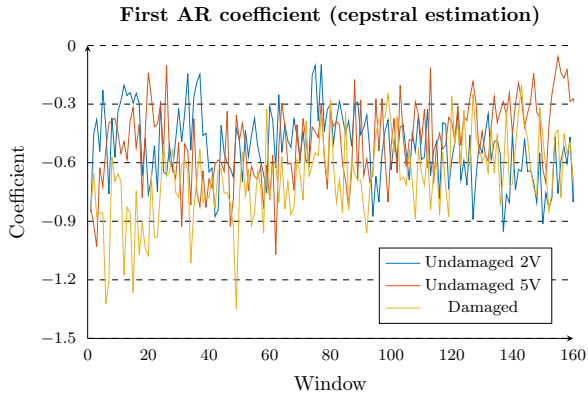
[4]More information about the test setup or the data set can be found on the website of Los Alamos National Laboratory `https://www.lanl.gov/projects/national-security-education-center/engineering/software/shm-data-sets-and-software.php`. The data set used in this example is the one named "Bookshelf Frame Structure – DSS 2000".

coefficients (order $n = 3$) of non-overlapping windows (window length $M = 50$) of the output of an accelerometer close to the defect, using the cepstral clustering approach in Algorithm 5. The interested reader finds information about the influence of these design parameters on the structural damage detection in [23].

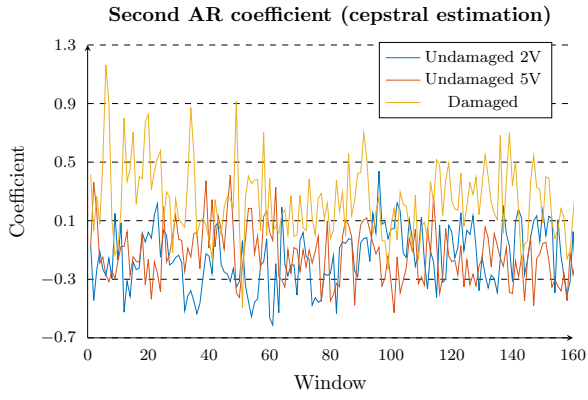
Input data on the movement of the shaker was not available. We thus, again, have to make the assumption that the dynamics contained in the output signal constitute the dynamics of the relevant model. In the cepstrum domain, this translates to $c_y(k) = c_h(k)$.

Results are shown in Figure 6.12. The experiment was done twice without damage, for different levels of excitation. The first experiment drives 2 Volts through the shaker (undamaged 2V), the second one applies 5 Volts (undamaged 5V). We then repeat the experiment with structural damage (removal of the bolt named L1C). We notice that the coefficients remain within the same range, for different levels of excitation of the shaker. The autoregressive coefficients in Figure 6.12 clearly differ from those in the undamaged situation, especially for the second and third coefficient (the first coefficient seems to be less sensitive to damage, an observation that has also been made by Sohn et al. [72]). These differences in coefficients then allow to distinguish between normal and damaged structures.

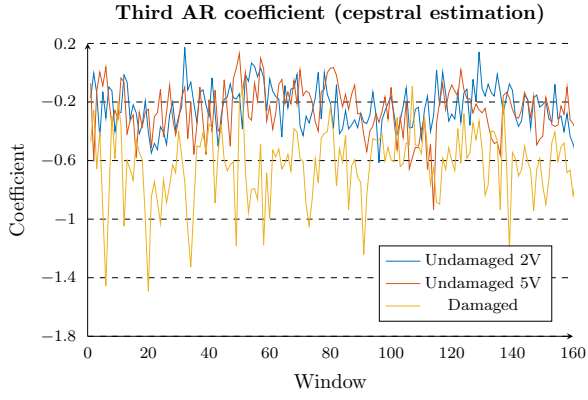
Finally, we tried to replicate the results with the least-squares approach. However, the difference between the damaged and undamaged autoregressive coefficients is not as distinctive (see for example the third coefficient in Figure 6.13). Perhaps the rather short window length is to blame here, in accordance with the results in Figure 6.2.



(a) First autoregressive coefficient for each window.



(b) Second autoregressive coefficient for each window.



(c) Third autoregressive coefficient for each window.

Figure 6.12: The coefficients of the third order autoregressive model fitted on the output of one accelerometer attached to the excited three-story frame clearly indicate when the frame contains some structural damage. Results of two experiments without, and one experiment with structural damage are shown. Especially the second and third damaged autoregressive coefficients differ clearly from their undamaged counterparts.

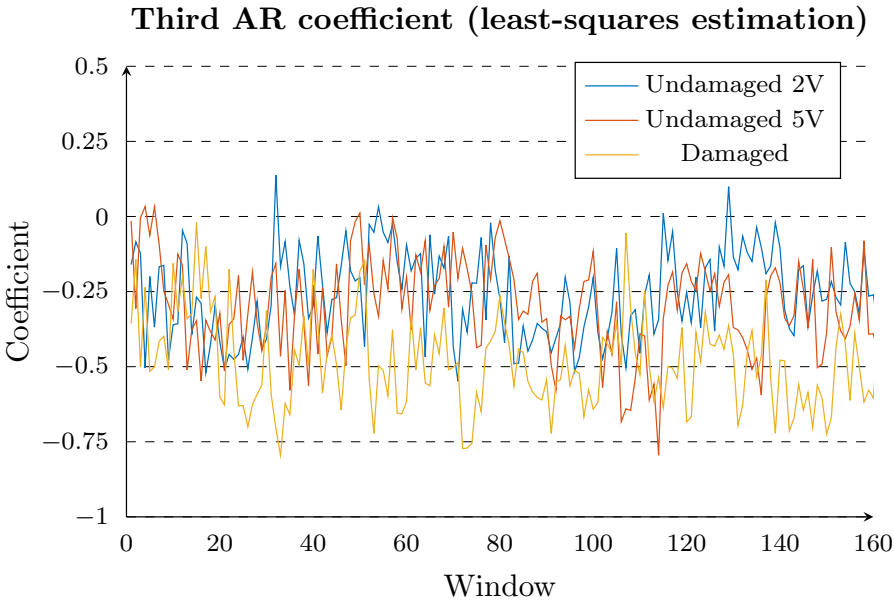


Figure 6.13: Third autoregressive coefficient for each window, now obtained with least-squares estimation. The differences of the autoregressive coefficients are less distinctive when we apply the least-squares approach instead of the cepstrum. We suspect that the length of the non-overlapping windows is the main culprit of this behaviour, as this problem seems to disappear with longer window sizes.

Chapter 7

A multiple-input multiple-output cepstrum

This paper was previously published as:
[44] Lauwers, O., Agudelo, O. M., and De Moor, B. A multiple-input multiple-output cepstrum. *IEEE Control Systems Letters 2*, 2 (2018), 272–277

Contribution: Oliver Lauwers was the main developer of the theoretical contributions of the paper, and contributed to the accompanying code. He performed the analysis of the experiments, and took the lead in writing the paper.

Abstract

This paper extends the concept of scalar cepstrum coefficients from single-input single-output linear time invariant dynamical systems to multiple-input multiple-output models, making use of the Smith-McMillan form of the transfer function. These coefficients are interpreted in terms of poles and transmission zeros of the underlying dynamical system.

We present a method to compute the MIMO cepstrum based on input/output signal data for systems with square transfer function matrices (i.e. systems with as many inputs as outputs). This allows us to do a model-free analysis.

Two examples to illustrate these results are included: a simple MIMO system with 3 inputs and 3 outputs, of which the poles and zeros are known exactly, that allows us to directly verify the equivalences derived in the paper, and a case study on realistic data. This case study analyses data coming from a (model of) a non-isothermal continuous stirred tank reactor, which experiences linear fouling. We analyse normal and faulty operating behaviour, both with and without a controller present. We show that the cepstrum detects faulty behaviour, even when hidden by controller compensation. The code for the numerical analysis is available online.

Linear systems; Fault detection

7.1 Introduction

In this paper, we present an extension of the definition of the cepstrum to MIMO systems. The main contributions of this paper are

- the definition of the MIMO cepstrum,
- its interpretation in terms of poles and zeros,
- a computational scheme to estimate the MIMO cepstrum in the case of a system with as many inputs as outputs. This allows a model-free analysis of the underlying dynamics of input/output signals. We present a control theory case study on linear fouling in a non-isothermal continuous stirred tank reactor.

For the case where there are unequal numbers of inputs and outputs, a computational scheme to estimate the cepstrum from input and output signals is still lacking.

The cepstrum is a long-standing and versatile technique in signal processing, first discussed in [8]. Originally, it was applied for the echo detection in seismic signals. The cepstrum has been used in a wide variety of applications, such as pitch detection in acoustic signals [59], analysis of mechanical problems [66] and human activity recognition [48].

Applications are not only diagnostic, but also include parameter estimation [59], system identification and prediction [17], and assessing dynamical (dis)similarity between signals [45]. One of the main advantages of cepstral techniques is a rich theoretical framework, with an interpretation of the coefficients in terms of poles and zeros of the model.

This notion of the cepstrum of a signal was developed in the context of stochastic Linear Time Invariant (LTI) Single-Input Single-Output (SISO) dynamical systems. A major drawback is the absence of a notion of cepstral coefficients in the case of multiple inputs and multiple outputs (MIMO systems). While it is possible to calculate cepstra of each individual output and input, and create a cepstral coefficient matrix, it is not clear how this can be interpreted in terms of poles and zeros of the MIMO system as a whole.

The definition of the cepstrum to MIMO systems presented in this paper produces a scalar coefficient sequence, that reduces to the normal definition of the cepstrum in the SISO case, but preserves the interpretation in terms of poles and zeros of the system in the MIMO case.

This paper is structured as follows. Section 7.2 presents the concepts, notation and definitions used throughout the paper. Section 7.3 extends the cepstrum to the MIMO case, and its interpretation in terms of poles and zeros of the model. It also introduces an algorithm to compute the cepstrum in the case of square transfer matrices. Section 7.4 presents two numerical illustrations: a simple, fully-known synthetic model and a case study on a realistic dataset concerning linear fouling in a non-isothermal continuous stirred tank reactor, violating some of the assumptions made in introducing the cepstrum. The techniques presented in this paper will turn out to be quite robust and allow us to analyse this realistic scenario. Section 7.5 will provide some general conclusions and possible paths for future work.

7.2 Concepts, notation and definitions

In this section, we explain some concepts, notation and definitions that are used in the rest of the paper. In Subsection 7.2.1, we give a brief overview of the cepstrum in the traditional SISO framework and repeat its interpretation

in terms of poles and zeros of the system. Subsection 7.2.2 introduces the concept of Smith-McMillan forms of MIMO transfer matrices, which we employ in defining a MIMO cepstrum.

7.2.1 SISO cepstrum

A LTI SISO dynamical system can be represented as a *state-space model*

$$\begin{cases} x(t+1) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}, \tag{7.1}$$

where t denotes (discrete) time, $x(t) \in \mathbb{R}^n$ are the *states* of the model, $u(t)$ and $y(t)$ *input* and *output* sequences respectively and A, B, C and D *system matrices* of appropriate dimensions. We assume the model to be *minimal* (i.e., *observable* and *controllable*).

Using the *z-transform* from [34] with $x(0) = 0$, this can be written as

$$Y(z) = H(z)U(z), \tag{7.2}$$

where $U(z)$ and $Y(z)$ are the *z-transform* of input and output respectively, and $H(z)$, the *transfer function* of the system, which is the z-transform of the *impulse response* of the system.

