KU LEUVEN

VUB VRIJE UNIVERSITEIT BRUSSEL

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Technology

From Volterra Kernels to State-Space: A Path to Structured Nonlinear Block-Oriented Modelling

Prabhu Vijayan

Dissertation presented in partial fulfillment
of the requirements for the degree of
Doctor of Engineering Technology (PhD)

November 2025

Supervisors:
Prof. dr. Mariya Ishteva
Prof. dr. ir. John Lataire
(Vrije Universiteit Brussel)
Dr. ir. Philippe Dreesen
(Maastricht University)

FROM VOLTERRA KERNELS TO STATE-SPACE: A PATH TO STRUCTURED NONLINEAR BLOCK-ORIENTED MODELLING

Prabhu VIJAYAN
Supervisors:
Prof. dr. Mariya Ishteva
Prof. dr. ir. John Lataire
(Vrije Universiteit Brussel)
Dr. ir. Philippe Dreesen
(Maastricht University)

Members of the
Examination Committee:
Prof. dr. ir. Han Haitjema, chair
Prof. dr. ir. Dimitrios Aggelis, chair
(Vrije Universiteit Brussel)
Prof. dr. ing. Peter Karsmakers
Prof. dr. ir. Bart Vanrumste
Dr. ir. Diana Sima
(icometrix)
Dr. Konstantin Usevich
(CNRS, Université de Lorraine)
Dr. ir. Koen Tiels
(Eindhoven University of Technology)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Technology (PhD)

November 2025

© 2025 Prabhu Vijayan
Uitgegeven in eigen beheer, Prabhu Vijayan, Kleinhoefstraat 4, B-2440 Geel (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.

Acknowledgements

If someone told me that a PhD was equal parts science, stubbornness and self-discovery, I would have laughed back then. Now I know better. Between MATLAB meltdowns and small wins that felt like miracles, this journey shaped me far beyond academia.

My deepest gratitude goes to my supervisors - Prof. Mariya Ishteva, Prof. John Lataire, and Dr. Philippe Dreesen. Your rock-solid support has been the foundation of this entire journey. Thank you for being remarkably accessible and generous with your time - from quick clarifications to those full-day discussions that turned abstract ideas into concrete results. Your patience, insight, and unwavering encouragement gave me the confidence to explore freely, even when the path was uncertain. I am deeply grateful for the balance you offered between freedom and guidance, trust and direction. This thesis, and the person I have become through it, owes much to your mentorship.

I sincerely thank the jury members for their time, expertise, and guidance. Some of you served as assessors throughout my PhD, and your careful reading and feedback on my thesis greatly improved its clarity, depth, and rigour.

I gratefully acknowledge the financial support received from the Research Foundation - Flanders (FWO) through projects G090117N and G033822N, the FNRS–FWO EOS Project 30468160, and the KU Leuven start-up grant STG/19/036.

To my parents and my sister, my constant source of comfort and strength, thank you. You gave me the freedom to experiment, fail, learn, and try again, without judgment. Even when my choices made things uncertain, you stood by me with quiet trust. You are my safe place, my calm amid the noise, and my reminder of what truly matters.

Geel, my charming little Belgian town, the quiet backdrop to some of the most beautiful years of my life, you have been so much more than a dot on the map. It was here in Belgium that I met Lotan, my husband, who was also pursuing his PhD at KU Leuven. Two people from different corners of the world, far from home, found each other here, and later sealed that story in Geel. To him, my rock and partner in every sense, and to his wonderful family and my in-laws, thank you for your endless love, warmth, and

unwavering support across the distance. Belgium brought us together, and for that, it will always hold a special place in my heart.

To my friends who turned research breaks into therapy sessions and random dinners into lifelong memories - much love, such wow. To Peter, Sathya, and Mubeen - you made Geel weekends fun and unpredictable in the best way. And to my landlord, Wim, my unofficial Belgian guardian - thank you for always ensuring I had a warm, welcoming place to stay and for your Saturday morning words of encouragement.

To my extended gang - Ramesh, Megs, Aju, Harish, Balaji, Deepak, Jay, Ila, Suchi, Anees, Kiran, Vino, Sara, T.S., S.N., Aneestus, Prabhu, Vivek - thank you for your wisdom and constant pep talks. And to Ashok, Appu, Nikunj, Ahmed, Kumar, Jithu, Julan, Piyush, and the whole crew - you've all shaped this journey in ways big and small. You taught me resilience, patience, and maybe a bit of sass.

To my colleagues from Geel, Brussels and Leuven - thank you for the brainstorming sessions, the shared laughter and the collective effort to make science just a little less lonely. A special thanks to our administrative colleagues, whose help in organising events and keeping everything running smoothly made academic life so much easier (and a lot more enjoyable).

To my yoga teacher - thanks for keeping me grounded (literally) and reminding me to breathe through every kind of stretch, including the PhD kind.

This PhD wasn't just about equations and experiments. It taught me courage, balance, and the fine art of keeping calm when nothing compiles. It taught me how to love deeply, live fully, and occasionally laugh at chaos.

As I close this chapter and step into the next, one thing's for sure - this PhD was never mine alone. It was built on the kindness, brilliance, and love of everyone around me. To all of you - thank you for being part of my story.

Popularized Abstract

Many systems around us seem unpredictable. We often don't know *why* things happen the way they do. But here's something fascinating: even when we don't fully understand the internal workings of a system, we can still learn a lot just by observing what goes in (the *input*) and what comes out (the *output*). Can you believe that these two observations alone are often enough to predict how a system might behave in the future? Sounds surprising? Let's explore this with a more familiar example.

Consider one of the most complex systems we know: *human beings*. Our behaviour, emotions, and decisions can be wildly unpredictable. Yet, professionals like doctors, psychologists, and therapists make educated predictions about our conditions and responses every day. How do they do this? By studying *input-output relationships*.

A gastroenterologist, for instance, examines what we consume (food, beverages, and medicine) and the outcomes that follow, such as acidity or ulcers. A cardiologist observes our physical activity and checks for resulting issues like high cholesterol or hypertension. A psychologist might focus on what events trigger fear or stress, and how those inputs result in emotional responses like trauma or anxiety.

In each case, time plays a crucial role. A cardiologist doesn't just ask *what* you eat, but *how often*, for *how long*, and *since when*. This is called *time dependence*. Even more intriguing is *time interaction*: how something that happened in your childhood still influences your health or behaviour as an adult. And when the effects are not proportional, say, a small input leads to a disproportionately large output, we call this *nonlinearity*.

The way we observe and interact with a system also matters. For example, a doctor or psychologist can only understand the underlying issues if the right questions are asked, i.e., in technical terms, if the *excitation* is informative enough to uncover the system's internal patterns. Understanding a system depends not just on the data itself, but also on how that data was collected: the right experiments reveal the right insights.

Another powerful tool professionals use is *prior knowledge*. For example, knowing

your age, gender, genetic conditions, or lifestyle habits helps them make better, faster decisions. This type of background information provides context that can enhance the accuracy of predictions.

So, when all these elements (inputs, outputs, time effects, nonlinearity, excitation, and prior knowledge) come together, professionals try to identify patterns, diagnose problems, and suggest solutions. In engineering and science, this process is known as *system identification*: understanding an unknown system based on the data we observe.

Of course, this isn't always straightforward. There are often unexplained factors, such as knowledge gaps, unusual cases, randomness, and incorrect measurements, that cloud the picture. In our language, we call that *noise*. It's like trying to have a conversation in a crowded room: the message is still there, but it's harder to hear.

To make sense of it all, scientists and engineers rely on *mathematical models* - simplified and structured representations of how a system behaves. These models help us make predictions, test hypotheses, and even control systems automatically. But just like doctors need accurate patient records, these models are only useful if they are built well - with the right assumptions, the right structure, and reliable data.

Just like a doctor must choose the right kind of test or diagnosis tool, engineers must decide on the right *model structure* to represent a system accurately. And just like a treatment must work not just once but over time and across patients, a model must be *validated* to ensure it generalises well beyond the data it was built on. If a diagnosis is too simplistic, it misses crucial details (*underfitting*); if it's overly complicated, it might explain everything yet predict nothing (*overfitting*). The same balance is needed when building models.

In my research, I explore ways to build better mathematical models for complex, nonlinear systems, using *input-output data, prior knowledge, and modern tools like tensor methods and Volterra-based representations*. The goal is to improve how we identify and describe these systems, especially when the data is limited or noisy. Just like in medicine, the better we understand the system, the better our chances of intervening effectively, before things go out of control.

Gepopulariseerde Samenvatting

Er zijn veel systemen om ons heen die onvoorspelbaar lijken. We weten vaak niet *waarom* dingen gebeuren zoals ze gebeuren. Maar hier is iets fascinerends: zelfs als we de interne werking van een systeem niet volledig begrijpen, kunnen we nog steeds veel leren door simpelweg te observeren wat er binnenkomt (de *input*) en wat eruit komt (de *output*). Kun je geloven dat deze twee observaties alleen al vaak voldoende zijn om te voorspellen hoe een systeem zich in de toekomst zou kunnen gedragen? Klinkt dat verrassend? Laten we dit eens onderzoeken met een bekender voorbeeld.

Neem een van de meest complexe systemen die we kennen: de *mens*. Ons gedrag, onze emoties en onze beslissingen kunnen enorm onvoorspelbaar zijn. Toch doen professionals zoals artsen, psychologen en therapeuten dagelijks gefundeerde voorspellingen over onze aandoeningen en reacties. Hoe doen ze dit? Door *input-outputrelaties* te bestuderen.

Een gastro-enteroloog kijkt bijvoorbeeld naar wat we consumeren (voedsel, dranken, medicijnen) en de gevolgen daarvan, zoals zuurtegraad of maagzweren. Een cardioloog observeert onze fysieke activiteit en let op daaruit voortvloeiende problemen zoals een hoog cholesterolgehalte of hypertensie. Een psycholoog kan zich richten op welke gebeurtenissen angst of stress veroorzaken en hoe die inputs emotionele reacties zoals trauma of angst veroorzaken.

In elk geval speelt tijd een cruciale rol. Een cardioloog vraagt niet alleen *wat* je eet, maar ook *hoe vaak*, *hoe lang* en *sinds wanneer*. Dit wordt *tijdsafhankelijkheid* genoemd. Nog intrigerender is *tijdsinteractie*: hoe iets dat in je kindertijd is gebeurd, nog steeds je gezondheid of gedrag als volwassene beïnvloedt. En wanneer de effecten niet proportioneel zijn, bijvoorbeeld wanneer een kleine input leidt tot een onevenredig grote output, noemen we dit *niet-lineariteit*.

De manier waarop we een systeem observeren en ermee interageren, is ook van belang.

Een arts of psycholoog kan bijvoorbeeld de onderliggende problemen alleen begrijpen als de juiste vragen worden gesteld, oftewel, technisch gezien, als de *excitatie* voldoende informatief is om de interne patronen van het systeem te onthullen. Het begrijpen van een systeem hangt niet alleen af van de data zelf, maar ook van hoe die data is verzameld: de juiste experimenten leveren de juiste inzichten op.

Een ander krachtig hulpmiddel dat professionals gebruiken, is *voorkennis*. Kennis van bijvoorbeeld uw leeftijd, geslacht, genetische aandoeningen of leefgewoonten helpt hen om betere en snellere beslissingen te nemen. Dit soort achtergrondinformatie biedt context die de nauwkeurigheid van voorspellingen kan verbeteren.

Wanneer al deze elementen (input, output, tijdseffecten, niet-lineariteit, excitatie en voorkennis) samenkomen, proberen professionals patronen te identificeren, problemen te diagnosticeren en oplossingen voor te stellen. In de techniek en wetenschap staat dit proces bekend als *systeemidentificatie*: het begrijpen van een onbekend systeem op basis van de data die we observeren.

Natuurlijk is dit niet altijd eenvoudig. Er zijn vaak onverklaarbare factoren, zoals kennislacunes, ongebruikelijke gevallen, willekeur en onjuiste metingen, die het beeld vertroebelen. In onze taal noemen we dat *ruis*. Het is alsof je een gesprek probeert te voeren in een volle ruimte: de boodschap is er nog steeds, maar is moeilijker te horen.

Om dit allemaal te begrijpen, vertrouwen wetenschappers en ingenieurs op *wiskundige modellen* - vereenvoudigde en gestructureerde representaties van hoe een systeem zich gedraagt. Deze modellen helpen ons voorspellingen te doen, hypothesen te testen en zelfs systemen automatisch te controleren. Maar net zoals artsen nauwkeurige patiëntendossiers nodig hebben, zijn deze modellen alleen nuttig als ze goed zijn opgebouwd - met de juiste aannames, de juiste structuur en betrouwbare gegevens.