The transfer function is a rational function of z, with both numerator and denominator polynomials. We can write

$$H(z) = g\frac{b(z)}{a(z)}, \tag{7.3}$$

where $g \in \mathbb{R}$ is the constant *gain* of the system, and $a(z)$ and $b(z)$ are monic polynomials of degree n, the roots of which are respectively *poles* (denoted by $\alpha_i,\ i \in \{1, 2, \ldots, n\}$) and *zeros* (denoted by $\beta_i,\ i \in \{1, 2, \ldots, n\}$) of the system. For simplicity we assume the system to be *stable* (i.e. $|\alpha_i| < 1,\ \forall i$), *causally invertible* (i.e. n poles and n zeros)[1], *minimum-phase* (i.e. $|\beta_i| < 1,\ \forall i$)[2] and *minimal* (i.e. $\alpha_i \neq \beta_j,\ \forall i,\ \forall j$). To keep notation simple, we assume that all

[1]Note that we can assume this without loss of generality. For the purpose of this paper, we can always add zeros at $z = 0$ without changing results. We will see that the expression of the cepstrum in terms of the poles and zeros of the model, Equation (7.7), is not impacted by zeros at $z = 0$. For a more detailed discussion, see [17].

[2]The assumptions of stable and minimum-phase systems assures us that the series expansion in Equation (7.29) in the Appendix converges. Unstable poles and maximum-phase zeros can be included, as in the SISO case, but some conceptual issues remain. For a detailed discussion, see [59].

poles and zeros are *simple* (i.e., they have multiplicity 1) throughout this text. Extensions to multiple poles and zeros are possible but would burden notation.

The transfer function leads to the notion of *power spectral density*, defined as [34]

$$\Phi_H(\mathrm{e}^{i\omega}) = H(\mathrm{e}^{i\omega})\overline{H(\mathrm{e}^{i\omega})} = \left|H(\mathrm{e}^{i\omega})\right|^2 = \left|g\frac{b(\mathrm{e}^{i\omega})}{a(\mathrm{e}^{i\omega})}\right|^2. \tag{7.4}$$

Here, the subscript $\cdot_H$ denotes that it is the power spectral density of the transfer function. Similar notation for input and output leads to Φ_U and Φ_Y. Here, $U(\mathrm{e}^{i\omega})$ and $Y(\mathrm{e}^{i\omega})$ are estimated empirically using *Welch's method* [45, 84]. The overbar denotes the complex conjugate, i the imaginary unit and ω is the angular frequency.

The *(power) cepstrum*[3] of a system is then defined as

$$c_H(k) = \mathcal{F}^{-1}(\log \Phi_H(\mathrm{e}^{i\omega})), \tag{7.5}$$

where $\mathcal{F}^{-1}$ is the *inverse Fourier transform*. Similar definitions hold for the cepstra of input and output, c_U and c_Y.

The rationale behind the use of the cepstrum in signal processing comes from *homomorphic signal processing* [59]. A look at Equation (7.5) shows us that the cepstrum takes convolutions in the time domain to a multiplication in the power spectral domain and then to additions via the logarithm. To return to (a transformed version of) the time domain, the inverse Fourier transform is applied. The original convolutional structure is thus equivalent to an additive one. In other words, a convolution of two signals in the time domain, $u = u_1 * u_2$, will result in an addition of their cepstra, $c_U = c_{U_1} + c_{U_2}$. This rationale serves as an intuitive interpretation of the cepstrum: it serves as a transformation of the time domain that simplifies the convolutional structure typical in signal processing. A physical interpretation in terms of poles and zeros is given below.

The cepstrum coefficients can be interpreted in terms of poles and zeros of the underlying system. In particular, for a system with transfer function

$$H(z) = g\frac{b(z)}{a(z)} = g\frac{\prod_{i=1}^{n}(1-\beta_i z^{-1})}{\prod_{i=1}^{n}(1-\alpha_i z^{-1})}, \tag{7.6}$$

[3]The terminology *power* cepstrum comes from the fact that it is derived from the *power* spectrum of the signal. A similar concept, based on the z-transform itself, is known as the *complex* cepstrum. In this paper, we only discuss the power cepstrum. We will use the terms power cepstrum and cepstrum interchangeably.

we can show (see the Appendix for a derivation) that

$$c_H(k) = \sum_{j=1}^{n} \frac{\alpha_j^{|k|}}{|k|} - \sum_{j=1}^{n} \frac{\beta_j^{|k|}}{|k|} \quad \forall k \neq 0,$$
$$c_H(0) = \log(g^2). \tag{7.7}$$

These expressions link the power cepstrum to the poles and zeros of the underlying dynamics. It is this connection that makes the cepstrum a powerful signal processing technique. E.g., this property allows the cepstrum to be employed to define a similarity measure that takes into account underlying dynamics of signals [45, 18]. It is therefore a logical conclusion to demand that any extension of the cepstrum to the MIMO case should retain this interpretability.

In what follows, we will define an extension that is connected in a similar way to poles and zeros of the MIMO system. To do so, we introduce the Smith-McMillanform, making the connection between the transfer matrix and its poles and zeros more explicit.

7.2.2 Smith-McMillan form

For a system with l inputs and m outputs, a transfer matrix is a $m \times l$ matrix, the elements of which are rational functions. In this section, we explain the *Smith-McMillan form* of a rational matrix. This form of the transfer function is a pseudo-diagonal matrix, with non-zero diagonal elements consisting of polynomials. For the definition in this Subsection, we rely heavily on [34].

Pseudo-diagonalising the transfer matrix is done by applying *elementary operations* on a rational matrix, which are

- multiplication of a row/column by a constant,
- switching positions of two rows/columns,
- addition of a polynomial multiple of one row/column to another.

These elementary operations can be represented as matrices, which multiply a rational matrix from the left for row operations, and from the right for column operations.

Combining the corresponding row operations into a *unimodular* (i.e. of constant determinant) polynomial matrix $V_1(z)$, and the corresponding column operations

into a unimodular polynomial matrix $V_2(z)$, we can write for a transfer matrix $H(z)$

$$V_1(z)H(z)V_2(z) = M(z), \tag{7.8}$$

with

$$M(z) = \begin{pmatrix} \text{diag}\left\{\frac{b_i(z)}{a_i(z)}\right\} & 0 \\ 0 & 0 \end{pmatrix}, \tag{7.9}$$

where $a_{i+1}(z)|a_i(z)$ (i.e. $a_{i+1}(z)$ exactly divides $a_i(z)$), $b_i(z)|b_{i+1}(z)$ and $i \in \{1, 2, \ldots, r\}$, with r the normal rank of $H(z)$. Any constant gains are absorbed into the unimodular matrices in Equation (7.8).

We denote

$$b(z) = \prod_{i=1}^{r} b_i(z), \qquad a(z) = \prod_{i=1}^{r} a_i(z), \tag{7.10}$$

the *zero* and *pole polynomial* respectively, i.e. the solutions of $b(z) = 0$ and $a(z) = 0$, equal to the zeros and poles of the transfer matrix $H(z)$.

In fact, poles and zeros can be (and are) defined for MIMO systems as the roots of the diagonal elements of the Smith-McMillan form. A more detailed discussion on MIMO poles and zeros can be found in [34].

7.3 MIMO cepstrum

For the MIMO case, the assumptions on the transfer matrix are the same as in the SISO case, presented directly after Equation (7.3), with one notable exception: as the transfer matrix is of size $m \times l$, and therefore not necessarily square, we cannot assume invertibility of $H(z)$. We replace it by an assumption on the normal rank of the transfer matrix:

$$r = \min\{m, l\}. \tag{7.11}$$

Based on the size of $H(z)$ (i.e. the amount of inputs, l relative to the amount of outputs, m), this assumption is equivalent to:

$\mathbf{m > l}$ we assume *left-invertibility*, i.e., there exists an $l \times m$ matrix $L(z)$ such that $L(z)H(z) = \mathbb{1}_l$, where $\mathbb{1}_l$ is the $l \times l$ identitiy matrix; this matrix $L(z)$ is the *left-inverse* of $H(z)$,

$\mathbf{m < l}$ we assume *right-invertibility*, i.e., there exists an $m \times l$ matrix $R(z)$ such that $H(z)R(z) = \mathbb{1}_m$, where $\mathbb{1}_m$ is the $m \times m$ identitiy matrix; this matrix $R(z)$ is the *right-inverse* of $H(z)$,

$\mathbf{m = l}$ we assume the matrix to be invertible.

7.3.1 Definition of the MIMO cepstrum

With the assumptions made above, we extend the cepstrum to the MIMO case, for a transfer matrix $H(z)$ with Smith-Mcmillan form $M(z)$ and $m < l$, as

$$c_M(k) = \mathcal{F}^{-1}\left(\log\det\left(M(\mathrm{e}^{i\omega})\overline{M(\mathrm{e}^{i\omega})}^{\mathsf{T}}\right)\right), \tag{7.12}$$

with $\cdot^{\mathsf{T}}$ denoting the matrix transpose. For the case where $m > l$ the definition is the same, but with the position of the transpose switched. For $m = l$, these are equivalent.

In the next Subsection, we will show how to interpret this extension of the cepstrum in terms of poles and zeros of the transfer matrix.

7.3.2 Interpretation in terms of poles and zeros

In this section, we work with the assumption that the transfer matrix is right-invertible (i.e., $m < l$). The other cases are completely analogous, with transposes switching places, and we will not repeat the derivation.

From Equation (7.9), and the assumption that $H(z)$ (and therefore $M(z)$) has normal rank $r = \min\{m, l\} = m$, we have (with $i \in \{1, 2, \ldots, r\}$)

$$M(\mathrm{e}^{i\omega})\overline{M(\mathrm{e}^{i\omega})}^{\mathsf{T}} = diag\left\{\left|\frac{b_i(\mathrm{e}^{i\omega})}{a_i(\mathrm{e}^{i\omega})}\right|^2\right\}. \tag{7.13}$$

Taking the determinant, and applying Equation (7.10), we find

$$\det\left(M(\mathrm{e}^{i\omega})\overline{M(\mathrm{e}^{i\omega})}^{\mathsf{T}}\right) = \left|\frac{b(\mathrm{e}^{i\omega})}{a(\mathrm{e}^{i\omega})}\right|^2. \tag{7.14}$$

This ratio of the zero and pole polynomial, however, is equivalent to the SISO case, where these polynomials are directly encapsulated in the transfer function (7.6). The MIMO problem can now be solved in exactly the same way, following the derivation in the Appendix. We again get the result

$$\begin{aligned} c_M(k) &= \sum_{j=1}^{n} \frac{\alpha_j^{|k|}}{|k|} - \sum_{j=1}^{n} \frac{\beta_j^{|k|}}{|k|} \quad \forall k \neq 0. \\ c_M(0) &= 0. \end{aligned} \tag{7.15}$$

Note that the difference for $k = 0$ between the MIMO and SISO cases stems from the fact that, in the Smith-McMillan form, the constant gain is absorbed

in the unimodular matrices in Equation (7.8). This gives an interpretation of the MIMO cepstrum in terms of poles and zeros of the underlying model.

The power cepstrum can now be readily derived whenever the Smith-McMillan form (7.8) is available. In general, it is not straightforward to compute it given only input and output signals. When $m \neq l$, we have no way of computing the cepstrum without explicitly estimating a model and deriving the Smith-McMillan form. For the case where $m = l$, we present a way to do so in the next Subsection.

7.3.3 Computation when $m = l$

When there are as many inputs as outputs, we will prove that definition (7.12) is equivalent (except for $k = 0$) to

$$c_{H,\text{comp}}(k) = \mathcal{F}^{-1}(\log \det \Phi_H(\mathrm{e}^{i\omega})), \tag{7.16}$$

with Φ_h the transfer matrix power spectrum, defined as

$$\Phi_H(\mathrm{e}^{i\omega}) = H(\mathrm{e}^{i\omega})\overline{H(\mathrm{e}^{i\omega})}^{\mathsf{T}}. \tag{7.17}$$

We can calculate the determinant of the power spectrum, using the Smith-McMillan form (7.8), as

$$\det \Phi_H = \det\left(V_1^{-1} M V_2^{-1} \overline{V_1^{-1} M V_2^{-1}}^{\mathsf{T}}\right). \tag{7.18}$$

Here, V_1^{-1} and V_2^{-1} are the inverse of the unimodular matrices in Equation (7.8), and therefore themselves unimodular. We dropped the variables to make notation easier, but we understand all the matrices involved to be evaluated on the unit circle, e.g., $\Phi_H = \Phi_H(\mathrm{e}^{i\omega})$. Working out the transpose, and using the fact that the determinant is a multiplicative map[4] for square matrices (i.e, for square matrices A and B, $\det(AB) = \det(A)\det(B)$), we can write

$$\det \Phi_H = \det\left(V_1^{-1}\overline{V_1^{-1}}^{\mathsf{T}}\right) \det\left(M\overline{M}^{\mathsf{T}}\right) \det\left(V_2^{-1}\overline{V_2^{-1}}^{\mathsf{T}}\right). \tag{7.19}$$

Unimodular matrices are matrices that have, by definition, a constant determinant. Denote the determinant of V_1^{-1} as c_{V_1} and that of V_2^{-1} as c_{V_2}, which leads to

$$\det \Phi_H = |c_{V_1}|^2 |c_{V_2}|^2 \det\left(M\overline{M}^{\mathsf{T}}\right). \tag{7.20}$$

[4] This is the troublesome step when $m \neq l$ and there will not necessarily be a straightforward equivalence between $\det \Phi_H$ and $\det\left(M\overline{M}^{\mathsf{T}}\right)$. A generalization of the multiplicative property, the Binet-Cauchy theorem [39, 80], may offer a solution, which we will not explore further here.

Using this, and comparing Equation (7.16) and (7.12),

$$c_{H,\text{comp}}(k) = c_M(k) + \mathcal{F}^{-1} \log \left(|c_{V_1}|^2 |c_{V_2}|^2 \right). \tag{7.21}$$

Following the derivation in the Appendix, we see that this leads to

$$\begin{aligned} c_{H,\text{comp}}(k) &= c_M(k) && \forall k \neq 0, \\ c_{H,\text{comp}}(0) &= \log \left(|c_{V_1}|^2 |c_{V_2}|^2 \right). \end{aligned} \tag{7.22}$$

While it is true that we cannot, in general, know the size of the error of our estimation on $c_M(0)$, in practical applications of the cepstrum (see for example the distances defined in [54, 18, 45]), this coefficient is often not important, as it contains only information on the gain of the system.

One last step, to be able to compute the MIMO cepstrum based on input/output data of a system, is to estimate $\log\det \Phi_H$. We know $\Phi_Y = \Phi_H \Phi_U$, and easily see that, for square systems,

$$\log\det \Phi_H = \log\det \Phi_Y - \log\det \Phi_U. \tag{7.23}$$

Equivalently, since the inverse Fourier transform is a linear operator, we can calculate the cepstra of input and output with Equation (7.16) and write

$$c_{H,\text{comp}}(k) = c_{Y,\text{comp}}(k) - c_{U,\text{comp}}(k), \quad \forall k \neq 0. \tag{7.24}$$

Since $u(k)$ and $y(k)$, the input and output signals, are given, we have found a data-driven, model-free way to compute the MIMO cepstrum for systems with $m = l$.

In the next Section, we will give some numerical illustrations and applications.

7.4 Illustration and application

In this Section, we first give a numerical illustration on synthetic data coming from a simple model. This will provide insight in the techniques presented in this paper. Afterwards, we show numerical results for a case study on faults in a Non-isothermal Continuous Stirred Tank Reactor. This is a realistic dataset and violates some of the assumptions made in this text. The cepstrum turns out to be quite robust. The code used to analyse these models is available online[5].

[5] https://github.com/Olauwers/MIMOcepstrum

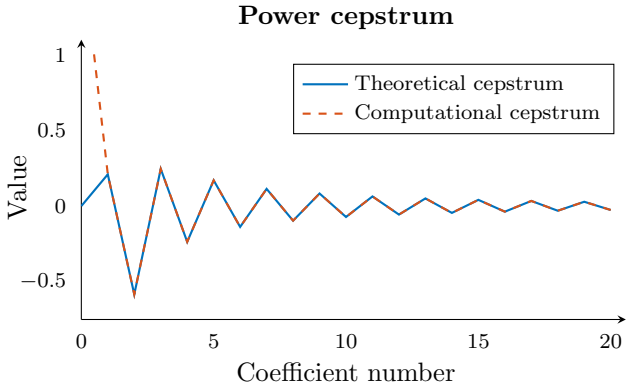


Figure 7.1: The theoretical cepstrum from Equation (7.15) and the estimate presented in Subsection 7.3.3 for the synthetic data from Subsection 7.4.1. We see that the computational cepstrum is a very good estimate of the exact one. The differences (for $k \neq 0$) are of order 10^{-4}.

7.4.1 Numerical illustration

The first example serves as a numerical illustration. It consists of a fully known, synthetic dataset, generated from an easy-to-understand model. The purpose of this first example is to show that the results obtained in Subsection 7.3.3, namely the equivalence between the proposed algorithm and the theoretical results from Subsection 7.3.2, indeed hold true numerically.

We implement (in MATLAB) a simple MIMO system with 3 inputs and 3 outputs. The poles, zeros and gains of the individual entries are chosen randomly, but it is made sure that the transmission zeros of the MIMO system are minimum-phase. The seed of the random number generator was set to default, for reproducibility. The system is generated as a state-space model, which behaves better numerically in the MIMO case. The transmission zeros of the MIMO system are $\beta_i = \{-0.9681, 0.4419, 0.0916 \pm 0.1453i\}$ and the poles are $\alpha_i = \{0.1786 \pm 0.3300i, -0.2769 \pm 0.1793i, 0.0634\}$.

We generate a white noise input of length 2^{16}, with random gains for every individual input channel. The computation in Subsection 7.3.3 is used as an estimate of the cepstrum, and compared with the exact cepstrum in Equation (7.15). Results are shown in Fig. 7.1. The computational cepstrum is indeed a very good estimate of the exact one.

The white noise data results in a power cepstrum that is non-zero for $c_{H,\text{comp}}(0)$, but vanishes everywhere else, analogous to the SISO case [17]. This was to be

expected, but provides an extra argument that the definition presented in this text is a natural generalization.

7.4.2 Case study: fouling in non-isothermal continuous stirred tank reactor

Continuous Stirred Tank Reactors (CSTRs) are one of the most important and fundamental units in chemical industry. They are characterized by highly nonlinear dynamics and they pose a challenging problem for fault prognosis and early fault detection algorithms. The CSTR considered in this section is a non-isothermal reactor where a single, first order, irreversible, and exothermic reaction takes place ($A \rightarrow B$).

The model of the reactor consists of the following nonlinear ordinary differential equations:

$$\frac{dC_A}{dt} = \frac{q}{V}\left(C_{A_f} - C_A\right) - C_A k_0 e^{-\frac{E}{RT}} + v_1 \tag{7.25}$$

$$\frac{dT}{dt} = \frac{q}{V}\left(T_f - T\right) - \frac{\Delta H}{\rho C_p} C_A k_0 e^{-\frac{E}{RT}} + \frac{UA}{V\rho C_p}\left(T_j - T\right) + v_2$$

with C_A and T the reactant concentration and temperature respectively inside the reactor, T_j the jacket temperature, q the flow rate of the feed flow, T_f the temperature of the feed flow, C_{A_f} the concentration of the reactant in the feed flow, and v_1, v_2 independent system noise processes, with $v_i \sim N(0, \sigma^2_{v_i} = 0.01)$. Parameters of the model and their numerical values (taken from [46]) are: $V = 100$ L (volume of the mix), $k_0 = e^{13.4}$ min^{-1} (kinetic constant), $E/R = 5360$ K (E is the activation energy and R is the ideal gas constant), $(-\Delta H) = 17835.821$ J/mol (heat of the reaction), $\rho = 1000$ g/L (density), $C_p = 0.239$ J/g/K (specific heat), and $UA = 11950$ J/min/K (U is the overall heat transfer coefficient, A is the area of the heat exchange between reactor wall and jacket) in ideal conditions (no fouling). Under normal conditions, the operating point of the reactor is given by $C_A^* = 0.2$ mol/L, $T^* = 446$ K, $q^* = 100$ L/min, $T_j^* = 419$ K, $T_f^* = 400$ K and $C_{A_f}^* = 1$ mol/L.

A control system consisting of two PID controllers and a decoupler keeps C_A and T around their nominal values C_A^* and T^*, by manipulating the jacket temperature T_j and the flow rate of the feed flow q. The control law in the Laplace domain is as follows [46]:

$$\begin{bmatrix} q(s) \\ T_j(s) \end{bmatrix} = \begin{bmatrix} 5 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} (K_p + K_d s + K_i/s) E_C(s) \\ (K_p + K_d s + K_i/s) E_T(s) \end{bmatrix} \tag{7.26}$$

with $E_C(s) = C_A(s) - C_A^*(s)$, $E_T(s) = T(s) - T^*(s)$, $K_p = 1$, $K_d = 0.1$ and $K_i = 10$.

Fouling, the accumulation of unwanted material on a heat transfer surface that increases its thermal resistance, is one of the most serious issues in heat transfer equipment. We distinguish two types of fouling: asymptotic and linear. In asymptotic fouling the resistance to heat transfer increases fast when the operation starts and becomes asymptotic to a steady state value at the end. In linear fouling the resistance increases linearly during the entire process operation. Here, we consider the second type, linear fouling. The overall heat transfer coefficient U multiplied by the heat exchange area A in (7.25) is given by the equation (t in minutes)

$$U(t)A = \begin{cases} 11950, & t \leq 5000 \\ 11950 - 0.8365(t - 5000), & t > 5000 \end{cases}. \tag{7.27}$$

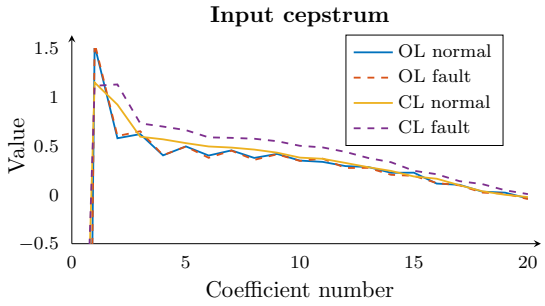
Two datasets of 10000 points have been generated (sampling time = 1 min), one when the controller is active and one when the controller has been switched off. Only the controlled (C_A and T) and manipulated variables (q and T_j) are measured, which are contaminated with measurement noise: $\eta_{C_A} \sim N(0, \sigma^2_{C_A} = 10^{-5})$, $\eta_T \sim N(0, \sigma^2_T = 0.005)$, $\eta_q \sim N(0, \sigma^2_q = 10^{-6})$, and $\eta_{T_j} \sim N(0, \sigma^2_{T_j} = 10^{-6})$.