Net zoals een arts de juiste test- of diagnosetool moet kiezen, moeten ingenieurs de juiste *modelstructuur* kiezen om een systeem nauwkeurig weer te geven. En net zoals een behandeling niet één keer moet werken, maar over een langere periode en voor alle patiënten, moet een model *gevalideerd* worden om ervoor te zorgen dat het generaliseert tot ver buiten de data waarop het is gebaseerd. Als een diagnose te simplistisch is, mist deze cruciale details (*underfitting*); als deze te ingewikkeld is, kan deze alles verklaren, maar niets voorspellen (*overfitting*). Dezelfde balans is nodig bij het bouwen van modellen.

In dit werk onderzoek ik manieren om betere wiskundige modellen te bouwen voor complexe, niet-lineaire systemen, met behulp van *input-output data, voorkennis en moderne tools zoals tensormethoden en Volterra-gebaseerde representaties*. Het doel is om de manier waarop we deze systemen identificeren en beschrijven te verbeteren, vooral wanneer de data beperkt of ruizig zijn. Net als in de geneeskunde geldt: hoe beter we het systeem begrijpen, hoe groter de kans dat we effectief kunnen ingrijpen voordat de zaken uit de hand lopen.

Abstract

Mathematical models are essential for understanding, analysing, and predicting the behaviour of real-world systems across a wide range of domains. System identification provides a structured framework for building these models from observed data, enabling accurate predictions and informed decision-making. However, identifying models of complex systems is challenging, particularly in the presence of nonlinearity, noise, or incomplete measurements. Addressing these challenges requires a modelling approach, tailored to the system under study, one that effectively integrates domain knowledge, mathematical insight, and computational tools. While linear approximation techniques are well-established and computationally efficient, they often fail to capture the nonlinear behaviour present in many real-world systems, leading to poor generalisation and loss of interpretability. This work proposes the development of a simple, interpretable nonlinear dynamical model that reduces reliance on expert user input.

We present a novel modelling framework for identifying the nonlinear state-space form of block-oriented systems, specifically Hammerstein, Wiener, and Wiener-Hammerstein structures, by exploiting structural insights rather than optimising linear approximations. Major challenges in this context include selecting the appropriate block configuration, determining system order, and addressing time-consuming identifiability issues, particularly in the presence of complex nonlinearities and high output noise.

To address these challenges, the Volterra series is used as an intermediate modelling step. Due to its linear-in-the-parameters formulation, the Volterra series provides a tractable yet expressive tool for modelling nonlinear systems. When prior knowledge of the underlying block-oriented structure is available, it can be directly translated into a constrained Volterra kernel structure, resulting in a *prior-informed* Volterra kernel that significantly reduces model complexity. However, a key limitation of the Volterra framework is the 'curse of dimensionality', which limits memory length and model order, especially for large datasets in Wiener and Wiener-Hammerstein structures.

To mitigate this, the estimated Volterra kernels are compressed via low-rank approximations, enabling efficient identification of the block-oriented model. This is

followed by regularisation in the reduced parameter space to promote smoothness and decay in the linear dynamics, further improving model interpretability and robustness. From this regularised and optimised block-oriented model, parameters for a Polynomial Nonlinear State-Space (PNLSS) model can be extracted, without requiring an iterative procedure, using standard subspace identification techniques.

The proposed approach is validated through a series of simulations, experiments with measurement datasets, and benchmark datasets from the system identification community. Additionally, the algorithm is tested on several nonlinear systems from the process industry that do not typically conform to a block-oriented model structure. Across all test cases, the block-oriented model identified from the optimised and regularised *prior-informed* Volterra kernel consistently provides a good approximation of the true system, both in terms of output prediction and parameter accuracy. While prediction accuracy decreases with higher noise levels and increased nonlinear complexity, the overall quality of the block-oriented model remains robust.

However, the PNLSS model extracted from the block-oriented model yields mixed results. In some cases, it outperforms the baseline PNLSS models obtained from the linear approximation techniques; in other cases, it performs similarly or slightly worse. These variations suggest that while the proposed Volterra-based approach is highly effective up to the block-oriented model stage, the subsequent realisation of a PNLSS model from this block representation may not always lead to improved output approximation. However, the method offers a systematic, interpretable, and data-efficient alternative to traditional identification workflows, particularly valuable when structural insights into the system are available.

Beknopte Samenvatting

Wiskundige modellen zijn essentieel voor het begrijpen, analyseren en voorspellen van het gedrag van systemen in de praktijk in een breed scala aan domeinen. Systeemidentificatie biedt een gestructureerd raamwerk voor het bouwen van deze modellen op basis van waargenomen data, wat nauwkeurige voorspellingen en weloverwogen besluitvorming mogelijk maakt. Het identificeren van modellen van complexe systemen is echter een uitdaging, met name in de aanwezigheid van niet-lineariteit, ruis of onvolledige metingen. Het aanpakken van deze uitdagingen vereist een modelleringsaanpak die is afgestemd op het bestudeerde systeem en die domeinkennis, wiskundig inzicht en computationele tools effectief integreert. Hoewel lineaire benaderingstechnieken goed ingeburgerd en computationeel efficiënt zijn, slagen ze er vaak niet in om het niet-lineaire gedrag in veel systemen in de praktijk vast te leggen, wat leidt tot slechte generalisatie en verlies van interpreteerbaarheid. Dit werk stelt de ontwikkeling voor van een eenvoudig, interpreteerbaar niet-lineair dynamisch model dat de afhankelijkheid van input van deskundige gebruikers vermindert.

We presenteren een nieuw modelleringskader voor het identificeren van de niet-lineaire toestandsruimtevorm van blokgeoriënteerde systemen, met name Hammerstein-, Wiener- en Wiener-Hammerstein-structuren, door gebruik te maken van structurele inzichten in plaats van lineaire benaderingen te optimaliseren. Belangrijke uitdagingen in deze context zijn het selecteren van de juiste blokconfiguratie, het bepalen van de systeemvolgorde en het aanpakken van tijdrovende identificatieproblemen, met name in de aanwezigheid van complexe niet-lineariteiten en hoge outputruis.

Om deze uitdagingen aan te pakken, wordt de Volterra-reeks gebruikt als een tussenliggende modelleringsstap. Dankzij de lineair-in-de-parameters-formulering biedt de Volterra-reeks een hanteerbare maar expressieve tool voor het modelleren van niet-lineaire systemen. Wanneer voorkennis van de onderliggende blokstructuur beschikbaar is, kan deze direct worden vertaald naar een beperkte Volterra-kernelstructuur, wat resulteert in een *prior-informed* Volterra-kernel die de modelcomplexiteit aanzienlijk vermindert. Een belangrijke beperking van het Volterra-kader is echter de 'vloek van de dimensionaliteit', die de geheugenlengte en modelvolgorde beperkt, met name voor

grote datasets in Wiener- en Wiener-Hammerstein-structuren.

Om dit te beperken, worden de geschatte Volterra-kernels gecomprimeerd via lage-rangbenaderingen, wat een efficiënte identificatie van het blokmodel mogelijk maakt. Dit wordt gevolgd door regularisatie in de gereduceerde parameterruimte om de vloeiendheid en afname in de lineaire dynamiek te bevorderen, wat de interpreteerbaarheid en robuustheid van het model verder verbetert. Uit dit geregulariseerde en geoptimaliseerde blokmodel kunnen parameters voor een Polynomial Nonlinear State-Space (PNLSS)-model worden geëxtraheerd, zonder dat een iteratieve procedure nodig is, met behulp van standaard subruimte-identificatietechnieken.

De voorgestelde aanpak wordt gevalideerd door middel van een reeks simulaties, experimenten met meetdatasets en benchmarkdatasets van de systeemidentificatie-gemeenschap. Daarnaast wordt het algoritme getest op verschillende niet-lineaire systemen uit de procesindustrie die doorgaans niet voldoen aan een blokmodelstructuur. In alle testcases levert het blokmodel dat is geïdentificeerd met behulp van de geoptimaliseerde en geregulariseerde *prior-informed* Volterra-kernel consistent een goede benadering van het werkelijke systeem, zowel wat betreft de outputvoorspelling als de parameternauwkeurigheid. Hoewel de voorspellingsnauwkeurigheid afneemt bij hogere ruisniveaus en een verhoogde niet-lineaire complexiteit, blijft de algehele kwaliteit van het blokmodel robuust.

Het PNLSS-model dat uit het blokmodel is geëxtraheerd, levert echter gemengde resultaten op. In sommige gevallen presteert het beter dan de basis-PNLSS-modellen die zijn verkregen met lineaire benaderingstechnieken; in andere gevallen presteert het vergelijkbaar of iets slechter. Deze variaties suggereren dat, hoewel de voorgestelde Volterra-gebaseerde aanpak zeer effectief is tot aan het blokmodelstadium, de daaropvolgende realisatie van een PNLSS-model op basis van deze blokrepresentatie niet altijd leidt tot een verbeterde outputbenadering. De methode biedt echter een systematisch, interpreteerbaar en data-efficiënt alternatief voor traditionele identificatieworkflows, met name waardevol wanneer structurele inzichten in het systeem beschikbaar zijn.

List of Abbreviations

AE Autoencoder. 249

ALS Alternating Least Squares. 117

ANNs Artificial Neural Networks. 5

ANOVA Analysis Of Variance. 152

BIBO Bounded Input Bounded Output. 142

BLA Best Linear Approximation. 9

CVA Canonical Variate Analysis. 43

DaISy Database for the Identification of Systems. 203

DBSCAN Density-Based Spatial Clustering of Applications with Noise. 78

DFT Discrete Fourier Transform. 26

ERA Eigensystem Realisation Algorithm. 135

FIR Finite Impulse Response. 13

FRF Frequency Response Function. 25

grlex graded lexicographic. 99

HTLS Hankel Total Least Square. 68

i.i.d. independent and identically distributed. 29

LS least squares. 7

LSS Linear State-Space. 12

LSTM Long Short-Term Memory. 5

LTI Linear Time-Invariant. 22

MIMO Multiple Inputs, Multiple Outputs. 34

MOESP Multivariable Output Error State-Space. 43

MPC Model-based Predictive Control. 250

MSE Mean Square Error. 23

N4SID Numerical Algorithms for State-Space Subspace Identification. 43

NARMAX Nonlinear AutoRegressive Moving Average model with eXogenous inputs. 248

NARX Nonlinear AutoRegressive model with eXogenous inputs. 3

NLFIR NonLinear Finite Impulse Response. 28

NLSS Nonlinear State-Space. 9

PACE Polynomial Amplitude Complex Exponentials. 76

PARAFAC PARAllel FACtor decomposition. 118

PDF probability density function. 26, 27

PI-MOESP Past Input - Multivariable Output Error State-Space. 52

PISPO Periodic Input, Same Periodic Output. 32

PNARX Polynomial Nonlinear AutoRegressive model with eXogenous inputs. 10

PNLSS Polynomial Nonlinear State-Space. 9

PO-MOESP Past Output - Multivariable Output Error State-Space. 52

PSD Power Spectral Density. 30

RMS Root Mean Square. 29

RRMSE relative root mean square error. 110

SISO Single Input, Single Output. 12

SNL static nonlinear. 155

SNR Signal-to-Noise Ratio. 30

SUS system under study. 1

SVD singular value decomposition. 41

TI Time-Invariant. 22

TLS total least-squares. 68, 109

TOMFAC TOeplitz Matrix FActor Computation. 118

TTD Tensor Train Decomposition. 170

TTM tensor train matrix. 171

VDMD Vandermonde decomposition. 67

WGN White Gaussian Noise. 28

WH Wiener-Hammerstein. 11

List of Symbols

α_k	Damping factor of the kth exponential
$\boldsymbol{\Delta}$	Controllability matrix
$\boldsymbol{\Gamma}$	Observability matrix
$\boldsymbol{\Sigma}$	Singular value matrix
$\boldsymbol{\theta}_{\text{L}}$	Optimised impulse response coefficients
$\boldsymbol{\theta}_{\text{NL}}$	Optimised static nonlinearity coefficients
$\boldsymbol{\zeta}, \boldsymbol{\eta}$	Vectors of nonlinear monomials in state and output equation, respectively
$\boldsymbol{A}$	State transition matrix
$\boldsymbol{b}$	Input vector in the state evolution equation
$\boldsymbol{c}$	Output vector in the output equation
$\boldsymbol{E}$	Nonlinear mappings in state evolution equation
$\boldsymbol{f}$	Nonlinear mappings in output equation
$\boldsymbol{H}$	Hankel matrix containing Markov parameters
$\boldsymbol{h}$	Vector containing Volterra kernel coefficients for order 0 to D
$\boldsymbol{p}_j$	Polynomial coefficients of static nonlinearity
$\boldsymbol{s}$	Impulse response of first LTI block
$\boldsymbol{t}$	Impulse response of second LTI block
$\boldsymbol{T}_L$	Confluent Vandermonde matrix
$\boldsymbol{U}$	Input regression matrix

$\boldsymbol{U}$	Matrix containing left singular vectors
$\boldsymbol{u}$	Input signal
$\boldsymbol{u}_k$	Vector containing a constant, current and past M inputs
$\boldsymbol{u}_k^D$	D times Kronecker product of $\boldsymbol{u}_k$
$\boldsymbol{V}$	Matrix containing right singular vectors
$\boldsymbol{V}_L$	Vandermonde matrix
$\boldsymbol{w}$	Decay weight vector
$\boldsymbol{x}_k$	State vector at time k
$\boldsymbol{y}$	Output signal
Δt	Sampling time interval
ϵ_k	Prediction error at time k
$\hat{y}_k$	Predicted output at time k
λ	Eigenvalues of a system
$\mathbb{E}\{\cdot\}$	Expectation
μ_d, λ_d	Regularisation parameters
ϕ_k	Phase of the kth exponential
a_k	Amplitude of the kth exponential
c_k	Complex amplitude of the kth exponential
$c_{k\ell}$	Complex amplitude of the kth exponential, for the ℓth repeated mode
D	Degree of nonlinearity in SNL block
d	Feed-through coefficient in the output equation
D_+	Extended degree of nonlinearity in SNL block
$f_{\max}$	Maximum frequency
f_k	Frequency of the kth exponential
h_0	Zeroth-order coefficient of Volterra model
h_d	d^{th}-order coefficient of Volterra model

K	Number of distinct poles
M	Model order (Memory length in the case of the Volterra model)
$M_{\mathrm{h}+}$	Extended memory length of second LTI block
$M_{\mathrm{h}-}$	Model order of second LTI block of PNLSS model
M_{h}	Model order or Memory length of second LTI block
$M_{\mathrm{w}+}$	Extended memory length of first LTI block
$M_{\mathrm{w}-}$	Model order of first LTI block of PNLSS model
M_{w}	Model order or Memory length of first LTI block
M_k	Pole multiplicity structure
m_k	Multiplicity of the kth pole
N	Number of training samples
q_k	Output of the SNL at time k
R	Model order of an autonomous LTI system
u_k	Input at time k
v_k	Input to the SNL at time k
x_n	Output of an autonomous LTI system at time n
y_k	Output at time k
z_k	Pole of the kth exponential

Contents

Popularized Abstract iii

Gepopulariseerde Samenvatting v

Abstract vii

Beknopte Samenvatting ix

List of Abbreviations xiii

List of Symbols xvii

Contents xix

List of Figures xxiii

List of Tables xxvii

1 Introduction 1
- 1.1 Classification of Systems . . . 2
- 1.2 System Identification . . . 3
  - 1.2.1 Data Collection . . . 4
  - 1.2.2 Model Selection . . . 4
  - 1.2.3 Model Fitting . . . 7
  - 1.2.4 Model Validation . . . 8
- 1.3 Overview of Nonlinear Models . . . 9
  - 1.3.1 Block-oriented Models . . . 10
  - 1.3.2 Nonlinear State-Space Model . . . 12
  - 1.3.3 Volterra Series . . . 13
- 1.4 Motivation and Research Objectives . . . 13
- 1.5 Structure of the Thesis . . . 18

2 Principles and Techniques of Linear System Identification 21
2.1 Foundation to Linear Systems . . . 22
2.2 Classical Methods . . . 23
2.3 Overview of Best Linear Approximation . . . 25
2.3.1 Class of Excitation Signals . . . 29
2.3.2 Basics of BLA . . . 32
2.3.3 Properties of BLA . . . 33
2.3.4 Non-parametric Estimation of the BLA for Periodic Data . . . 34
2.3.5 Non-parametric Estimation of the BLA for Non-periodic Data 36
2.3.6 Non-parametric BLA to Parametric BLA . . . 37
2.4 Overview of Subspace-based State-Space Identification . . . 42
2.4.1 Preliminaries . . . 44
2.4.2 Multivariable Output Error State-Space Method . . . 51
2.4.3 Numerical Algorithms for State-Space Subspace Identification 52
2.4.4 MATLAB Commands . . . 56
2.5 Conclusion . . . 57

3 Case Study: Subspace-based Estimation of Multiple Poles 59
3.1 Background and Problem Statement . . . 60
3.2 Mathematical Preliminaries . . . 62
3.2.1 Homogeneous Linear Difference Equation . . . 62
3.2.2 Eigenstructure of Linear Systems: Diagonalisation and Jordan Canonical Form . . . 63
3.3 Harmonic Retrieval of Simple Poles . . . 66
3.4 Generalised Signal Model . . . 69
3.5 Matrix Form of Generalised Signal Model . . . 72
3.6 Proposed Model vs. Polynomial Amplitude Complex Exponentials Model 76
3.7 Estimation of Pole Multiplicity . . . 77
3.8 Numerical Simulations . . . 81
3.8.1 Signal Reconstruction using the Estimated Parameters . . . 81
3.8.2 Estimation of True Multiplicity Structure . . . 83
3.8.3 Real-valued Signal with Noise . . . 83
3.9 Conclusion . . . 85

4 Nonlinear Models for System Identification 89
4.1 Foundation and Focus . . . 90
4.2 Block-oriented Models . . . 90
4.2.1 Hammerstein Model . . . 91
4.2.2 Wiener Model . . . 93
4.2.3 Wiener-Hammerstein Model . . . 94
4.2.4 Challenges in Identification . . . 94
4.3 Polynomial Nonlinear State-Space Model . . . 96
4.3.1 Multi-Index Framework for Multivariate Polynomial Expansion 98

4.3.2 Ordering Multivariate Monomials . . . 99
4.3.3 Polynomial Approximation and Vector Construction . . . 99
4.3.4 Stability Correction . . . 100
4.3.5 Nonlinear Optimisation of the Linear Model . . . 105
4.4 Volterra Series . . . 107
4.5 Identification of Block-oriented Models . . . 109
4.5.1 Decoupling the WH System into Subsystems . . . 109
4.5.2 Pole Allocation Based on the BLA . . . 110
4.5.3 Representing Block-oriented Models Using the PNLSS model 112
4.5.4 Diagonal Volterra Kernel-Based Modelling of WH system . . 116
4.5.5 Tensor-Based Methods for WH Identification . . . 117
4.5.6 Recent Contributions in Nonlinear System Identification . . . 119
4.6 Conclusion . . . 120

5 *Prior-Informed* Volterra Kernel Estimation for Hammerstein Model 121
5.1 Representation of Volterra Kernel . . . 122
5.2 Construction of Hammerstein Kernel . . . 125
5.2.1 *Prior-Informed* (Regularised) Kernel Estimation . . . 128
5.2.2 Reconstructing Hammerstein Structure using Hammerstein Kernel . . . 131
5.2.3 Nonlinear Optimisation in the Block-oriented Model Parameter Space . . . 132
5.2.4 Realising PNLSS Model from the Optimised Hammerstein Kernel . . . 134
5.3 Results and Evaluation - Hammerstein System . . . 143
5.3.1 Numerical Simulations . . . 145
5.3.2 Experimental Validation . . . 158
5.4 Conclusion . . . 164

6 Volterra Kernel Estimation for Wiener System 166
6.1 Construction of Wiener Kernel . . . 167
6.2 Tensor Basics . . . 169
6.2.1 Tensor Train Decomposition . . . 170
6.2.2 Tensor Train Matrices . . . 171
6.2.3 Tensor Diagrams . . . 172
6.3 Wiener Kernel as Tensor Network . . . 173
6.4 Pseudoinverse in TTM Form . . . 175
6.4.1 Construction of Tensor Network . . . 177
6.4.2 Unique Minimum-Norm Solution in TTM Format . . . 182
6.4.3 Impact of Measurement Noise on the TT-Ranks of $\boldsymbol{h}_{\mathrm{symm}}$ . . . 185
6.5 Reconstructing Wiener Structure using Wiener Kernel . . . 186
6.6 Shape-enforced Optimisation of Wiener Kernel . . . 188
6.7 Realising PNLSS Model from the optimised Wiener Kernel . . . 189

6.8 Results and Evaluation - Wiener System . . . 190
6.8.1 Numerical Simulations . . . 191
6.8.2 Experimental Validation . . . 199
6.8.3 Nonlinear Dynamical Systems beyond Block-oriented Structures 203
6.9 Conclusion . . . 209

7 *Prior-Informed* Volterra Kernel Estimation for WH System 210
7.1 Construction of Wiener-Hammerstein Kernel . . . 211
7.2 Reconstructing WH Structure using WH Kernel . . . 215
7.3 Shape-enforced Optimisation of WH Kernel . . . 217
7.4 Realising PNLSS Model from the optimised WH kernel . . . 219
7.5 Results and Evaluation - WH System . . . 220
7.5.1 Numerical Simulations . . . 221
7.5.2 Experimental Validation . . . 228
7.5.3 Nonlinear System Identification Benchmark . . . 230
7.6 Conclusion . . . 233

8 Valorisation Plan 234
8.1 Sharing Research Tools and Data with the Community . . . 234
8.2 Towards a Reusable Software Pipeline . . . 236
8.3 Conclusion . . . 236

9 Conclusions 239
9.1 Revisiting the Research Objectives . . . 239
9.2 Scope for Future Research . . . 245

A Tensor Preliminaries 251
A.1 Introduction to Tensors . . . 251
A.1.1 Tensor Unfolding (Matricisation) . . . 251
A.1.2 Tensor Reshaping and Vectorisation . . . 252
A.1.3 Tensor-Matrix Products, Vector Contractions, and the Kronecker Product . . . 253

Bibliography 255

Curriculum Vitae 273

Publication List 275

List of Figures

1.1 Block schematic of a system 2
1.2 RC circuit 2
1.3 The modelling continuum: Physics vs. Data-driven 4
1.4 A mass-spring-damper system 5
1.5 Standard block-oriented models 11
1.6 Outline of the thesis 16
1.7 Flow of the remainder of the chapters 18

2.1 Different characteristics of stochastic input 27
2.2 Effect of input frequency content on the BLA 28
2.3 Gaussian noise excitation 30
2.4 Random phase multisine 31
2.5 BLA representation 33
2.6 BLA - Design of experiment 34
2.7 Zero-input response 46

3.1 Schematic of an exponential polynomial modelling and estimation . . 61
3.2 Generalised signal models and pole-zero maps 70
3.3 Performance of Subspace vs. Linear prediction 82
3.4 Error analysis of multiplicity estimation for case 1 84
3.5 Subspace-based signal estimation for case 1 85
3.6 MSE and multiplicity estimation for case 2 86
3.7 Performance of pole estimation and signal prediction 86

4.1 Static nonlinearity 91
4.2 Unbounded State and Output 101
4.3 Parameters influencing stability 102

5.1 Illustration of 3-way Volterra kernel tensor 125
5.2 Hammerstein model structure 126
5.3 Illustration of a 3-way symmetric Hammerstein kernel tensor 127

5.4 Low-rank structure in the noiseless case . . . 139
5.5 SVD in the noisy case . . . 140
5.6 Illustration of static nonlinearities used in simulation . . . 146
5.7 Simulation dataset - Hammerstein system with dead zone nonlinearity 149
5.8 Output prediction accuracy at SNR 5 dB - Hammerstein model . . . 152
5.9 Box plots of RRMSE (%) for PNLSS models under different nonlinearities and SNR levels – Hammerstein system . . . 155
5.10 Multiple comparison results (Tukey–Kramer test) – Hammerstein system 156
5.11 Parameter estimation accuracy at SNR 5 dB - Hammerstein model . . 157
5.12 Performance comparison of PNLSS models - Hammerstein model . . 157
5.13 True and estimated LTI filter characteristics - Hammerstein Model . . 158
5.14 Model performance on validation and test sets (dead zone, SNR 5 dB) 159
5.15 Measurement setup of Hammerstein system . . . 159
5.16 Diode circuit used in the SNL block . . . 160
5.17 Measurement dataset - Hammerstein system with nonlinearity built using a diode circuit . . . 161
5.18 FRF of 3rd-order inverse Chebyshev filter . . . 163
5.19 Output prediction and performance comparison of PNLSS models at SNR 25 dB - Hammerstein system . . . 164
5.20 Model performance on validation and test sets of Hammerstein measurement . . . 165

6.1 Wiener model structure . . . 168
6.2 Tensor cores . . . 170
6.3 Elements of tensor diagrams . . . 172
6.4 Tensor train matrix representation . . . 172
6.5 Construction of TTM of repeated Khatri-Rao product . . . 179
6.6 Comparison of conventional and modified upper bounds of TTM ranks 181
6.7 Tensor diagram of SISO Volterra model . . . 181
6.8 TTM-based pseudoinverse . . . 183
6.9 Simulation dataset - Wiener system with a third-order polynomial nonlinearity . . . 192
6.10 Output prediction accuracy at SNR_2 - Wiener Model . . . 195
6.11 Box plots of RRMSE (%) for PNLSS models under different nonlinearities and SNR levels - Wiener system . . . 196
6.12 Multiple comparison results (Tukey–Kramer test) – Wiener system . . 197
6.13 Parameter estimation accuracy at SNR_2 - Wiener Model . . . 198
6.14 True and estimated LTI filter characteristics and SNL block - Wiener model . . . 198
6.15 Performance comparison of PNLSS models - Wiener model . . . 199
6.16 Model performance on validation and test sets - Wiener Model . . . 199
6.17 Measurement setup of Wiener system . . . 200