We estimate the cepstra of input, output and of the process itself and show that the MIMO cepstrum indeed captures the dynamics of the processes and controller involved. The 200 data points around the fault are omitted. Results are shown and discussed in Fig. 7.2a for the input signals, Fig. 7.2b for the output signals and Fig. 7.2c for the process dynamics. Notice that the cepstrum of the process explicitly shows the change in the reactor dynamics, whether the controller is on or off, and detects hidden faults in a process, without having to model the process.

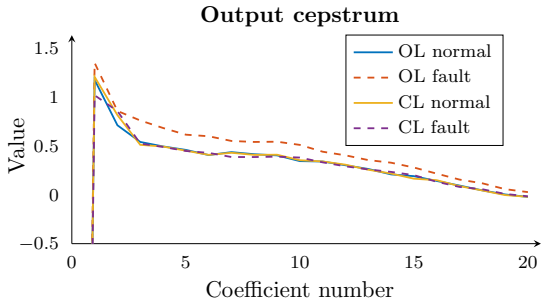
This makes the MIMO cepstrum a very promising technique for building new anomaly detection or fault prognosis algorithms, for example by extending and employing the distance measure from [17, 45, 54] in a clustering algorithm.

7.5 Conclusions and future work

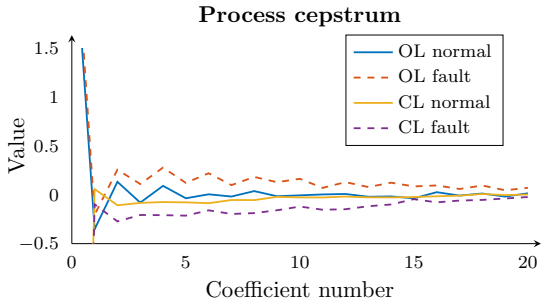
In this paper, we introduced a new definition for the power cepstrum, that extends it to systems with multiple inputs and outputs, based on the Smith-McMillan form of the system. This new power cepstrum is then interpreted



(a)



(b)



(c)

Figure 7.2: The cepstrum coefficients for input, output and process. We show both normal and faulty operating behaviour, with (CL) and without (OL) the controller. The cepstral coefficients for the process in (c) clearly show different fingerprints for the different cases. The normal operating behaviour changes only slightly when turning on the controller, capturing the extra dynamics of the controller. However, faulty operating behaviour results in a deviation from the normal operating behaviour, both in OL and, notably, in CL, capturing faults hidden by the controller. We can see the different contributions to this process cepstrum (see Equation (7.24)) in (a) and (b): (a) shows the change in the input dynamics in the faulty regime, when the controller starts compensating. (b) shows the change in the output in the faulty regime when the controller is turned off.

in terms of the poles and zeros of the underlying models. For systems with as many inputs as outputs, we provide a method to calculate the power cepstrum based on input and output data.

We then illustrate the theoretical results from this paper with a numerical example on a simple synthetic model, and on a realistic dataset concerning a Non-isothermal Continuous Stirred Tank Reactor. We show that the methods and interpretations presented here indeed hold numerically.

The Reactor case study shows potential to leverage the extended power cepstrum as an anomaly detection technique. We believe that every fault will have its own "fingerprint" in the cepstrum domain, which will then allow us to discern between different (perhaps compounded) types of faults.

Future work includes extending the distance in [45] to the MIMO case, providing data-driven ways to compute the cepstrum in the case of unequal number of inputs and outputs and proving links with canonical correlations and mutual information, as in the SISO case [17].

Acknowledgements

Work financially supported by FWO-Flanders (EOS Project no 30468160 (SeLMA)), several PhD postdoc grants (FWO) and innovation mandate grant (VLAIO), EU H2020-SC1-2016-2017 (Grant 727721: MIDAS Meaningful Integration of Data, Analytics and Services), KU Leuven BOF C16/15/059 (nD Roots) & C32/16/013. O. L. is a FWO SB fellow.

7.A Interpretation of SISO cepstrum in terms of poles and zeros

In this Appendix, we derive the results in Equation (7.7). Starting from Equation (7.6) and using Equation (7.4), we write

$$\begin{aligned}\log(\Phi_H(\mathrm{e}^{i\omega})) &= \log(H(\mathrm{e}^{i\omega})\overline{H(\mathrm{e}^{i\omega})}) \\ &= \log(g^2) + \sum_{i=1}^{n} \left(\log\left(1-\beta_i \mathrm{e}^{-i\omega}\right) + \log\left(1-\overline{\beta}_i \mathrm{e}^{i\omega}\right)\right) \\ &\quad - \sum_{i=1}^{n} \left(\log\left(1-\alpha_i \mathrm{e}^{-i\omega}\right) + \log\left(1-\overline{\alpha}_i \mathrm{e}^{i\omega}\right)\right).\end{aligned} \tag{7.28}$$

Employing the series expansion

$$\log(1-x) = -\sum_{k=1}^{\infty} \frac{x^k}{k} \quad \forall |x| < 1, \tag{7.29}$$

we find

$$\begin{aligned} &\log(\Phi_H(\mathrm{e}^{i\omega})) \\ &= \log(g^2) - \sum_{i=1}^{n} \left(\sum_{k=1}^{\infty} \frac{\beta_i^k}{k} \mathrm{e}^{-ik\omega} + \sum_{k=1}^{\infty} \frac{\overline{\beta}_i^k}{k} \mathrm{e}^{ik\omega} \right) \\ &\quad + \sum_{i=1}^{n} \left(\sum_{k=1}^{\infty} \frac{\alpha_i^k}{k} \mathrm{e}^{-ik\omega} + \sum_{k=1}^{\infty} \frac{\overline{\alpha}_i^k}{k} \mathrm{e}^{ik\omega} \right). \end{aligned} \tag{7.30}$$

The final step consists of noting that the $c_H(k)$, are the inverse Fourier transform of $\log(\Phi_H)$, or

$$\sum_{k=-\infty}^{\infty} c_H(k) e^{-ik\theta} = \log \Phi_H(e^{i\theta}). \tag{7.31}$$

Matching Equations (7.31) and (7.30), we obtain Equation (7.7).

Note that this, in essence, would mean we can reverse this procedure and calculate poles and zeros starting from the power cepstra. This would force us to make some further assumptions on amount of poles and zeros, and is beyond the scope of this paper.

Chapter 8

The MIMO weighted cepstral distance

The extended definition of the power cepstrum to the case of MIMO systems from Chapter 7 lends itself to a straightforward extension of the weighted cepstral norm.

We start from two MIMO systems, with transfer function matrices H_1 and H_2, having an equal number of inputs and outputs. Assume these transfer functions have no zeros, and contain only poles[1]. We will denote the poles of the first system as $\alpha_{1,i}\ \forall i \in \{1, p\}$, and those of the second system as $\alpha_{2,j}\ \forall j \in \{1, q\}$. A block diagram of such a system is shown in Figure 8.1.

Given inputs, $u_1 \in \mathbb{R}^m$ and $u_2 \in \mathbb{R}^m$, and outputs, $y_1 \in \mathbb{R}^m$ and $y_2 \in \mathbb{R}^m$, of these systems we can compute their power cepstra, as in Chapter 7[2], $c_{H_i} = c_{Y_i} - c_{U_i}$. From here, it is straightforward to define

$$d_c^2((y_1, u_1), (y_2, u_2)) = \sum_{k=0}^{\infty} k \left(c_{H_1}(k) - c_{H_2}(k)\right)^2. \tag{8.1}$$

We can now try to find the interpretation in terms of model parameters and in terms of subspace angles as in the SISO case.

[1] The extension to systems with (transmission) zeros as well as poles is completely analogous to the SISO case. We use all-pole models here for ease of notation.

[2] We also adopt notation from Chapter 7, and capitalize the subscripts on the power cepstrum, to make the distinction between the MIMO and SISO power cepstra.

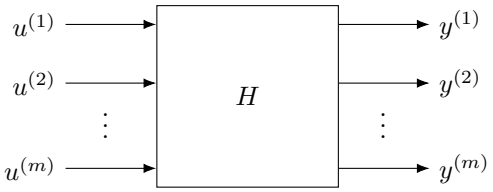


Figure 8.1: A multiple-input multiple-output (MIMO) system, with transfer function matrix H, m inputs and m outputs. The input vector at any point in time consists of m scalar components: $u(k) = (u^{(1)}(k), u^{(2)}(k), \dots, u^{(m)}(k))^\intercal$, and analogously for the output $y(k) = (y^{(1)}(k), y^{(2)}(k), \dots, y^{(m)}(k))^\intercal$.

8.1 Relation to model parameters

Equation (7.15) links these (scalar) cepstral coefficients to the poles of their respective systems. This interpretation is completely analogous to the one for the SISO case (Equation (2.31)). We can now employ the same machinery as in the SISO case and prove the following theorem.

Theorem 17. *Let H_1 and H_2 be two stable, square, all-pole MIMO LTI systems, with output-input pair (y_1, u_1) and (y_1, u_1), respectively, and simple poles $\alpha_{1,i}$, $i \in \{1, \dots, p\}$, and $\alpha_{2,j}$, $j \in \{1, \dots, q\}$. We can then interpret the MIMO weighted cepstral distance between H_1 and H_2 from Equation* (8.1) *in terms of poles as*

$$d_c^2\left((y_1, u_1), (y_2, u_2)\right) = \log \frac{\displaystyle\prod_{i=1}^{p}\prod_{j=1}^{q} \left|1 - \alpha_{1,i}\bar{\alpha}_{2,j}\right|^2}{\displaystyle\prod_{i,j=1}^{p} \left(1 - \alpha_{1,i}\bar{\alpha}_{1,j}\right) \prod_{i,j=1}^{q} \left(1 - \alpha_{2,i}\bar{\alpha}_{2,j}\right)}, \tag{8.2}$$

which is the MIMO extension of Theorem 1.

Proof. The proof closely follows the one for the SISO case in Theorem 1 (Section 2.5).

We have for the MIMO cepstra, from Equation (7.15)

$$c_{H_1}(k) = \sum_{j=1}^{p} \frac{\alpha_{1,j}^{|k|}}{|k|}, \quad c_{H_2}(k) = \sum_{j=1}^{q} \frac{\alpha_{2,j}^{|k|}}{|k|}, \qquad \forall k \neq 0. \tag{8.3}$$

The Theorem then follows,

$$d_c^2\left((y_1,u_1),(y_2,u_2)\right) = \sum_{k=1}^{\infty} k \left(\sum_{j=1}^{p} \frac{\alpha_{1,j}^k}{k} - \sum_{j=1}^{q} \frac{\alpha_{2,j}^k}{k} \right)^2$$

$$= \sum_{j=1}^{p}\sum_{k=1}^{\infty} k \left(\frac{\alpha_{1,j}^k}{k}\right)^2 + \sum_{j=1}^{q}\sum_{k=1}^{\infty} k \left(\frac{\alpha_{2,j}^k}{k}\right)^2 - 2\sum_{j=1}^{p}\sum_{l=1}^{q}\sum_{k=1}^{\infty} k \left(\frac{\alpha_{1,j}^k}{k}\frac{\alpha_{2,l}^k}{k}\right)$$

$$= \log \frac{\prod_{i=1}^{p}\prod_{j=1}^{q} |1-\alpha_{1,i}\bar{\alpha}_{2,j}|^2}{\prod_{i,j=1}^{p}(1-\alpha_{1,i}\bar{\alpha}_{1,j}) \prod_{i,j=1}^{q}(1-\alpha_{2,i}\bar{\alpha}_{2,j})}, \tag{8.4}$$

where in the last equality we used the series expansion from Equation (2.26). □

The MIMO weighted cepstrum can thus be related to a model norm, just as in the SISO case. An extension to pole-zero models is readily available using the same procedure as in 2.6.1.