6.18 Measurement dataset - Wiener system with nonlinearity built using a diode circuit . . . 200
6.19 FRF of third-order Chebyshev (Type I) Filter . . . 201
6.20 Visualisation of Wiener kernel up to third-order . . . 202
6.21 Output prediction accuracy and parameter comparison of PNLSS models at SNR 25 dB - Wiener System . . . 202
6.22 Model performance on validation and test sets - Wiener system . . . 203
6.23 Predicted vs. true pH values for the neutralisation process . . . 205
6.24 Predicted vs. true outlet liquid temperature in the heat exchanger . . . 207
6.25 Predicted vs. true outputs of the continuous stirring tank reactor . . . 208

7.1 Reshaped symmetric second-order Wiener-Hammerstein kernel $\boldsymbol{H}$, showing banded structure. . . . 213
7.2 Illustration of 3-way Wiener-Hammerstein kernel tensor . . . 214
7.3 Simulation dataset - WH system with a third-order polynomial nonlinearity 222
7.4 Output prediction accuracy at SNR 10 dB - WH system . . . 224
7.5 Box plots of RRMSE (%) for PNLSS models under different nonlinearities and SNR levels - WH system . . . 225
7.6 Multiple comparison results (Tukey–Kramer test) – WH system . . . 226
7.7 Parameter estimation accuracy at SNR 10 dB . . . 227
7.8 True and estimated LTI filter characteristics . . . 227
7.9 True and estimated pole-zero map and SNL coefficients . . . 228
7.10 Performance comparison of PNLSS models - WH Model . . . 228
7.11 Model performance on validation and test sets - WH Model . . . 229
7.12 Measurement setup of Wiener-Hammerstein system . . . 229
7.13 Benchmark dataset - Wiener-Hammerstein system with nonlinearity built using a diode circuit . . . 231
7.14 FRFs of 3rd-order Chebyshev type I and II filters . . . 232
7.15 Visualisation of WH kernel up to third-order . . . 232
7.16 Output prediction accuracy and performance comparison of PNLSS models . . . 233
7.17 Model performance on validation and test sets . . . 233

List of Tables

3.1 True parameters of the signals 82

5.1 List of models used throughout the thesis 149
5.2 Performance on Simulated Test Dataset with 3 Types of Nonlinearity - Hammerstein System 150
5.3 General structure of ANOVA table 153
5.4 Decision rule (common thresholds) for p-values 153
5.5 Effect size interpretation for η^2 154
5.6 ANOVA Table - Hammerstein simulation dataset 154
5.7 Performance on the Measurement Dataset 162

6.1 Performance on Simulated Test Dataset with 3 Types of Nonlinearity - Wiener System 194
6.2 ANOVA Table - Wiener simulation dataset 196
6.3 Performance on the Measurement Dataset 201
6.4 Performance on Process Industry System I - Wiener Model 205
6.5 Performance on Process Industry System II - Wiener Model 206
6.6 Performance on Process Industry System III - Wiener Model 208

7.1 Performance on Simulated Test Dataset with 3 Types of Nonlinearity - Wiener-Hammerstein System 223
7.2 ANOVA Table - WH simulation dataset 225
7.3 Performance on the Measurement Dataset 230
7.4 Performance on the Benchmark Dataset 231

9.1 Performance Summary of the Hammerstein System for Simulation (Sim.) and Measurement (Meas.) Datasets 241
9.2 Performance Summary of the Wiener System for Simulation (Sim.), Measurement (Meas.), and Benchmark (Ben.) Datasets 242
9.3 Performance Summary of the WH System for Simulation (Sim.), Measurement (Meas.), and Benchmark (Ben.) Datasets 244

Chapter 1

Introduction

Humans have sought to understand and predict the world around them for centuries. Epidemiologists, for example, used models to forecast the spread of COVID-19, allowing governments to plan healthcare resources and implement timely interventions (Shinde et al., 2020; Ojokoh et al., 2022). Several nations employ advanced air defence technologies to predict and intercept incoming threats, protecting residents from missile attacks (Hamtil et al., 2015; Papp, 2017). Conservationists monitor the risks of deforestation with Global Forest Watch (Zhang et al., 2020; Shea, 2022), a tool to analyse satellite data to predict and track illegal logging, aiding forest management, and ensuring transparency to hold governments and industries accountable. These predictions rely on models—simplified representations of reality constructed using past observations, prior knowledge, and constraints. These models simplify complex ideas into simpler concepts, enabling us to make informed decisions and plan for the future.

In science and engineering, such simplifications are known as systems, entities that interact with their environment through well-defined inputs and outputs. System identification is the process of representing a system through a mathematical model based on observed data. It plays a critical role in bridging theory and application. By understanding the relationships between inputs and outputs, system identification enables prediction, control, design, and optimisation in diverse fields, including medicine, military applications, climate science, and engineering.

However, constructing accurate models to represent such complex real-world systems is a challenging task, particularly when the recorded data originates from systems that are noisy, incomplete, or nonlinear. To address these challenges, one must tailor the modelling approach to the specific system under study (SUS), exploiting domain knowledge, mathematical tools, and computational techniques to reveal the underlying dynamics.

1.1 Classification of Systems

In the study of system identification, systems are broadly categorised based on their input-output relationships. A block representation of a system is given in Figure 1.1.

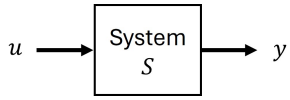


Figure 1.1: Block representation of a system defining signals. u: Input signal (could be scalar, vector, time-domain, frequency-domain, etc.), y: Output signal, and S: The system block, which can represent any mapping $S : u \rightarrow y$

A system with input u and output y is called linear if and only if it follows the principles of homogeneity and scaling (together referred to as superposition). This means that, at the level of input and output signals, the response to a linear combination of inputs is the corresponding linear combination of the individual responses. Mathematically, linear systems are often represented by linear differential equations for continuous-time systems, or difference equations for discrete-time systems. A classical example of a continuous-time linear system is a resistor-capacitor (RC) circuit (see Figure 1.2), where the output voltage across the capacitor, denoted by $v_{\text{out}}(t)$, depends linearly on the input voltage $v_{\text{in}}(t)$ and the rate of change of output $\dot{v}_{\text{out}}(t)$, as described by the following differential equation:

$$RC\dot{v}_{\text{out}}(t) + v_{\text{out}}(t) = v_{\text{in}}(t). \tag{1.1}$$

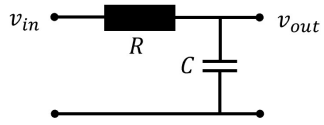


Figure 1.2: Series RC circuit

To analyse the system in the frequency domain, we apply the Laplace transform, which converts time-domain functions (e.g., $v_{\text{in}}(t)$) into functions of the complex frequency variable s. The Laplace transform of $v_{\text{out}}(t)$ is denoted by $V_{\text{out}}(s)$, and similarly for the input. Applying the Laplace transform to the differential equation (assuming zero initial conditions), we obtain:

$$V_{\text{out}}(s) = \frac{1}{1 + RCs} V_{\text{in}}(s), \tag{1.2}$$

where s is the Laplace variable representing complex frequency. This input-output relationship, defined in the Laplace domain, is called the transfer function.

Due to their simplicity, linear systems are widely used to model applications in control theory, signal processing, and telecommunications (Chen, 1984; Antsaklis and Michel, 2005; Ogata, 2010).

Systems, where the change in output is not proportional to the change in input, are generally termed nonlinear systems. Nonlinearity is of particular interest to many scientists, as most real-life systems inherently exhibit this property. A practical example of a nonlinear physical system is an automotive suspension system, particularly with hydraulic dampers. The damper subsystem resists motion by generating a force that depends on the relative velocity of the piston (i.e., the speed difference between the two ends of the damper). This force-velocity relationship is typically nonlinear and further depends on operating conditions such as temperature and input frequency. Due to the presence of memory effects and nonlinear relationships, such systems are often modelled using a Nonlinear AutoRegressive model with eXogenous inputs (NARX) (Alghafir and Dunne, 2012), which is traditionally defined in discrete-time. A simplified NARX model for the damper subsystem with relative piston velocity as input u_k, and damping force output y_k is of the form:

$$y_k = a_1 y_{k-1} + a_2 y_{k-2} + b_1 u_{k-1} + b_2 u_{k-2} + c_1 u_{k-2}^2, \tag{1.3}$$

where the term u_{k-2}^2 allows the model to capture some of the nonlinear behaviour of the damper. Nonlinear models are essential in accurately capturing the dynamics of real-world processes. Due to their ability to represent complex behaviours, nonlinear systems are often used in time-series prediction, control, and modelling of physical systems with memory and nonlinear dynamics.

1.2 System Identification

System identification is the process of developing mathematical models of dynamic systems from observed input-output data. It provides a structured framework to bridge the gap between empirical observations and theoretical understanding by revealing the system's underlying dynamics. With accurately identified models, engineers and researchers can analyse system behaviour, enhance its performance, optimise processes, and address real-world challenges innovatively. The classic references for system identification are the works in Söderström and Stoica (1989); Ljung (1999); Pintelon and Schoukens (2012). This section outlines the essential steps in the system identification process: data collection, model selection, model fitting, and model validation.

1.2.1 Data Collection

This step involves gathering relevant input-output data from the SUS. The quality and quantity of the collected data directly impact the accuracy of the identified model. Key considerations in this phase include:

1. Proper design of experiments ensures the collected data adequately excites all relevant system dynamics. This includes choosing the type, amplitude, and frequency range of the input signals, i.e., the input should be sufficiently (persistently) exciting.
2. The sampling rate must be chosen carefully to capture the dynamics of the system without aliasing.
3. Measurements are prone to noise and disturbances. Thus, an over-parametrised model generalises the SUS to unseen data poorly.
4. Data preprocessing steps, such as filtering and normalisation, are often necessary to remove irrelevant components and prepare the data for analysis.

1.2.2 Model Selection

The crucial step in the system identification process is model selection due to the availability of numerous mathematical models to represent the SUS. The model selection can be based on several factors, including:

1. **Prior Knowledge**: Incorporating prior knowledge about the system, such as physical laws or structural constraints, can guide the selection of an appropriate model structure. Based on the extent of the available prior knowledge about the system, the modelling can be categorised as white-box, grey-box, and black-box (see Figure 1.3).

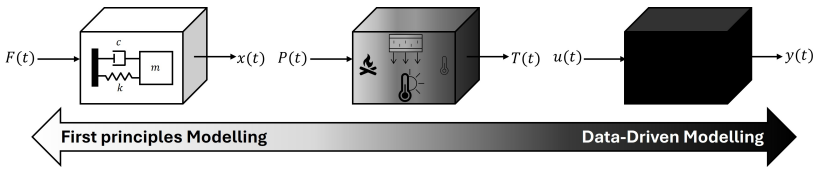


Figure 1.3: Illustration of the modelling continuum based on system knowledge and transparency: from physics-based white-box models, through grey-box models combining data and structure, to purely data-driven black-box models.

The white-box modelling approach is also known as first principles or physics-based modelling, where the system's structure, properties, and working principles are known,

and some unknown parameters need to be determined. Consider a mass-spring-damper system in Figure 1.4, with input force $F(t)$ and output displacement $x(t)$. The objective

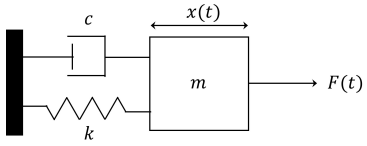


Figure 1.4: A schematic of a mass-spring-damper system with force input $F(t)$ and displacement output $x(t)$.

is to identify the system parameters such as mass m, damping coefficient c, and spring constant k from the input-output data. The governing equation of the system, expressed in continuous time, is derived from Newton's second law as

$$m\ddot{x}(t) + c\dot{x}(t) + kx(t) = F(t), \tag{1.4}$$

where $\dot{x}(t)$ and $\ddot{x}(t)$ are velocity and acceleration, respectively. The system's structure is fully known and deterministic, provided that there are no disturbances. The system behaves exactly as described by the equation; thus, it is a white-box model. White-box models are mostly theoretical because real-world systems often have unmodelled noise, disturbances, dynamics, and parameter variations. In such cases, the physics-based mathematical model of the system may not accurately represent the system.

On the contrary, the black-box models have no explicit knowledge of the internal structure of the system, physics, or governing equations. The modelling approach relies purely on the observed data, hence the name data-driven modelling approach. This approach can handle highly nonlinear systems and is often faster to develop than the first-principles models. The major shortcomings of this approach are no physical interpretation of the identified parameters, poor generalisation to unseen data, and the need for a lot of data samples to train the model.

A prominent example of black-box modelling is the use of artificial neural networks (ANNs), which can approximate a wide range of nonlinear input-output relationships without requiring prior structural knowledge of the system. Depending on the application, different neural network architectures can be employed: for instance, feedforward networks can capture static nonlinearities; recurrent or long short-term memory (LSTM) networks can model dynamic systems with temporal dependencies; and cascaded or deep feedforward networks can mimic layered nonlinearities (Ljung et al., 2020; Warsito et al., 2018). These models typically achieve high prediction accuracy when trained with sufficient data but often at the expense of interpretability, as their internal representations and parameters lack direct physical meaning.

Interpretability in the context of this thesis refers to the extent to which a model's parameters and structure can be associated with physically meaningful characteristics

of the underlying system. A model is considered interpretable when its parameters correspond to quantities that have a clear physical or functional interpretation, such as linear dynamics, static nonlinearities, or saturation effects, and when the overall model structure reflects a plausible mechanism of the real system. In this sense, interpretability is closely related to *plausibility*: an interpretable model not only reproduces the observed input-output behaviour but also provides insight into the underlying physics or signal-generation process.

There is a vast class of systems where partial knowledge of systems' internal structure and physics is available, yet data-driven estimation is used to determine the unknown parameters. These systems are grey-box models as they combine the former two modelling approaches. Consider the example of a room heating system shown in Figure 1.3, where $P(t)$ is the power supplied to the heater and $T(t)$ is the room temperature. The system is described in continuous time, where the rate of change of room temperature is given by subtracting the heat loss from the heat gain as below:

$$C\dot{T}(t) = P(t) - k\{T(t) - T_{\mathrm{o}}(t)\}, \tag{1.5}$$

where $T_{\mathrm{o}}(t)$ is the outside temperature, C is the heat capacity of the room and k is the heat loss coefficient. A commonly used representation for such systems is the transfer function model, which describes the input-output relationship of a system in the frequency or Laplace domain. Assuming zero initial conditions and $T_{\mathrm{o}}(t) = 0$ (i.e., modelling temperature relative to the outside), the differential equation (1.5) yields the following transfer function $G(s)$ from the heater power $P(s)$ to the room temperature $T(s)$:

$$G(s) = \frac{T(s)}{P(s)} = \frac{1}{Cs + k}, \tag{1.6}$$

where s is the complex frequency variable in the Laplace domain. The transfer function provides a compact and analytically tractable model that captures the input-output behaviour of the system without explicitly modelling all the internal states. In grey-box modelling, the structure of the transfer function is often derived from first principles, while parameters such as C and k are estimated from data. If $T_{\mathrm{o}}(t)$ is not negligible, the system becomes a two-input, one-output system, and a single transfer function from $P(t)$ to $T(t)$ no longer suffices. In such cases, a full description involves a pair of transfer functions: one from the heater input and another from the outside temperature. Neglecting the influence of $T_{\mathrm{o}}(t)$ may lead to biased or inaccurate model estimates unless it is either measured and included in the model or its effect is sufficiently small.

Although the systems' basic physics (heat transfer principles) are known, certain uncertainties prevent it from being a white-box model. The first-order heat transfer equation (1.5) does not entirely capture the real-world complexities such as radiation, convection, and heat storage in walls and furniture. The loss coefficient k varies with the insulation quality, ventilation, and other unknown disturbances that are difficult to measure. The outside temperature $T_{\mathrm{o}}(t)$ may not be available, and the contribution

of additional heat sources, such as sunlight and people in the room, is not considered. Thus, the model equation is only an approximation of the system, based on the available partial prior knowledge. The grey-box models have the advantages and disadvantages of both the white- and black-box models.

2. **Model Structure**: The popular model structure is the linear model due to its simplicity in modelling and estimation. However, many real-world systems do not admit a linear model structure. Thus, using linear models to represent these systems would inevitably result in poor approximation. Nonlinear models can represent both static nonlinearities (such as saturation, dead zone, or friction) and dynamic nonlinear behaviours (such as hysteresis or amplitude-dependent frequency responses) that linear models cannot capture. They provide greater flexibility and improved accuracy, making them suitable for a wide range of applications. However, parameter estimation in nonlinear models is more complex, often requiring iterative procedures that are computationally expensive and sensitive to initialisations. They also lack the intuitive interpretability that linear models offer, making it harder to gain insights into system behaviour.

3. **Application specific**: The intended application of system identification also plays a role in model selection. For example, in control systems, stability and real-time responsiveness are critical, whereas in weather forecast applications, accuracy is critical. Thus, choosing an appropriate model can lead to improved performance and reduced computational costs.

1.2.3 Model Fitting

Fitting the selected model is the process of estimating the unknown parameters using the observed data samples. Based on the selected model, this step might require solving an optimisation problem to minimise the error between the true and the predicted output. In practice, measurement noise is an unavoidable challenge. Thus, model parameters estimated from noisy data lead to a stochastic nature in their values. To address this, parameter estimation is often approached using statistical methods.

Least Squares (LS) is a popular statistical method used to estimate the model parameters by minimising the sum of squared differences between the true and the predicted output. For a system where the current output depends on a finite number of past input values, with input u_k, output y_k at time k, and memory length M, the model approximates the true output as:

$$\hat{y}_k(\boldsymbol{\theta}) = f(\,[u_k, \ldots, u_{k-M}];\, \boldsymbol{\theta}\,), \tag{1.7}$$

where $f(\,\cdot\,;\boldsymbol{\theta}\,)$ is a function parametrised by $\boldsymbol{\theta}$. The true output is modelled as:

$$y_k = \hat{y}_k(\boldsymbol{\theta}) + \epsilon_k(\boldsymbol{\theta}), \tag{1.8}$$

where $\epsilon_k(\boldsymbol{\theta})$ is the prediction error. The LS cost function is then given as:

$$V(\boldsymbol{\theta}) = \sum_{k=1}^{N} \epsilon_k(\boldsymbol{\theta})^2, \tag{1.9}$$

where N is the number of samples. The LS estimate $\hat{\boldsymbol{\theta}}_{\mathrm{LS}}$ is found by minimising the above cost function:

$$\hat{\boldsymbol{\theta}}_{\mathrm{LS}} = \arg\min_{\boldsymbol{\theta}} V(\boldsymbol{\theta}). \tag{1.10}$$

If the model is linear in the parameters, as given in (1.2) and (1.3), where the model output can be expressed as the inner product of a regressor vector and a parameter vector, i.e., $\hat{y}_k = \boldsymbol{\phi}_k^\top \boldsymbol{\theta}$, then the cost function becomes a quadratic function of $\boldsymbol{\theta}$, which has a closed-form solution:

$$\hat{\boldsymbol{\theta}}_{\mathrm{LS}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{y}, \tag{1.11}$$

where $\boldsymbol{\Phi}$ is the regression matrix built from input-output data, and $\boldsymbol{y} = [y_1, \ldots, y_N]^\top$ is the output vector. However, some models cannot be expressed in linear-in-the-parameters form. For instance, modelling of enzyme reaction rates in biochemical systems is a nonlinear process, where the output depends on a Michaelis–Menten-type nonlinear relationship (Fey and Bullinger, 2010). In a continuous stirred-tank reactor (CSTR), the substrate concentration u_k (input) affects the reaction rate or flow y_k (output) through a nonlinear enzymatic reaction. A simplified model is given by:

$$y_k = \frac{V_{\max} \cdot u_k}{K_M + u_k}, \quad k = 1, \ldots, N, \tag{1.12}$$

where $V_{\max}$ is the maximum reaction rate, and K_M is the Michaelis constant, which represents the substrate concentration at which the reaction rate is half of $V_{\max}$. This model is often used in enzyme kinetics to describe substrate saturation effects such as those encountered in fermentation processes or drug metabolism. Unlike models such as (1.2) and (1.3), which are linear-in-the-parameters, the Michaelis–Menten model in (1.12) is nonlinear-in-the-parameters because K_M appears in the denominator. As a result, it cannot be expressed as a linear combination of the unknown parameters. For such nonlinear-in-the-parameters models, the cost function $V(\boldsymbol{\theta})$ is generally non-convex, and iterative numerical techniques such as gradient descent, Gauss-Newton, or Levenberg-Marquardt are used to minimise it.

1.2.4 Model Validation

After estimating the model using measured data, the 'goodness' of the fit must be validated. Validation helps in assessing the accuracy and generalisability of the identified model. The common practice in system identification is to split the available

data into three datasets: an estimation (or training) dataset, a validation dataset, and a test dataset. The estimation dataset is used to train the model and to perform parameter optimisation, often resulting in several candidate models. The validation dataset is then used to evaluate these models, and the one that yields the lowest error on the validation data is selected as the best model. Finally, the test dataset (unseen during training and model selection) is used to evaluate this chosen model, and the corresponding error is reported as the final measure of 'goodness' of fit. This step enables the avoidance of data overfitting, especially in the case of black-box modelling. Overfitting occurs when the model is over-parametrised; consequently, the model starts to fit the noise.

1.3 Overview of Nonlinear Models

Modelling the dynamics of a system is crucial for predicting its behaviour, designing control systems, diagnosing faults, making informed decisions, and improving system efficiency and reliability. Many real-life systems typically exhibit nonlinear behaviour. Employing linear approximation techniques, such as the best linear approximation (BLA) (Pintelon and Schoukens, 2012), to identify such systems would inevitably result in significant deviations from their actual representation. Thus, a simple, interpretable nonlinear dynamical model that minimises dependence on expert user input (Khalil, 2002) is sought.

Various techniques have been developed for modelling nonlinear dynamic systems. These can be broadly categorised as follows, based on the role of prior knowledge in model construction:

1. **Knowledge-driven Nonlinear Modelling Approaches**: These rely on physical insights, domain knowledge, or predefined structural forms. While parameters are estimated from data, the model structure is informed or constrained by prior system understanding. Examples include:

- **Block-oriented models** combine static nonlinearities with linear dynamics to capture system behaviour efficiently (Schoukens and Tiels, 2017),
- **Volterra series** approximate a wide range of nonlinear systems using a functional expansion, making them well-suited for modelling nonlinear dynamic systems with fading memory (Wiener, 1942; Schetzen, 1980; Rugh, 1981; Boyd and Chua, 1985),
- **Nonlinear State-Space (NLSS) models** provide a general framework to model system dynamics with nonlinear functions in state and output equations (Jazwinski, 1970; Verdult, 2002; Anderson and Moore, 2012). A notable variant is the polynomial NLSS (PNLSS) (Paduart et al., 2007; Decuyper et al., 2021), where the nonlinearities are explicitly represented using polynomial functions,

- **NARX models** are valued for their flexibility in capturing complex input-output relationships using past inputs and outputs (Billings, 2013). When the nonlinear mapping is chosen to be polynomial, the model is referred to as a polynomial NARX (PNARX) (Nikolakopoulou and Braatz, 2023).

2. **Data-Driven Nonlinear Modelling Approaches**: These prioritise flexibility and rely primarily on observed data rather than prior physical knowledge to infer system behaviour. While prior knowledge can be incorporated (e.g., through architecture, basis selection or regularisation), these models typically function with minimal structural assumptions. Examples include:

- **Neural network-based models** learn complex nonlinear mappings directly from data, often treating the system as a black-box (Rojas, 1996),
- **Koopman operator theory** lifts nonlinear dynamics into higher-dimensional linear space, typically identified from data (Mauroy et al., 2020),
- **Dynamic Mode Decomposition (DMD)** extracts dominant spatio-temporal patterns from data to model system evolution (Kutz et al., 2016),
- **Sparse Identification of Nonlinear Dynamics (SINDy)** identifies governing equations from data using a sparse set of basis functions (Brunton et al., 2016),
- **Probabilistic methods:** The Bayesian approach incorporates uncertainty and can include prior distribution on model parameters (Peterka, 1981), while Hidden Markov Models capture systems with unobserved (hidden) states that evolve stochastically (Dymarski, 2011).

Despite their capabilities, many nonlinear modelling techniques suffer from high complexity, limited user-friendliness, significant computational cost, and application-specific challenges (Rojas, 1996). This thesis investigates how nonlinear dynamic systems can be effectively modelled using structured representations, such as block-oriented models, Volterra series, and nonlinear state-space models, based on input-output data. The focus lies on model classes that provide interpretability and incorporate prior knowledge, while remaining flexible enough to capture nonlinear behaviours. The following sections briefly discuss the nonlinear models employed in this research.

1.3.1 Block-oriented Models

Block-oriented models are a popular choice for nonlinear system identification due to their simplicity, interpretability, and effectiveness in capturing nonlinear dynamic behaviour. These models consist of interconnected static nonlinearities and linear dynamic subsystems, enabling modular design and clear physical interpretation. Their simplicity has enabled successful applications across fields such as digital communications, physiological systems, and chemical processes (Marmarelis, 1993;

Bershad et al., 2001). In this thesis, the focus is on three common block-oriented structures:

- Hammerstein - a static nonlinear block cascaded with a linear dynamic block,
- Wiener - a linear dynamic block followed by a static nonlinear block, and
- Wiener-Hammerstein (WH) - a static nonlinear block sandwiched between linear dynamic blocks (see Figure 1.5).