8.2 Relation to subspace angles

We can try a similar approach for the subspace angles, and try to see whether we can get a result similar to that of Lemma 3 (Subsection 2.7.3).

We start from the expression of the observability matrix of an n-th order MIMO system

$$\Gamma^i = \begin{pmatrix} (C)^\intercal & (CA)^\intercal & (CA^2)^\intercal & \cdots & (CA^{i-1})^\intercal \end{pmatrix}^\intercal, \tag{8.5}$$

with $A \in \mathbb{R}^{n\times n}$ and $C \in \mathbb{R}^{m\times n}$ the matrices from the state space representation of the system (m is the amount of outputs of the system). We assume A to be diagonalizable, and to have simple eigenvalues.

Under a similarity transformation, $\Lambda = TAT^{-1}$, we diagonalize A. The effect of this transformation on C is $C_V = CT^{-1}$. We can now derive

$$\Gamma^i = \begin{pmatrix}(C)^\intercal & (CA)^\intercal & (CA^2)^\intercal & \cdots & (CA^{i-1})^\intercal\end{pmatrix}^\intercal \tag{8.6}$$

$$= \begin{pmatrix}(C_V)^\intercal & (C_V\Lambda)^\intercal & (C_V\Lambda^2)^\intercal & \cdots & (C_V\Lambda^{i-1})^\intercal\end{pmatrix}^\intercal T \tag{8.7}$$

$$= \begin{pmatrix} \mathbb{1} & \mathbb{1} & \cdots & \mathbb{1} \\ \alpha_1\mathbb{1} & \alpha_2\mathbb{1} & \cdots & \alpha_n\mathbb{1} \\ \alpha_1^2\mathbb{1} & \alpha_2^2\mathbb{1} & \cdots & \alpha_n^2\mathbb{1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{i-1}\mathbb{1} & \alpha_2^{i-1}\mathbb{1} & \cdots & \alpha_n^{i-1}\mathbb{1} \end{pmatrix} \begin{pmatrix} c_{V,1} & 0 & \cdots & 0 \\ 0 & c_{V,2} & \cdots & 0 \\ & & & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & c_{V,n} \end{pmatrix} T \tag{8.8}$$

$$= \left(\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{i-1} & \alpha_2^{i-1} & \cdots & \alpha_n^{i-1} \end{pmatrix} \otimes \mathbb{1} \right) \begin{pmatrix} c_{V,1} & 0 & \cdots & 0 \\ 0 & c_{V,2} & \cdots & 0 \\ & & & \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & c_{V,n} \end{pmatrix} T, \tag{8.9}$$

where the α_l, $l \in \{1, \ldots, n\}$, are the poles of the system (which are the eigenvalues of A), $c_{V,l}$ denotes the l-th column of C_V (of dimension $m \times 1$), $\mathbb{1}$ is the $m \times m$ identity matrix, and $\otimes$ is the Kronecker product.

It is this Kronecker product that keeps us from a straightforward extension of the interpretation of the MIMO weighted cepstral distance in terms of subspace angles. The equivalence between the column space of the observability matrix and the column space of the Vandermonde matrix of the poles does not extend easily to the MIMO case, and the interpretation in terms of subspace angles hinged on this result in the SISO case. While the equivalence between the weighted cepstral distance and the interpretation in poles and zeros remains valid, it is, for now, not clear whether and how we can link it to a notion of subspace angles for the MIMO case.

Another open problem for the MIMO case is how to compute the power cepstrum of a system when the number of inputs is not equal to the number of outputs. While we have no analytic answer for this, we can present some results on a real-life application in cross-country electricity exchange, where almost none of the assumptions mentioned in the various Chapters of this thesis hold, and see what remains of the theoretical results in a practical setting.

8.3 International electricity markets

The following application was done in collaboration with the R&D department on *Economie, Fonctionnement et Etudes des Systèmes Energétiques* (EFESE) of Électricité de France (EDF), a global player in the electric utility market, headquartered in Paris.

The international energy markets in Europe are driven by electricity exchanges between countries. While on the economic side, this type of international market appears to be a straightforward endeavor, its implementation is muddied by technical constraints. Cross-country power lines have a maximal transport capacity, and this provides a hard limit on how much electrical power can be transmitted between countries. These issues are entrusted to the Transmission System Operators (TSO) of the different countries.

While the cross-country power lines are important in deriving the feasible exchange capacity, they do not tell the whole story. Indeed, any power line has a thermal constraint[3]. Any basic course on electricity will teach you that electricity does not flow from point A to point B in a grid, but rather permeates the entire network. Consequently, when deciding what the practical exchange capacity between two countries is, it is not enough to take into consideration the capacity of the cross-country power lines. We must take into account the thermal limits of all power lines within the network.

Historically, the TSOs of the different countries solved this problem nationally, and communicate their maximal exchange capacities to each other. Each country established a maximal exchange capacity for each of its borders. For each border, then, two countries calculated a maximal exchange level, and the minimal value of the two was chosen. This would then define a technical solution space, within which an economic equilibrium would be found. This method is known as the Available Transfer Capacity (ATC) mechanism. The exchange capacities thus established, often underestimate the true capacity. The main issue is that border capacities are interdependent. Calculating a maximum level of exchange for a border involves assumptions on the other borders, which were traditionally handled by including safety margins in the final solution.

Often, however, the true economic equilibrium is outside of the technical solution space resulting from the ATC method, resulting in price disparity[4] between the

[3]Transporting electrical power over a line increases its temperature, and this induces an upper bound on transport capacity. Above-ground power lines are a nice example. When their temperature increases, they expand, and start to sag down. When they start to curve down too low, they can induce technical and societal safety issues. This capacity limit will vary with changes in outside temperature and other factors.

[4]Aside from price disparity, there are other negative side effects. One of the more prominent ones is that when one country has an overcapacity of electricity production with low CO_2-

different countries.

Starting in 2015, the Central Western Europe region (CWE), consisting of Austria, Belgium, France, Germany, Luxembourg and the Netherlands, have replaced the ATC mechanism with a different approach, known as the Flow-Based (FB) market coupling mechanism. Here, each TSO defines a set of critical branches (CB) in their national grid, which are significantly impacted by cross-border trade. These are combined into one dataset of CBs for the whole CWE region. Each CB defines a linear constraint, and it is these constraints that define the technical solution space, known as the FB domain[5]. Logically, the FB domain is larger than and contains the ATC-based solution space. This mechanism takes care of the interdependency of all borders, by taking into account the impact on all power lines. As a result, price convergence is higher for the FB market coupling than for the ATC approach [42].

The data on these CBs is made public by the TSOs, and the computational scheme to find the FB domain is carried out by them. This algorithm is non-trivial, and entails a considerable computational effort [55]. An FB domain is constructed for every hour of the day. As such, it is a series of (publicly available) data evolving through time. Tracking certain points of the FB domain allows us to construct time series that give a more or less representative image of the market dynamics.

8.3.1 Data & objective

From a systems theoretical point of view, we can see the whole international electricity market in CWE as a huge dynamical system. Several environmental, business and policy variables, such as - but not limited to - weather conditions, production or outages of certain power plants, and governmental restrictions, serve as an input to the market, which then reacts in a dynamical way, and produces an FB domain as an output, which we can represent by a set of time series, as mentioned in the previous Subsection.

In this application, we see the problem as a system with 8 outputs, y, and 12 inputs, u. The outputs consist of the maximum and minimum exchange

emission (solar, wind, nuclear, ...), congesting the available transfer capacity, its neighboring country still has to fire up a fossil fuel-based plant (gas or coal), to meet national demand. Clean energy goes to waste, while polluting plants are still running.

[5]We are slightly simplifying here, for clarity. In reality, TSOs can introduce other constraints, to ensure safety, reliability, national policy compliance, ... These details are not too important for the present discussion, but a large body of information exists, not only with the TSOs, but also in the scientific literature, for example in [41, 77].

positions for each country (Belgium, France, Germany[6], the Netherlands) of all points in the total FB domain, and the inputs consist of the total renewables (wind onshore, wind offshore, solar) production for each of the four countries.

The time series are sampled 24 times every day, and we consider a period of 21 months, starting in January 2017. This data is windowed, with a window length of 14 days, and a new window starting every day. We will perform clustering on these windows.

The objective is now to get, for a given window of data, the subset of historical data that has similar dynamics. More specifically, we want to compare the current period (i.e., the data of the past 14 days) to the historical data set, and find the most relevant windows.[7]

To do so, we cluster the historical dataset, by calculating the MIMO power cepstrum, c_Y of outputs, y_i, and c_U of inputs, u_i, for window i and calculating weighted cepstral windows,

$$w_i(k) = \sqrt{k}(c_{Y_i}(k) - c_{U_i}(k)). \tag{8.10}$$

These are the weighted windows we use in a cepstral k-means algorithm, as proposed in Chapter 5.

Note that this application violates several of the technical assumptions that were made throughout this thesis. We have no guarantee the signals are persistently exciting, that they represent linear and (at least within a given window) time invariant, stable and inversely stable dynamics, there is a mismatch between the amount of inputs and the amount of outputs, ...

While it may seem that this fundamentally limits us to do anything with the methods proposed in this thesis, it is important to note that in practical applications, this is more often than not the case: real data is messier than one would like, most of the time. Indeed, we include this application exactly for this reason: even though most of the assumptions we made are not met, we will show that results are still sensible, showing the robustness of the developed machinery.

[6]Germany, Austria and Luxembourg constituted one unified "national" market, at the time considered in this example. Recently, they have moved to separate into smaller bidding zones.

[7]These windows will then be used by EDF to do predictive modeling on. As the variability in dynamics within a cluster is smaller than that in the whole dataset, we expect these cluster-specific models to have better predictive power, and this indeed turns out to be the case.

8.3.2 Clustering results

The results of the cepstral k-means clustering approach are shown in Figure 8.2. While this is a completely unsupervised and explorative algorithm, we can still give some clear interpretation to these results. Specifically, we immediately notice the seasonality we would expect from the electricity market: dynamics will vary throughout seasons, and summer will witness fundamentally different dynamics than winter.

Clusters 0, 1, 2, and (less clearly) 10 seem to occur mainly in summer, while cluster 6,7, and 9 fall mainly in winter. Since the algorithm did not get any information on which time of the year the window fell into, this gives strong credibility to the claim that the weighted cepstral distance is a sensible distance measure in this context.

It seems the algorithm indeed discovers these different modes of operation of the market. Within a season, we see multiple clusters, corresponding to the fact that within a season, there may be different modes of operation. The middle of the winter, for example, not only represents a much colder period than the start or end, but also includes Christmas holidays, which will have an impact on electricity demand.

The international electricity market is continuously changing, not only because of changes in the energy mix of countries, but also due to volatile demand, policy changes, and changes in the physical network infrastructure. It is not always easy and straightforward to keep track of these types of changes in dynamics. The exact network topology, for example, is confidential information owned by the TSOs, and not shared with other actors in the market. These changes in dynamics might be what we see in cluster 8: at the beginning of the observed time frame, this behavior does not occur, yet throughout time, it becomes more and more prevalent. Further testing should be done, but the algorithm developed here should allow to detect such large changes in market dynamics, rather than in the shapes of the output windows.