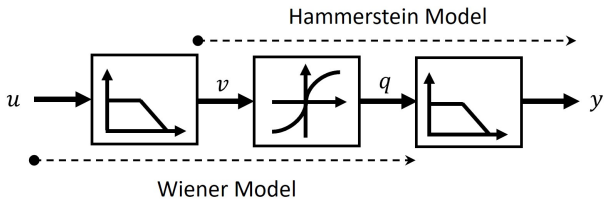


Figure 1.5: WH model with a static nonlinear block sandwiched between two linear dynamic blocks, also showing Wiener and Hammerstein models as special cases. The input is denoted by u, the output by y, and v and q represent intermediate signals.

Let u_k, y_k, v_k, and q_k denote the input, output, and intermediate signals at time k, respectively. The linear dynamic blocks are modelled using FIR filters with impulse responses $\boldsymbol{s}$ and $\boldsymbol{t}$ of memory lengths M_{w} and M_{h}, respectively, and the static nonlinearity is represented as a polynomial of order D and coefficients p_j, i.e.,

$$v_k = \sum_{i=0}^{M_{\mathrm{w}}} s_i u_{k-i}, \quad q_k = \sum_{j=1}^{D} p_j v_k^j, \quad y_k = \sum_{i=0}^{M_{\mathrm{h}}} t_i q_{k-i}. \tag{1.13}$$

Block-oriented models offer excellent interpretability and modular structure, which makes them attractive for both theoretical analysis and practical implementation. However, identifying the correct configuration, selecting model orders, and separating the linear and nonlinear components can be challenging, especially in the presence of measurement noise or strong nonlinearities. Classic references on block-oriented models include Hammerstein (1930); Wiener (1958); Narendra and Gallman (1966); Billings (1980); Hunter and Korenberg (1986); Westwick and Kearney (2003); Giri and Bai (2010); Mzyk (2014).

1.3.2 Nonlinear State-Space Model

The NLSS model is a flexible and powerful mathematical framework used to describe dynamic systems, as it provides a compact representation of internal system dynamics and allows for modelling complex, time-varying nonlinear phenomena. It separates the system's behaviour into states, inputs, and outputs, providing valuable insights into the dynamics of the system. Due to their ability to provide changes in internal system dynamics at every time step, these models are widely used in applications such as control systems and robotics, where fault diagnosis and timely action are necessary.

The NLSS model describes the system by the state equation that defines the evolution of state variables over time and the output equation that defines how the states affect the observed measurements. Let $\boldsymbol{x}_k \in \mathbb{R}^M$ denote the state vector at time k, M the model order, $u_k \in \mathbb{R}$ the input, and $y_k \in \mathbb{R}$ the output. The discrete-time representation of a linear state-space (LSS) model for a single-input single-output (SISO) system at time instant $k \in \{0, 1, \ldots, N\}$ is given by:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}u_k, \\ y_k &= \boldsymbol{c}^\top \boldsymbol{x}_k + du_k, \end{aligned} \tag{1.14}$$

where the matrix $\boldsymbol{A} \in \mathbb{R}^{M\times M}$ is the state transition matrix and $\boldsymbol{b} \in \mathbb{R}^M$ is the input vector in the state evolution equation. The output vector is denoted by $\boldsymbol{c} \in \mathbb{R}^M$ and $d \in \mathbb{R}$ is the feed-through coefficient in the output equation. The order M relates to, but is not necessarily equal to, the number of delayed inputs and outputs in the difference equation. In the nonlinear case, the model generalises to:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= z_1(\boldsymbol{x}_k, u_k), \\ y_k &= z_2(\boldsymbol{x}_k, u_k), \end{aligned} \tag{1.15}$$

where the nonlinear functions z_1, z_2 represent the state transition and output mapping, and they can be described using one of the several sets of basis functions, such as polynomials, sigmoids, wavelets, radial basis functions, etc.

The NLSS model can represent a wide range of nonlinear effects that can model real-world nonlinearities such as saturation, hysteresis, chaos, etc (Suykens et al., 1995; Paduart et al., 2007; Schön et al., 2011; Marconato et al., 2012). While they offer greater flexibility and accuracy, they also introduce challenges in parameter estimation, stability analysis, and computational complexity, often requiring advanced numerical techniques for system identification and prediction.

1.3.3 Volterra Series

The Volterra series (Schetzen, 1980) is a powerful tool for modelling nonlinear dynamic systems, due to its ability to capture memory effects. The Volterra series extends the linear Finite Impulse Response (FIR) model to higher orders of nonlinearity by adding homogeneous polynomials in the delayed input (memory effect). Thus, the model can capture complex nonlinearities such as harmonic distortion and intermodulation, making it useful in biomedical engineering, electrical engineering, and system identification.

The Volterra series describes the output y_k of a nonlinear system as a series expansion of the input u_k. It captures both linear and nonlinear interactions through a set of system-specific coefficients, known as the Volterra kernel. Since the model considers only a finite memory length M, it represents a truncated Volterra series, where the influence of past inputs is limited to a finite window. The components of the model are:

- h_0: a constant representing the system's baseline response,
- $h_d(\cdot, \cdots, \cdot)$: the dth-order Volterra kernel, which characterises the system's response to interactions between inputs at d number of instances,
- D: the maximum degree of nonlinearity in the model,
- M: the finite memory length of the system.

Given this, the discrete-time SISO Volterra model at time k is expressed as:

$$y_k = h_0 + \sum_{d=1}^{D} \left(\sum_{m_1=0}^{M} \cdots \sum_{m_d=0}^{M} h_d(m_1, \cdots, m_d) \prod_{j=1}^{d} u_{k-m_j} \right). \tag{1.16}$$

This represents the system's output as a weighted sum of past inputs, where the higher-order kernel captures nonlinear effects.

By systematically representing nonlinear effects while maintaining interpretability, the Volterra series provides a structured approach to modelling real-world systems. However, the complexity of the Volterra model grows rapidly with the degree of nonlinearity and memory length, making it computationally expensive for high-dimensional problems.

1.4 Motivation and Research Objectives

Nonlinear dynamical systems are prevalent in various real-world applications, including control systems, biomedical engineering, and signal processing. While linear models provide a well-established framework for system identification, they fail to capture

the complex input-output relationships in nonlinear systems. Block-oriented models, which combine linear dynamic blocks with static nonlinear functions, have gained significant attention due to their interpretability and applicability among nonlinear models. However, their identification remains a challenging problem.

This thesis explores a novel approach to identifying block-oriented models using the Volterra series, which also enables the identification of the NLSS form, bridging the gap between these different modelling paradigms. Block-oriented models are widely used in control, communications, and biomedical signal processing because they can approximate a broad class of nonlinear behaviours while remaining computationally efficient. Identifying block-oriented models is crucial because they offer physical interpretability (the linear and nonlinear components often have direct system-level meaning) and are computationally more efficient and structured than fully general nonlinear models.

However, the identification of block-oriented models is particularly challenging due to:

- **Model structure and hyperparameter selection:** Selecting the appropriate configuration of linear and nonlinear blocks (e.g. Hammerstein, Wiener, or WH), determining the correct system order, and addressing structural identifiability issues can be inherently difficult and time-consuming. In this thesis, it is assumed that the block-oriented structure and model orders (memory lengths of linear blocks and polynomial degrees of nonlinearities) are known a *priori*.
- **Dynamic and static effects separation:** Distinguishing between the system's static nonlinear behaviour and its dynamic linear response is often complicated, especially in the presence of complex nonlinearities and high levels of output noise.
- **Non-convex optimisation:** The identification process involves solving non-convex optimisation problems, leading to convergence to local minima and requiring good initial estimates.
- **Sensitivity to noise:** The accuracy of the predicted output and the identified static and dynamic models can degrade significantly in the presence of output noise.
- **Sensitivity to excitation:** Poor excitation signals fail to sufficiently excite the relevant dynamics or nonlinearities. Therefore, some block-oriented models have a restrictive assumption on the excitation. In this work, the minimum requirement of the input signal is to be sufficiently (persistently) excited.

While block-oriented models offer a structured approach to nonlinear system identification, their representation in state-space form allows for a more general and

flexible characterisation of nonlinear dynamics. The PNLSS model extends classical state-space representations by incorporating polynomial nonlinearities into the state evolution and output equations. The advantages of using a PNLSS formulation include its compactness (a concise mathematical description of the system) and its applicability to a broader class of nonlinear systems (including those not strictly conforming to block-oriented structures). However, while the PNLSS model offers a more structured representation compared to the Volterra model, identifying its parameters directly from input-output data remains challenging. Although the parameter space is typically lower-dimensional than that of the Volterra model, the optimisation problem is still high-dimensional, non-convex, and prone to convergence to local minima, making reliable parameter estimation difficult in practice. A good initialisation is critical to achieving reliable parameter estimates, motivating the need for systematic approaches to obtain an initial guess.

The Volterra series provides a general mathematical framework for representing nonlinear systems as a series of polynomial convolution terms. While computationally expensive for direct use, it is a powerful intermediate representation for identifying structured nonlinear models like block-oriented and PNLSS models (see Figure 1.6). The key reasons for using the Volterra series as an intermediate step in the estimation of block-oriented and PNLSS models are as follows:

- **Universality:** The Volterra series can approximate fading memory systems, which makes it a strong mathematical tool for extracting essential characteristics of the systems.
- **Linear-in-the-parameter characteristic:** The Volterra kernel estimation problem can be written in the form of the linear regression problem, which makes the estimation simple yet powerful.
- **Structural Richness:** The Volterra series captures nonlinear system dynamics through structured kernels of different degrees and memory depths. While direct interpretability of higher-order kernels is often limited, their organised structure can still provide insights into dominant nonlinear effects and temporal interactions when visualised or sparsely parametrised.
- **Non-parametric model:** The Volterra model offers a flexible input-output representation that does not require a *priori* assumptions about the internal structure of the system, making it a flexible and unbiased intermediate representation.

By using the Volterra series-based parameter estimation as an intermediate step, this thesis proposes a novel methodology to establish links between these nonlinear model classes, improving identification accuracy and robustness. A key aspect of this methodology is the use of prior structural knowledge about the underlying block-oriented

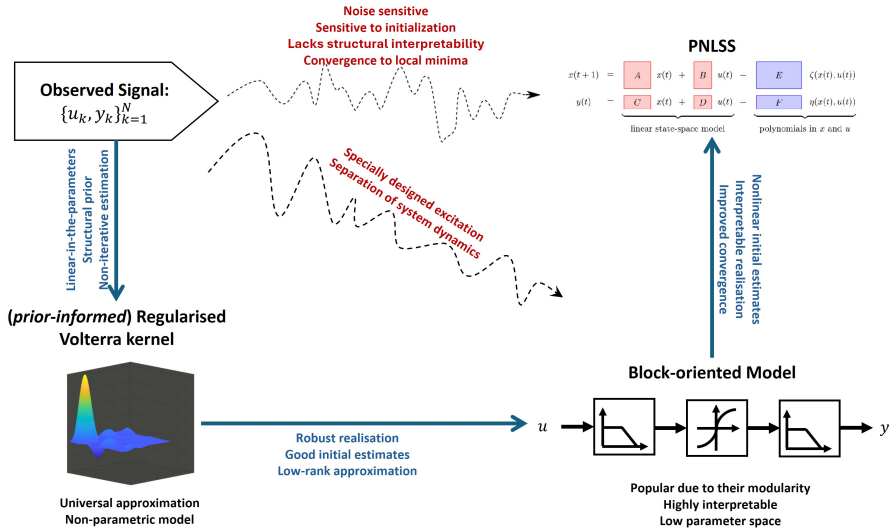


Figure 1.6: Proposed structured pathway for nonlinear system identification: From observed input-output signals to interpretable and robust models via Volterra, block-oriented, and PNLSS representations. The observed discrete-time input–output signal data serves as the starting point. The direct identification paths toward block-oriented models and PNLSS models (dashed arrows) are challenging and sensitive to noise, initialisation, and local minima (drawbacks highlighted in red). The proposed sequential path (solid blue arrows) proceeds from the input–output data to the Volterra model (leveraging linear-in-parameters estimation with structural priors), then to block-structured models (for modularity and interoperability), and finally to PNLSS models (initialised with robust and physically meaningful priors).

model to guide the estimation of the Volterra kernels. By embedding this structural information into the estimation process, the resulting Volterra kernel, referred to in this work as *prior-informed* Volterra kernel, reflects the system's known modularity and nonlinear characteristics. This enhanced kernel, informed by the system's structure, forms the basis for improved model transformation and initialisation across nonlinear model classes.

Based on the motivation and challenges outlined above, this thesis pursues a set of interconnected objectives aimed at advancing the identification of nonlinear dynamical systems through Volterra-based and block-oriented modelling frameworks. The work also includes an auxiliary contribution addressing a fundamental problem in linear system identification. The **primary research objectives and resulting contributions** of this thesis are summarised below.

1. **Estimation of systems with repeated poles:** A dedicated method is proposed for identifying discrete-time LTI systems that exhibit repeated poles (i.e., pole multiplicities greater than one). Traditional subspace-based identification methods often struggle to accurately recover such systems. The proposed approach improves numerical robustness and ensures consistent estimation of pole locations and multiplicities, providing reliable information about the underlying linear subsystem that is essential for careful model development.

2. **Volterra-based modelling of nonlinear systems:** A unified Volterra series-based framework is developed for identifying and refining parameters in nonlinear block-oriented models, including Hammerstein, Wiener, and WH systems. This framework enables:

    - Systematic extraction of Volterra kernels corresponding to each block-oriented structure.
    - Estimation of *prior-informed* (regularised) Volterra kernels using available knowledge of the block configuration.
    - Reconstruction and refinement of block-oriented model parameters through shape-constrained optimisation to improve accuracy and robustness.

3. **Extension to PNLSS modelling:** The Volterra-based parameter estimates are transformed into structured and interpretable initial estimates for PNLSS models. This provides a systematic link between block-oriented and PNLSS frameworks, improving convergence, reducing model complexity, and enhancing interpretability.

4. **Validation through simulations and experimental data:** The proposed methodologies are validated through extensive numerical simulations, measurement data, and benchmark datasets from the system identification community. Each stage of the modelling pipeline - Volterra kernel estimation, block-oriented model optimisation, and PNLSS initialisation - is implemented and tested for all three block-oriented structures (Hammerstein, Wiener, and WH), with appropriate methodological adaptations for each case.

Overall, this research establishes a unified and systematic methodology that leverages the Volterra series framework to enhance both block-oriented and nonlinear state-space model identification. The outcomes have significant implications for system modelling, control design, fault diagnosis, and signal processing, offering a more robust and interpretable foundation for nonlinear system identification.

In this thesis, while the focus remains on interpretable nonlinear models such as the Volterra series, block-oriented, and PNLSS models, neural networks serve as an important reference class of data-driven methods. Their performance and architectural

diversity are briefly discussed in the concluding section to contextualise the presented modelling framework within the broader landscape of nonlinear system identification.

1.5 Structure of the Thesis

This thesis is organised to guide the reader from the fundamentals of system identification to advanced kernel-based techniques for nonlinear models. The chapters are structured to gradually build conceptual depth and technical sophistication, with a focus on Volterra-based identification of block-oriented systems and their transformation into PNLSS models. The flow and interconnection of the chapters are illustrated in Figure 1.7.

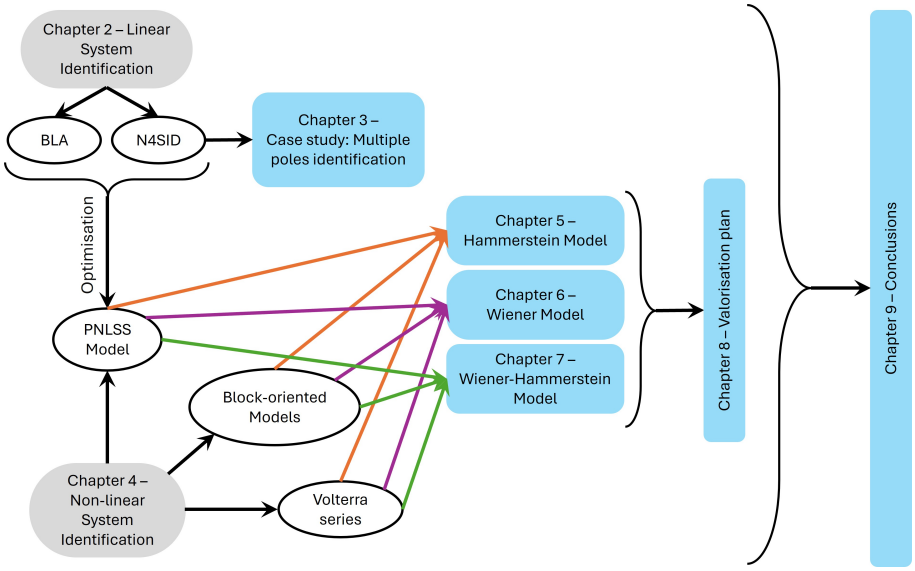


Figure 1.7: A schematic overview of the thesis structure, illustrating the progression from linear system identification to nonlinear kernel-based modelling of three block-oriented systems and their PNLSS realisations. The chapters highlighted in blue represent the original contributions of this thesis.

Chapter 2 lays the theoretical foundation for linear system identification, covering classical techniques, excitation signal classes, and the concept of the BLA. It also details subspace-based state-space identification methods, providing the necessary background for later extensions to nonlinear systems.

A focused case study is presented in Chapter 3, where the problem of estimating systems with repeated poles is explored using advanced signal models and matrix formulations. Simulation results validate the approach and highlight its robustness to noise.

The discussion transitions to nonlinear models in Chapter 4, which introduces block-oriented structures (Hammerstein, Wiener, WH) and the PNLSS framework. The chapter also surveys the Volterra series and establishes the challenges in identifying block components from input–output data.

Chapters 5 through 7 each focus on a specific block-oriented model: Hammerstein, Wiener, and Wiener–Hammerstein. These chapters follow a consistent methodology: constructing a structured Volterra kernel, extracting the model parameters, applying shape-enforced optimisation, and realising a PNLSS representation. Tensor-based techniques are introduced in the Wiener case to manage the complexity of symmetric high-order kernels.

In Chapter 8, the practical dimension of this research is addressed. It outlines the dissemination plan for software and data, and proposes a modular software framework to facilitate reproducibility and extension by the research community.

The thesis concludes with Chapter 9, which synthesises the contributions, reflects on the research questions, and outlines future directions, particularly for extending the presented framework to more general nonlinear systems and higher-dimensional tensor formulations.

Chapter 2

Principles and Techniques of Linear System Identification

Abstract: The chapter begins with classical estimation techniques, such as least-squares estimation and their role in capturing linear system dynamics. The Best Linear Approximation (BLA) is discussed in detail, including its definition, properties, and estimation methods for both periodic and non-periodic excitation. Emphasis is placed on how the spectral characteristics of the excitation signal influence the resulting linear model. The chapter then transitions to other subspace-based identification techniques, covering the theoretical foundations and algorithms behind methods such as Multivariable Output Error State-Space (MOESP) and Numerical Algorithms for Subspace State-Space (N4SID) system identification. Practical considerations, including MATLAB implementation, are also discussed to provide a comprehensive introduction to parametric state-space modelling.

Note on Chapter Content: This chapter provides a foundational overview of linear system identification methods that serve as baseline models for comparison with more advanced identification techniques discussed in later chapters of this thesis. While the content presented here is not an original contribution, it is compiled and explained based on existing literature, with interpretations and illustrative examples drawn from the author's understanding.

2.1 Foundation to Linear Systems

Definition 2.1.1 (Linear system)**.** Systems that satisfy the principles of homogeneity and additivity (together called the superposition principle) at the level of input and output signals are called linear systems.

Let $S\{\cdot\}$ denote the linear operator that maps an input signal $\{u_k\}_{k\in\mathbb{Z}}$ to an output signal $\{y_k\}_{k\in\mathbb{Z}}$ in discrete-time.

Homogeneity: For any scalar $\gamma \in \mathbb{C}$ and input signal u_k, if $y_k = S\{u_k\}$, then

$$S\{\gamma u_k\} = \gamma S\{u_k\} = \gamma y_k. \tag{2.1}$$

Additivity: For any two input signals $u_k^{(1)}, u_k^{(2)}$, with outputs

$$y_k^{(1)} = S\{u_k^{(1)}\}, \quad y_k^{(2)} = S\{u_k^{(2)}\},$$

the system satisfies:

$$S\{u_k^{(1)} + u_k^{(2)}\} = S\{u_k^{(1)}\} + S\{u_k^{(2)}\} = y_k^{(1)} + y_k^{(2)}, \tag{2.2}$$

Superposition principle: Combining the above two properties, for scalars $\gamma_1, \gamma_2 \in \mathbb{C}$:

$$S\{\gamma_1 u_k^{(1)} + \gamma_2 u_k^{(2)}\} = \gamma_1 S\{u_k^{(1)}\} + \gamma_2 S\{u_k^{(2)}\} = \gamma_1 y_k^{(1)} + \gamma_2 y_k^{(2)}. \tag{2.3}$$

Definition 2.1.2 (Time-Invariant system)**.** A system is said to be Time-Invariant (TI) if its input-output behaviour does not change over time. That is, the input delayed by $\tau \in \mathbb{Z}$ results in the output delayed by the same τ. If $y_k = S\{u_k\}$, then for the delayed input $u_{k-\tau}$, the output is

$$S\{u_{k-\tau}\} = y_{k-\tau}, \quad \forall k, \tau \in \mathbb{Z}.$$

A system that is both linear (Definition 2.1.1) and time-invariant (Definition 2.1.2) is called a Linear Time-Invariant (LTI) system. The LTI property is used to capture the linear dynamic behaviour of the block-oriented models, introduced in Section 1.3.1.

Definition 2.1.3 (Causal system)**.** A system is said to be causal if its output at any time k depends only on the past and current input values, but not on future inputs. Formally, a system is causal if the output y_k depends only on inputs $u_{k-i}, \forall i \geq 0$.

This property, known as causality, is essential for modelling physical systems, as real-world systems cannot respond to future inputs. For instance, the following LTI system is causal:

$$y_k = \sum_{i=0}^{\infty} h_i u_{k-i}, \tag{2.4}$$

where the impulse response coefficients h_i are zero for $i < 0$. Similarly, a simple moving average filter, given by:

$$y_k = \frac{1}{N} \sum_{i=0}^{N-1} u_{k-i}, \tag{2.5}$$

is also causal, since it averages only the current and past inputs.

Using the measured input-output data, a mathematical model can be developed to represent the unknown LTI system. The idea is to estimate a model that describes the system's dynamic behaviour and can be generalised for its use in prediction and control. Some systems, like RLC circuits and mass-spring-damper systems, can be approximated as an LTI system within a certain operating range. Assuming that the data samples are collected as explained in Section 1.2.1, we proceed with the model selection step in Section 1.2.2.

2.2 Classical Methods

LTI systems are commonly modelled using linear-in-the-parameters models, which can be efficiently estimated using linear least-squares (LS) techniques, due to their simplicity and computational efficiency. In the LS approach, the parameter estimates are obtained by minimising the mean square error (MSE) between the true and the predicted outputs. Given a set of $N \in \mathbb{Z}$ data samples, the MSE between the true output $\boldsymbol{y}$ and predicted output $\hat{\boldsymbol{y}}$ is given by:

$$\text{MSE} = \frac{1}{N} \sum_{k=1}^{N} (y_k - \hat{y}_k)^2. \tag{2.6}$$

For a linear-in-the-parameters model of order $M \in \mathbb{Z}$, the measured output $\boldsymbol{y}$ at time k can be approximated as follows:

$$y_k \approx \alpha + \sum_{i=0}^{M} \beta_i u_{k-i}, \tag{2.7}$$

where $\alpha, \beta_0, \ldots, \beta_M \in \mathbb{R}$ are the unknown coefficients. In matrix form, it is written as

$$\boldsymbol{y} = \boldsymbol{\Phi}\,\boldsymbol{\theta} + \boldsymbol{e}, \tag{2.8}$$

where $\boldsymbol{\Phi}$ is the design matrix of size $N \times (M+2)$, $\boldsymbol{\theta}$ is the coefficient vector of size $(M+2)$, and $\boldsymbol{y}, \boldsymbol{e} \in \mathbb{R}^N$ are the output and error vectors, respectively. Since $\boldsymbol{e}$ is unknown, the vector $\boldsymbol{\theta}$ is estimated by minimising the MSE. The parameter estimates are given by:

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{y}. \tag{2.9}$$

This is commonly known as linear least-squares or ordinary least-squares, as the assumed function is linear in the unknown coefficients. Additionally, if the measurement error in the output does not have constant variance across all inputs, then the weighted LS approach can be used; however, this is not considered in this thesis.

Another popular framework to represent linear systems is the LSS model (1.14) due to its ability to break down a single higher-order differential/difference equation into multiple first-order equations using state variables. Consider the second-order differential equation of mass-spring-damper system (1.4) example, with force input $u(t)$ and displacement output $y(t)$,

$$m\ddot{y}(t) + c\dot{y}(t) + ky(t) = u(t), \tag{2.10}$$

where $\dot{y}(t), \ddot{y}(t)$ are velocity and acceleration. The spring stiffness k and damping coefficient c are the material properties, and the mass m is the inertial property of the object. Although the example is presented in continuous-time to illustrate the physical intuition behind the state-space model structure, no continuous-to-discrete transformation is performed in this thesis. All modelling and system identification tasks are conducted directly in discrete time.