One could counter these results by posing that maybe the problem at hand would allow for easily interpretable clustering results in any case, and any clustering algorithm will be able to find different dynamics. In the following Subsection, we therefore compare with some other approaches, and show the merit of using the deterministic weighted cepstral distance.

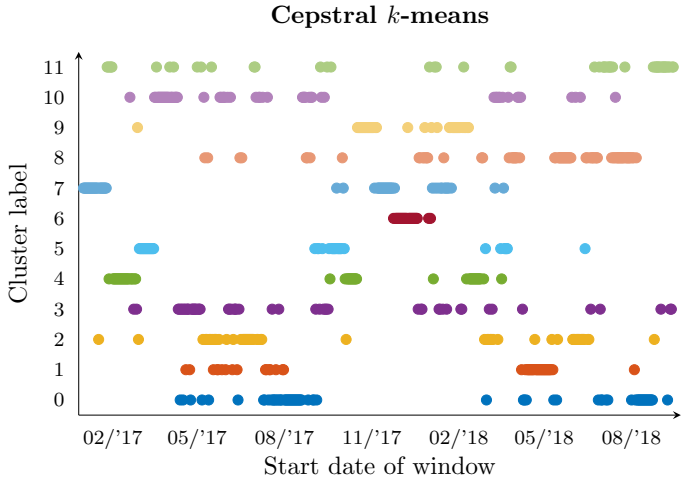


Figure 8.2: The results of clustering input and output time series coming from the international electricity markets, using k-means clustering on the weighted cepstral windows from Equation (8.10). For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. This is the straightforward extension of the deterministic weighted cepstral distance to the rectangular MIMO case. The step between windows is one day. Note that we clearly see seasonal patterns in the results, with clusters 0, 1 and 2 clearly belonging to summer, and clusters 6, 7 and 9 to winter.

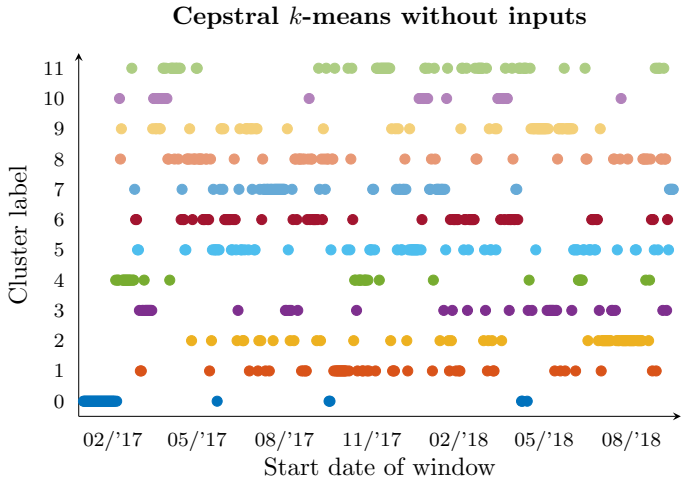


Figure 8.3: The results of clustering input and output time series coming from the international electricity markets, using k-means clustering on the weighted cepstral windows from Equation (8.10), with the $u_i(k)$ omitted. For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. While some structure is still seen in the results, it is less straightforward to interpret the different clusters.

8.3.3 Comparison

Stochastic weighted cepstral distance We start by looking at the relative merit of subtracting the cepstra of the inputs from the output cepstra. Indeed, one could argue that the dynamics of the output series representing the Flow-Based domain contain the relevant information, and we are interested in all of the dynamics. However, there are two important remarks to be made.

First, the input dynamics consist of very complex behavior, as the renewables production is governed by climatological factors. We would thus expect these dynamics to be very strongly present, and maybe even dominate, in the output series.

Secondly, these are not the dynamics we are interested in: we usually have available quite good day-ahead predictions of weather factors, policy is not likely to change overnight without any prior warning, and some educated estimates of production unit availability[8] and demand for tomorrow can be made as well. What we are interested in, is how the market will react to these inputs. In system theoretical terms, we are interested in the transfer function of the inputs to the outputs. We want to be able to compare the FB domain signals in different periods, even when climatological factors are completely different.

Figure 8.3 shows the results when we omit the term $c_{U_i}(k)$ in Equation (8.10). Note that the seasonality is much less clear in these clusters, and they seem to be chosen much more at random than in the deterministic case. This is to be expected. Most probably, the dynamics of what type of non-base load production (i.e., solar, wind onshore, wind offshore) is operational, dominates - which correlates with climatological factors.

This is again an example that, in the deterministic case, the model power cepstrum really is a necessary extension of the stochastic power cepstrum, even in the MIMO case, and apparently even when there are unequal amounts of inputs and outputs!

Traditional k-means We compare the cepstral k-means algorithm to traditional k-means. We concatenate the different dimensions of the outputs, so for the multivariate output signal $y(k)$, with components $y_i(k)$, $i \in \{1, 2, \ldots, 8\}$, we thus construct the vector

$$(y_1(0), \ldots, y_1(k), y_2(0), \ldots, y_2(k), \ldots, y_8(0), \ldots, y_8(k)). \tag{8.11}$$

[8] Part of the production is operated by EDF itself, and they therefore have some reliable information on this subject; if they know they will have to temporarily close down a large plant for maintenance tomorrow, they can factor this into their predictions.

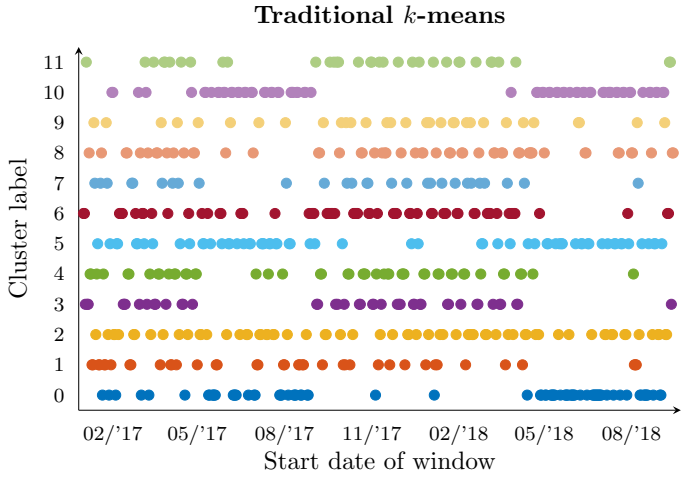


Figure 8.4: The results of clustering output time series coming from the international electricity markets, using k-means clustering on the concatenated output signals in Equation (8.11). For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. These results do not inspire a lot of confidence from a dynamical point of view. Consecutive windows have a lot of overlap, and thus exhibit similar dynamics. However, we see that consecutive windows are often in different clusters, when using this approach.

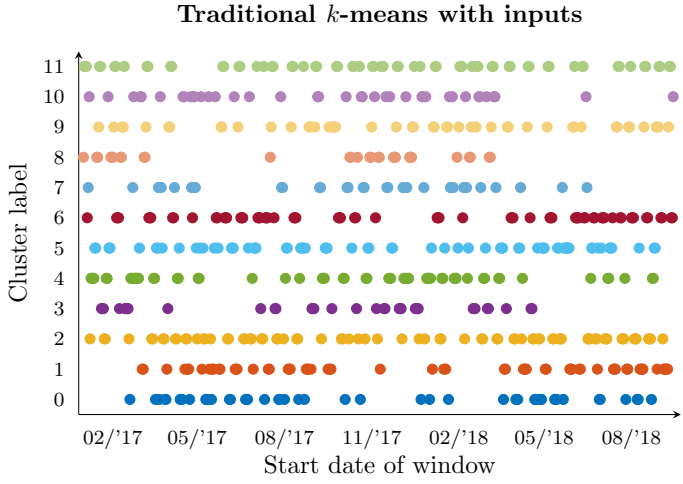


Figure 8.5: The results of clustering input and output time series coming from the international electricity markets, using k-means clustering on the concatenated output and input signals. For each 14-day window, the cluster label is plotted. Colors and the value on the y-axis are redundant. No clear improvement over Figure 8.4 is seen.

This is done for each window. We then run the k-means algorithm on these vectors, which uses the Euclidean distance behind the scenes.

Results are shown in Figure 8.4. The seasonality, while still present, is much less pronounced in these clusters. Moreover, where the cepstral k-means approach mainly resulted in blocks of consecutive windows belonging to the same cluster, the behavior here is much more erratic.

Indeed, very often, windows are in a cluster, without the previous and consecutive window being in that same cluster as well. Note, however, that consecutive windows have a large amount of overlap. In fact, there is only one day of non-overlapping data between windows. From a dynamical point of view, a window being present in a cluster, without at least either the previous or consecutive window being present as well is therefore a suspicious result. While this happens sporadically in the cepstral k-means approach as well, it seems that, in the case of traditional k-means (with the Euclidean distance), most clusters exhibit this behavior. This does not improve the credibility[9] of this approach in our context.

There is no straightforward way to extend the traditional k-means analysis to take the input data into account, other than simply concatenating the inputs to the outputs, resulting in the vector

$$(y_1(0), \ldots, y_1(k), \ldots, y_8(0), \ldots, y_8(k), u_1(0), \ldots, u_1(k), \ldots, u_{12}(0), \ldots, u_{12}(k)).$$

Results of this procedure are shown in Figure 8.5. No clear difference with the rest of the analysis made here is seen.

[9]Let us stress, once more, this is not to say that the Euclidean distance measure is a bad measure. We only mean to say that what the Euclidean distance is measuring is not what we are looking for, when trying to discern similar dynamics in the historical data set.

Chapter 9

Conclusion

The main objective of this thesis was to develop an interpretable, usable and computable formal framework for time series clustering. The different subgoals to achieve this were summarized in the scientific programme in Figure 1.3.

As we set out to do in Chapter 1, we presented results on the weighted power cepstral distance, and its use in time series clustering, as it was defined for autonomous and stochastic sytems [54, 19].

The chapters of this thesis consisted of two papers and new material, previously unpublished. In Chapter 3, we proposed an extension of the distance measure to input/output-signal pairs coming from deterministic systems, and corroborated our results with synthetic data coming from a simple electric circuit. We proceeded in Chapter 4 by interpreting this extended distance measure in terms of subspace angles, and poles and zeros. We tested sensitivity of the results on certain algorithmic and application-dependent parameters.

Chapter 5 developed a clustering algorithm, cepstral k-means, and an interpretation for the resulting clusters in terms of system theoretical notions. We showed this approach solves the open problem of time series subsequence clustering. We illustrated the power of this approach in an unsupervised fault detection algorithm for wind turbine generators.

A system identification technique for autoregressive systems was presented in Chapter 6, estimating the difference equation describing a system, starting from its power cepstrum. Several illustrations were worked out, including the classic example of Yule's sunspot data, and an application in structural health monitoring. This allows explicit representations of cluster centers coming from

a cepstral clustering procedure: the models estimated from the cluster centers are low-biased estimators of the average dynamics.

The notion of the power cepstrum was extended to the case of MIMO systems in Chapter 7, and an algorithm to compute it was developed for square systems. A case study on a continuous stirred tank reactor was analyzed.

A system identification technique for autoregressive systems was presented in Chapter 6, estimating the difference equation describing a system, starting from its power cepstrum. Several illustrations were worked out, including the classic example of Yule's sunspot data, and an application in structural health monitoring. This allows explicit representations of cluster centers coming from a cepstral clustering procedure: the models estimated from the cluster centers are low-biased estimators of the average dynamics.

Implementations in Python and/or Matlab of the different methods and applications were provided.

We have developed the rigorous, interpretable, usable and computable time series clustering framework we set out to find in Chapter 1. We hope to have convinced the reader that the weighted cepstral distance measure is an appropriate, versatile and robust measure in many applications, and especially in those that are traditionally seen as engineering applications.

The many numerical illustrations and applications on both synthetic and real-life data sets were worked out to convey the importance of the choice of similarity measure, not only in time series analysis, but in machine learning in general, and the tremendous impact it can have on results in practice. More specifically, we have exposed the importance of understanding how a clustering algorithm quantifies similarity, and whether it is a useful measure in the context at hand or not.

We believe firmly that this approach has the ability to greatly further time series research, and will lead to deeper insights, more powerful algorithms, and impactful applications.

9.1 Impact on other time series analysis applications

While we have focused on unsupervised clustering applications, the weighted cepstral distance could also improve results for other types of time series analysis.

An immediate application could be time series classification. In fact, our

research has had direct impact in this regard: based on the results from the paper in Chapter 3, [27] presents a time series classification framework, with good performance on the problem of detecting mobile phone use while driving. They, too, note the usability (eliminating the subjective feature selection step) and computability (the methods are computationally inexpensive) of these approaches.

Developments like this underline the need in the broader field of time series analysis for the rigorous framework like the one we developed for clustering. It is encouraging to see our methods being used to meet this need. In turn, we can use results like this to reflect on our own research.

For example, [27] offers some methods to automatically tune parameters, like the cutoff parameter discussed in Chapter 4. While being able to tune such parameters based on the known labels is an advantage of the supervised method developed there, the insights they generate could carry over to the unsupervised clustering framework developed here, and it would be worthwhile to investigate this further. The application tackled in the paper deals with multiple signals. One way the classifier method could be improved, is by utilizing the MIMO cepstrum, as it was dicussed in Chapters 7 and 8.

The connection to the positive definite subspace angles kernel in Chapter 4 allows the distance measure to also be used in kernel-based methods, such as support vector machines.

We did not look into these types of applications in detail in this thesis, but for the purposes of anomaly detection, classification, recommendation systems, ..., the distance measure presented here could prove useful, and its applicability is certainly not constricted only to clustering applications.

The importance of similarity measures in machine learning is being increasingly recognized by the academic community, giving rise, for example, to projects like SIMBAD [60], focusing on machine learning based on (dis)similarity measures.

9.2 Future work

Even though we hope to have given a coherent and consistent account of our work on weighted cepstral techniques, academic research is never complete. We therefore present some avenues for future work.

Model norm The weighted cepstral distance is firmly embedded in systems theory, with its connection to a model norm. This is what allowed us to express,

for example, cluster centers in terms of average models, and cluster variance in terms of the error models. Since this notion is so central to our results, it would of course be interesting to study them even more in-depth from the theoretical side.

Especially the connection with Dirichlet spaces and norms [3], briefly touched upon in Subsection 2.5.2, could provide a promising avenue to extend the weighted cepstral norm to more general cases. Even non-linear systems could fit in this framework. In any case, mapping insights from one field to the other could highly improve, generalize and extend the techniques developed here.

An immediate improvement would be to use the insights from Dirichlet space functional theory on the weight of the zero-th cepstrum coefficient, to turn the model semi-norm into a proper norm, and its system theoretical and practical implications in real-life challenges.

Multidimensional systems For now, the whole analysis has been done for systems in one dimension: time. However, an extension to multidimensional systems, which also consider spatial dynamics, should be possible.

This requires an extension of the notion of power cepstrum to this type of systems, like we proposed for the MIMO cepstrum in Chapter 7. In the course of this PhD, we have obtained some very partial preliminary results towards this goal, in the context of a master's thesis [58], but no completely satisfactory extension was found.

The definition of the MIMO power cepstrum hinged crucially on the Smith-McMillan form of the transfer function matrix. No straightforward extension of this notion exists for multidimensional systems. However, [5] introduces a weaker version in the context of model matching and interpolation problems, and could provide a starting point for developing a multidimensional power cepstrum.

MIMO systems While we have made some progress in extending our formal framework to MIMO systems in Chapters 7 and 8, the full interpretation (for example in terms of subspace angles) remains a missing link (as mentioned in the scientific programme we set out in Figure 1.3).

Fully fleshing out the weighted cepstral distance, both in terms of the subspace angles, and computationally for rectangular systems, remains an open problem. The collaboration with EDF, on pattern recognition in the international energy market, on the other hand, shows that such a formal extension of the framework could result in meaningful results for practical applications.

Time series subsequence clustering applications Having opened up the applicational field of time series subsequence clustering, it would be very interesting to try out the developed techniques on real-life challenges.

These applications could be in many different areas, such as anomaly detection in machinery, as explored for wind turbine generators in Subsection 5.3. Another promising type of application is event detection in biomedical data, such as ECG or EEG signals. In combination with the extension to multidimensional systems, the formal framework could even be applied to pattern recognition for video data.

Cepstral identification of more general model classes At present, the cepstral system identification technique presented in Chapter 6 was defined only for all-pole models. The power cepstral coefficients, however, also contain information on the zeros of the underlying dynamics.

Since these zeros enter with a minus sign in Equation (2.31), whereas the poles enter with a plus, there is some hope we could separate them, and extend this extremely efficient identification method to pole-zero models.

The formal framework developed here is a first building block in a broader programme of interpretable time series analysis. We have fully leveraged theoretical insights from established and mathematically rigorous field like systems theory in developing practically applicable algorithms. The paths for future research set out above give some guidelines on where to go from here.

As time series become more and more ubiquitous, and the importance of similarity measures is increasingly recognized, we believe this work is a meaningful contribution to the different fields involved in (time series) data analysis. While this text does not claim to hold all the answers, it does propose a formal way to think about these problems, and maximally fuses theoretical knowledge from rich, traditional scientific fields, such as systems theory and signal processing, with the more voluntaristic, flexible and black-box methods of modern machine learning techniques.

Above all, we hope to have conveyed the message that combining this theoretical knowledge of generations of mathematicians, scientist and engineers with the ever-increasing computational insights and capabilities of modern technology, will generate fundamental knowledge on applied techniques, improving understanding of the user and performance of practical results. Exploiting these links will be paramount in furthering both academic and applied advances in data analysis and machine learning.

Bibliography

[1] Afsari, B., Chaudhry, R., Ravichandran, A., and Vidal, R. Group action induced distances for averaging and clustering linear dynamical systems with applications to the analysis of dynamic scenes. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012), IEEE, pp. 2208–2215.

[2] Akaike, H. Stochastic theory of minimal realization. *IEEE Transactions on Automatic Control 19*, 6 (1974), 667–674.

[3] Arcozzi, N., Rochberg, R., Sawyer, E. T., and Wick, B. D. The Dirichlet space: A survey. *New York J. Math. A 17* (2011), 45–86.

[4] Attivissimo, F., Savino, M., and Trotta, A. A study on nonlinear averagings to perform the characterization of power spectral density estimation algorithms. *IEEE Transactions on Instrumentation and Measurement 49*, 5 (2000), 1036–1042.

[5] Ball, J. A., and Malakorn, T. Multidimensional linear feedback control systems and interpolation problems for multivariable holomorphic functions. *Multidimensional Systems and Signal Processing 15*, 1 (2004), 7–36.

[6] Blum-Smith, B., and Coskey, S. The fundamental theorem on symmetric polynomials: History's first whiff of Galois theory. *The College Mathematics Journal 48*, 1 (2017), 18–29.

[7] Boets, J., De Cock, K., De Moor, B., Espinoza, M., et al. Clustering time series, subspace identification and cepstral distances. *Communications in Information & Systems 5*, 1 (2005), 69–96.

[8] Bogert, B. P., Healy, M. J., and Tukey, J. W. The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. In *Proceedings of the Symposium on Time Series Analysis* (1963), vol. 15, pp. 209–243.

[9] Boklan, K. D. A note on identities for elementary symmetric and power sum polynomials. *Discrete Mathematics, Algorithms and Applications 10*, 01 (2018), 1850004.

[10] Bornn, L., Farrar, C. R., Park, G., and Farinholt, K. Structural health monitoring with autoregressive support vector machines. *Journal of Vibration and Acoustics 131*, 2 (2009), 021004:1–9.

[11] Brigham, E. O. *The Fast Fourier Transform and Its Applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[12] Chandorkar, M., Mall, R., Lauwers, O., Suykens, J. A., and De Moor, B. Fixed-size least squares support vector machines: Scala implementation for large scale classification. In *2015 IEEE Symposium Series on Computational Intelligence* (2015), IEEE, pp. 522–528.

[13] Cheever, E. Linear physical systems analysis. Swarthmore College Department of Engineering, http://lpsa.swarthmore.edu/, 04/22/2016.

[14] Chen, F., Deng, P., Wan, J., Zhang, D., Vasilakos, A. V., and Rong, X. Data mining for the Internet of Things: literature review and challenges. *International Journal of Distributed Sensor Networks 501* (2015), 431047.

[15] Chen, J. R. Making subsequence time series clustering meaningful. In *Fifth IEEE International Conference on Data Mining (ICDM'05)* (2005), IEEE, pp. 8–pp.

[16] Clette, F., Svalgaard, L., Vaquero, J. M., and Cliver, E. W. Revisiting the sunspot number. *Space Science Reviews 186*, 1-4 (2014), 35–103.

[17] De Cock, K. Principal Angles in System Theory, Information Theory and Signal Processing, 2002. *PhD thesis*, 293 pp, KU Leuven , May 2002.

[18] De Cock, K., and De Moor, B. Subspace angles and distances between ARMA models. In *Proc. of the Intl. Symp. of Math. Theory of Networks and Systems* (2000), vol. 1.

[19] De Cock, K., and De Moor, B. Subspace angles between ARMA models. *Systems & Control Letters 46*, 4 (2002), 265–270.

[20] EDF. Hack the wind 2018: Wind turbines failures detection. https://opendata.edp.com/pages/hackthewind/.

[21] Elkan, C. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)* (2003), pp. 147–153.

[22] Fakhrazari, A., and Vakilzadian, H. A survey on time series data mining. In *2017 IEEE International Conference on Electro Information Technology (EIT)* (2017), IEEE, pp. 476–481.

[23] Figueiredo, E., Figueiras, J., Park, G., Farrar, C. R., and Worden, K. Influence of the autoregressive model order on damage detection. *Computer-Aided Civil and Infrastructure Engineering 26*, 3 (2011), 225–238.

[24] Fugate, M. L., Sohn, H., and Farrar, C. R. Vibration-based damage detection using statistical process control. *Mechanical Systems and Signal Processing 15*, 4 (2001), 707–721.

[25] Fujimaki, R., Hirose, S., and Nakata, T. Theoretical analysis of subsequence time-series clustering from a frequency-analysis viewpoint. In *Proceedings of the 2008 SIAM International Conference on Data Mining* (2008), SIAM, pp. 506–517.

[26] Fulcher, B. D., and Jones, N. S. Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering 26*, 12 (2014), 3026–3037.

[27] Gelmini, S., Formentin, S., Strada, S., Tanelli, M., and Savaresi, S. fierClass: A multi-signal, cepstrum-based, time series classifier. *Engineering Applications of Artificial Intelligence 87* (2020), 103262.

[28] Golub, G. H., and Van Loan, C. F. *Matrix Computations*, fourth ed. JHU press, 2012.

[29] Hofmann, T., Schölkopf, B., and Smola, A. J. Kernel methods in machine learning. *The annals of statistics* (2008), 1171–1220.

[30] Hotelling, H. Relations between two sets of variates. *Biometrika 28*, 3/4 (1936), 321–377.

[31] Huang, X., Acero, A., and Hon, H.-W. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice hall PTR, 2001.

[32] Hubert, L., and Arabie, P. Comparing partitions. *Journal of Classification 2*, 1 (1985), 193–218.

[33] Jordan, C. Essai sur la géométrie à n dimensions. *Bulletin de la Société Mathématique de France 3* (1875), 103–174.

[34] Kailath, T. *Linear Systems*. Prentice-Hall Englewood Cliffs, NJ, 1980.

[35] Kaufman, L., and Rousseeuw, P. Clustering by means of medoids. In *Statistical Data Analysis Based on the L1-Norm and Related Methods* (1987), North-Holland, pp. 405–416.

[36] Keogh, E. Exact indexing of dynamic time warping. In *Proceedings of the 28th International Conference on Very Large Data Bases* (2002), VLDB Endowment, pp. 406–417.

[37] Keogh, E., and Lin, J. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and Information Systems 8*, 2 (2005), 154–177.

[38] Kim, K., Parthasarathy, G., Uluyol, O., Foslien, W., Sheng, S., and Fleming, P. Use of scada data for failure detection in wind turbines. In *ASME 2011 5th International Conference on Energy Sustainability* (2011), American Society of Mechanical Engineers Digital Collection, pp. 2071–2079.

[39] Knill, O. Cauchy–Binet for pseudo-determinants. *Linear Algebra and its Applications 459* (2014), 522–547.

[40] Kooistra, J. Newzoo's 2019 global mobile market report. *Newzoo, September* (2019).

[41] Kristiansen, T. The flow based market coupling arrangement in Europe: Implications for traders. *Energy Strategy Reviews 27* (2020), 100444.

[42] KU Leuven Energy Institute. Cross-border electricity trading: Towards flow-based market coupling. February 2015, `https://set.kuleuven.be/ei/factsheets`.

[43] Kwac, J., Flora, J., and Rajagopal, R. Household energy consumption segmentation using hourly data. *IEEE Transactions on Smart Grid 5*, 1 (2014), 420–430.

[44] Lauwers, O., Agudelo, O. M., and De Moor, B. A multiple-input multiple-output cepstrum. *IEEE Control Systems Letters 2*, 2 (2018), 272–277.

[45] Lauwers, O., and De Moor, B. A time series distance measure for efficient clustering of input/output signals by their underlying dynamics. *IEEE Control Systems Letters 1* (June 2017), 286–291.

[46] Li, G., Qin, S. J., Ji, Y., and Zhou, D. Reconstruction based fault prognosis for continuous processes. *Control Engineering Practice 18*, 10 (2010), 1211 – 1219.

[47] Liao, T. W. Clustering of time series data – a survey. *Pattern Recognition 38*, 11 (2005), 1857–1874.

[48] Liu, H., Yu, L., Wang, W., and Sun, F. Extreme learning machine for time sequence classification. *Neurocomputing 174* (2016), 322–330.

[49] Ljung, L. *System Identification: Theory for the User*. Prentice Hall information and system sciences series. Prentice Hall PTR, 1999.

[50] Lloyd, S. Least squares quantization in pcm. *IEEE Transactions on Information Theory 28*, 2 (1982), 129–137.

[51] MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Sympos. Math. Statist. and Probability (Berkeley, Calif., 1965/66)*. Univ. California Press, Berkeley, Calif., 1967, pp. Vol. I: Statistics, pp. 281–297.

[52] Manzato, S., White, J. R., LeBlanc, B., Peeters, B., and Janssens, K. Advanced identification techniques for operational wind turbine data. In *Topics in Modal Analysis, Volume 7*. Springer, 2014, pp. 195–209.

[53] Martí, L., Sanchez-Pi, N., Molina, J. M., and Garcia, A. C. B. Anomaly detection based on sensor data in petroleum industry applications. *Sensors 15*, 2 (2015), 2774–2797.

[54] Martin, R. J. A metric for ARMA processes. *IEEE Transactions on Signal Processing 48*, 4 (2000), 1164–1170.

[55] Matthes, B., Spieker, C., and Rehtanz, C. Flow-based parameter determination in large-scale electric power transmission systems. In *2017 IEEE Manchester PowerTech* (2017), IEEE, pp. 1–6.

[56] Mercer, J. Xvi. Functions of positive and negative type, and their connection the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character 209* (1909), 415–446.

[57] Mihoubi, M. Bell polynomials and binomial type sequences. *Discrete Mathematics 308*, 12 (2008), 2450–2459.

[58] Moons, M. Multidimensionele uitbreiding van een cepstrale afstandsmaat voor clustering van ingangs-/uitgangssystemen. *Master thesis*, KU Leuven, 2018.

[59] Oppenheim, A., and Schafer, R. *Digital Signal Processing (First edition)*. Prentice Hall International Editions. Prentice-Hall International, 1975.

[60] Pelillo, M. Introduction: The SIMBAD Project. In *Similarity-Based Pattern Analysis and Recognition*. Springer, 2013, pp. 1–10.

[61] Percival, D., and Walden, A. *Spectral Analysis for Physical Applications*. Cambridge University Press, 1993.

[62] Pereira, C. M. M., and de Mello, R. F. Common dissimilarity measures are inappropriate for time series clustering. *Revista de Informática Teórica e Aplicada 20*, 1 (2013), 25–48.

[63] Pintelon, R., Schoukens, J., and Renneboog, J. The geometric mean of power (amplitude) spectra has a much smaller bias than the classical arithmetic (RMS) averaging. *IEEE Transactions on Instrumentation and Measurement 37*, 2 (1988), 213–218.

[64] Rand, W. M. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association 66*, 336 (1971), 846–850.

[65] Randall, R., and Smith, W. New cepstral methods for the diagnosis of gear and bearing faults under variable speed conditions. In *ICSV23 Conference, Athens* (2016).

[66] Randall, R. B. A history of cepstrum analysis and its application to mechanical problems. *Mechanical Systems and Signal Processing 97* (2017), 3–19.

[67] Ratanamahatana, C. A., and Keogh, E. Everything you know about dynamic time warping is wrong. In *Third Workshop on Mining Temporal and Sequential Data* (2004).

[68] Ridi, A., Gisler, C., and Hennebert, J. Automatic identification of electrical appliances using smart plugs. In *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)* (2013), IEEE, pp. 301–305.

[69] Saito, N., and Coifman, R. R. *Local Feature Extraction and its Applications Using a Library of Bases*. PhD thesis, Yale University, 1994.

[70] Schoukens, J., and Renneboog, J. Modeling the noise influence on the Fourier coefficients after a discrete Fourier transform. *IEEE Transactions on Instrumentation and Measurement*, 3 (1986), 278–286.

[71] Schroeder, M. Direct (nonrecursive) relations between cepstrum and predictor coefficients. *IEEE Transactions on Acoustics, Speech, and Signal Processing 29*, 2 (1981), 297–301.

[72] Sohn, H., Czarnecki, J. A., and Farrar, C. R. Structural health monitoring using statistical process control. *Journal of Structural Engineering 126*, 11 (2000), 1356–1363.

[73] Suykens, J. A., Van Gestel, T., De Brabanter, J., De Moor, B., and Vandewalle, J. *Least Squares Support Vector Machines*. World scientific, 2002.

[74] TrendMiner. https://www.trendminer.com/.

[75] Uddin, Z., Ahmad, A., Qamar, A., and Altaf, M. Recent advances of the signal processing techniques in future smart grids. *Human-centric Computing and Information Sciences 8*, 1 (2018), 2.

[76] Vafeiadis, T., Krinidis, S., Ziogou, C., Ioannidis, D., Voutetakis, S., and Tzovaras, D. Robust malfunction diagnosis in process industry time series. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)* (July 2016), pp. 111–116.

[77] Van den Bergh, K., Boury, J., and Delarue, E. The flow-based market coupling in Central Western Europe: Concepts and definitions. *The Electricity Journal 29*, 1 (2016), 24–29.

[78] Van Overschee, P., and De Moor, B. *Subspace identification for linear systems: Theory - Implementation - Applications*. Springer Science & Business Media, 1996.

[79] Verleysen, M., and François, D. The curse of dimensionality in data mining and time series prediction. In *Computational Intelligence and Bioinspired Systems* (2005), Springer, pp. 758–770.

[80] Vishwanathan, S., and Smola, A. J. Binet-Cauchy kernels. In *Proceedings of the 17th International Conference on Neural Information Processing Systems* (2004), MIT Press, pp. 1441–1448.

[81] Vishwanathan, S., Smola, A. J., and Vidal, R. Binet-Cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes. *International Journal of Computer Vision 73*, 1 (2007), 95–119.

[82] Wang, K., Xu, C., Zhang, Y., Guo, S., and Zomaya, A. Y. Robust big data analytics for electricity price forecasting in the smart grid. *IEEE Transactions on Big Data 5*, 1 (2017), 34–45.

[83] Wang, W., Liu, H., Yu, L., and Sun, F. Human activity recognition using smart phone embedded sensors: A linear dynamical systems method. In *Neural Networks (IJCNN), 2014 International Joint Conference on* (2014), IEEE, pp. 1185–1190.

[84] Welch, P. The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics 15*, 2 (1967), 70–73.

[85] Werner, R. Sunspot number prediction by an autoregressive model. *Sun and Geosphere 7* (2012), 75–80.

[86] Willems, J. C. From time series to linear system – Part I. Finite dimensional linear time invariant systems. *Automatica 22*, 5 (1986), 561–580.

[87] Willems, J. C., Rapisarda, P., Markovsky, I., and De Moor, B. L. A note on persistency of excitation. *Systems & Control Letters 54*, 4 (2005), 325–329.

[88] Wolf, L., and Shashua, A. Learning over sets using kernel principal angles. *Journal of Machine Learning Research 4*, Oct (2003), 913–931.

[89] Yperman, J., Becker, T., Valkenborg, D., Popescu, V., Hellings, N., Van Wijmeersch, B., and Peeters, L. Machine learning analysis of motor evoked potential time series to predict disability progression in multiple sclerosis. *BioRxiv* (2019), 772996.

[90] Yule, G. U. On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers. *Philosophical Transactions of the Royal Society of London Series A 226* (1927), 267–298.

[91] Zolhavarieh, S., Aghabozorgi, S., and Teh, Y. W. A review of subsequence time series clustering. *The Scientific World Journal* (2014).

List of publications

[12] Chandorkar, M., Mall, R., Lauwers, O., Suykens, J. A., and De Moor, B. Fixed-size least squares support vector machines: Scala implementation for large scale classification. In *2015 IEEE Symposium Series on Computational Intelligence* (2015), IEEE, pp. 522–528

[45] Lauwers, O., and De Moor, B. A time series distance measure for efficient clustering of input/output signals by their underlying dynamics. *IEEE Control Systems Letters 1* (June 2017), 286–291

[44] Lauwers, O., Agudelo, O. M., and De Moor, B. A multiple-input multiple-output cepstrum. *IEEE Control Systems Letters 2*, 2 (2018), 272–277

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
STADIUS
Kasteelpark Arenberg 10 box 2446
B-3001 Leuven
https://www.esat.kuleuven.be/stadius/