The idea of state variables can be introduced to rewrite the model in LSS form as two first-order equations. Let $x_1 = y$ and $x_2 = \dot{y}$, then

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -\frac{k}{m}x_1 - \frac{c}{m}x_2 + \frac{1}{m}u. \end{aligned} \tag{2.11}$$

This is now a first-order system with state variables x_1 and x_2, that can be written in LSS form as follows:

$$\begin{aligned}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} &= \begin{bmatrix}0 & 1\\ -\frac{k}{m} & -\frac{c}{m}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{m}\end{bmatrix}u,\\ y &= \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix},\end{aligned} \tag{2.12}$$

where the state vector $\boldsymbol{x} = \begin{bmatrix}x_1 & x_2\end{bmatrix}^\top$ describes the system.

Further in this chapter, we will discuss the LSS form obtained from the input-output data using BLA and other subspace-based methods. This preliminary knowledge is essential before we extend the state-space model to accommodate nonlinearities in the system.

2.3 Overview of Best Linear Approximation

Definition 2.3.1 (Best Linear Approximation)**.** The Best Linear Approximation (BLA) is defined as the model G included in the linear models set $\mathcal{G}$, such that

$$G_{\mathrm{BLA}} = \arg\min_{G\in\mathcal{G}} \mathbb{E}\{|\tilde{y}_k - G(q)\,\tilde{u}_k|^2\}, \tag{2.13}$$

where $\tilde{u}_k = u_k - \mathbb{E}\{u_k\}$ and $\tilde{y}_k = y_k - \mathbb{E}\{y_k\}$ denote the zero-mean (de-trended) input and output signals, respectively. Here, $\mathbb{E}\{\cdot\}$ denotes the expectation operator, taken with respect to the joint distribution (or equivalently, the time average under stationarity) of the stochastic input and output signals, and $G(q)$ is the linear transfer function with shift operator q, i.e., $q^{-1}u_k = u_{k-1}$ (Pintelon and Schoukens, 2012; Enqvist, 2005; Enqvist and Ljung, 2005).

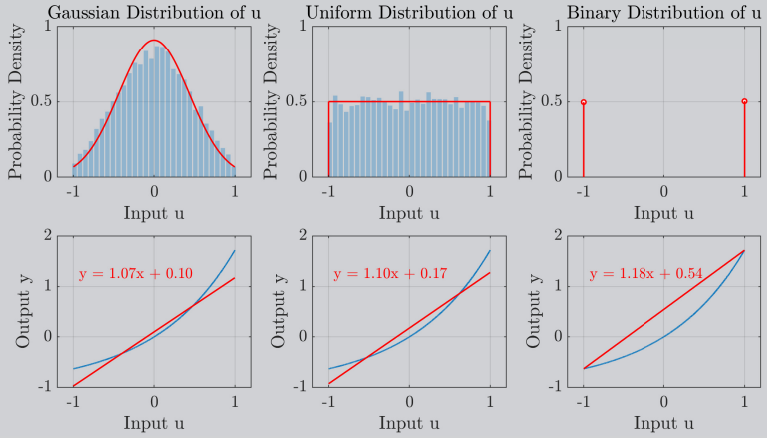
The BLA can also be obtained non-parametrically using the Frequency Response Function (FRF) of the data as follows:

$$G_{\mathrm{BLA}}(j\omega_i) = \frac{S_{yu}(j\omega_i)}{S_{uu}(j\omega_i)}, \tag{2.14}$$

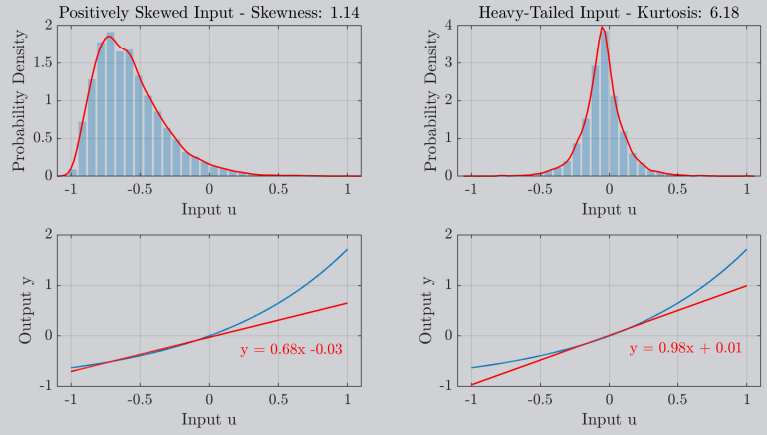
where $S_{yu}(j\omega_i)$ and $S_{uu}(j\omega_i)$ are power spectra from cross-correlation of output and input, and autocorrelation of input, respectively. Here, $\omega_i = 2\pi\frac{i}{N}$ denotes the

angular frequency corresponding to the ith frequency bin in the Discrete Fourier Transform (DFT), with $i = 0, 1, \ldots, N-1$, and N the total number of time samples. This is the classical Wiener-Hopf equation in the frequency domain. The amplitude distribution and frequency content of the input signal influence the BLA of a nonlinear dynamic system (Enqvist, 2005; Enqvist and Ljung, 2005; Pintelon and Schoukens, 2012).

Example 1. The influence of the amplitude distribution is shown in Figure 2.1 with the help of a small simulation example, where the true output $y = e^u - 1$ is a static nonlinear function. The system is excited by stochastic inputs with different probability density functions (PDF): Gaussian, uniform, and binary. As shown in Figure 2.1, each input distribution results in a different linear approximation of the nonlinear output. The BLA minimises the MSE between the true output and its linear approximation, which depends on how the input samples are distributed across the nonlinear curve. For instance, the Gaussian input concentrates more samples near the origin, leading to a good local linear fit around zero, while for the binary sequence, the fit is better at these sample points, and the uniformly distributed input spreads fit across the input range. Additionally, higher-order moments of the input distribution, such as skewness and kurtosis, influence how strongly different regions of the nonlinearity are emphasised during excitation, thereby affecting the BLA.



(a) Amplitude dependency



(b) Higher-order Moments dependency

Figure 2.1: Effect of input probability density functions (PDF) on the linear approximation of a nonlinear function: Gaussian input gives a local fit near zero (unless skewed). Uniform input spreads fit across the range, and Binary input captures edge behaviour.

Although the frequency content of the input signal also plays a critical role in shaping the BLA, particularly in dynamical systems, this effect cannot be illustrated using a purely static nonlinearity. The following example illustrates this dependence in a simple dynamical nonlinear system.

Example 2. The true system output at time k is modelled as: $y_k = \theta_0 u_k + \theta_1 u_{k-1}^2$, where $\theta_0 = 1, \theta_1 = 0.1$. This represents a Nonlinear Finite Impulse Response (NLFIR) system with both a linear and a nonlinear term. The system is excited using three different types of stochastic inputs with distinct power spectra: a sinusoid, band-limited noise, and white Gaussian noise (WGN) (see Definition 2.3.3). Figure 2.2 shows the input signals in the time domain (top row), their corresponding power spectral densities (second row), and the true output and the prediction error power spectra (bottom row). In the bottom row, the power spectrum of the true system output is shown in blue, while the error spectrum of the BLA-predicted output, i.e., the difference between the true and linearly approximated outputs with 40 taps, is shown in orange.

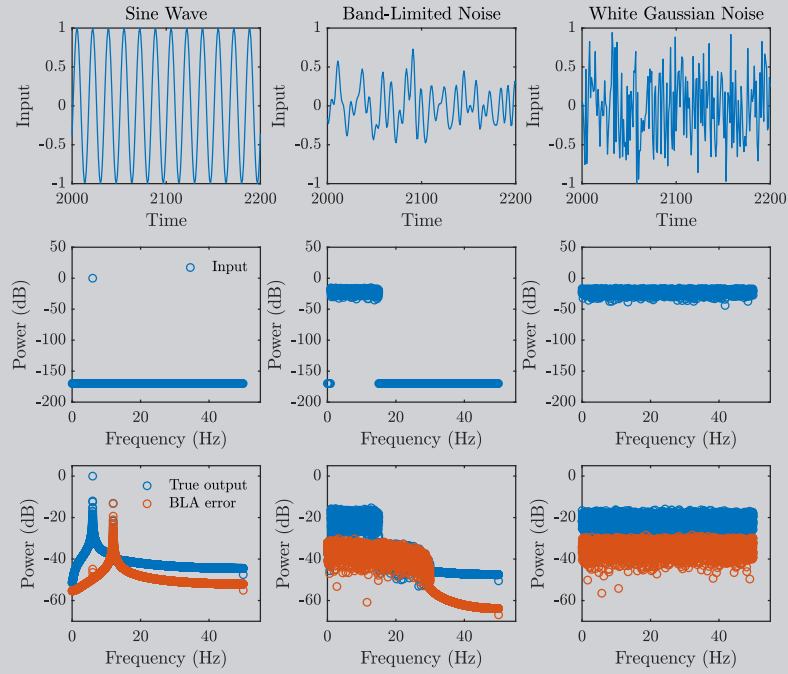


Figure 2.2: Effect of input frequency content on the approximation of FIR for a NLFIR system: sinusoidal input yields localised harmonics errors, Band noise causes in-band distortion, WGN spreads nonlinear error across the spectrum.

For the sinusoidal input, the true output spectrum shows energy concentrated at the fundamental frequency and its harmonics due to the nonlinear term, while the BLA

error spectrum is high outside the excitation frequency since the linear model cannot capture these harmonics. With band-limited noise, the output spectrum spreads within and beyond the excited band due to nonlinear interactions (the nonlinear term causes frequency mixing), and the BLA error is significant outside the excited band. WGN excites a broad range of frequencies, enabling the BLA to approximate the linear dynamics over the entire spectrum, leading to a more uniformly distributed error. This illustrates that the accuracy and interpretability of the BLA are strongly influenced by how well the input excitation covers the relevant frequency content of the system.

2.3.1 Class of Excitation Signals

The input signals used in this thesis belong to the class of Gaussian signals, which have a user-defined power spectrum. Additionally, the signals are assumed to be stationary, ensuring a well-defined power spectrum. Gaussian noise, Gaussian periodic noise, and random phase multisines are the excitation signals belonging to this class. However, Gaussian periodic noise and random phase multisines only truly belong to this class in the limit of an infinite number of excited frequency components.

Definition 2.3.2 (Gaussian noise)**.** Gaussian noise is a stochastic signal in which each sample is drawn from a Gaussian (normal) distribution, typically with zero mean and a specified standard deviation.

Definition 2.3.3 (White Gaussian noise)**.** White Gaussian noise is a special case of Gaussian noise where the samples are Gaussian, independent, and identically distributed (i.i.d.) over time.

This independence results in a flat power spectral density across all frequencies, i.e., the noise contains equal power at all frequency components.

Example 3. Consider a random sequence of $N = 10{,}000$ samples drawn from a Gaussian distribution, filtered with a 6th-order Chebyshev type I filter (cut-off frequency of 0.25 Hz, assuming sampling frequency of 1 Hz and 0.5 dB passband ripple). The filtered ('un-whitened') signal is normalised to have a Root Mean Square (RMS) value of one. Filtering the random Gaussian noise sequence serves to limit and shape its frequency content. The unfiltered random sequence is WGN, i.e., noise with a flat power

spectral density (PSD) that theoretically contains energy at all frequencies. However, in practical applications and experiments, we are often only interested in a specific frequency band. From the frequency content in Figure 2.3, we can see that filtering WGN to confine its energy to a specific frequency band results in a spectrum that is no longer perfectly flat. These irregularities arise from the finite length of the data samples, the inherent randomness of the noise, and the non-ideal characteristics of the filter (such as passband ripple).

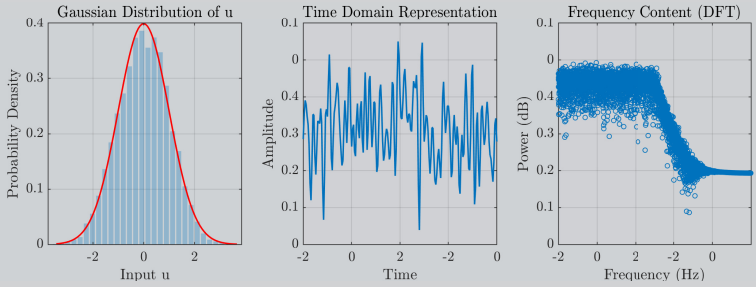


Figure 2.3: A filtered Gaussian noise input signal in the time domain and frequency domain, along with its distribution.

Consequently, the signal-to-noise ratio (SNR) varies across frequencies. However, Gaussian noise is widely used in system identification because of its statistical properties and ability to excite all system modes, providing a comprehensive test of system behaviour within the targeted frequency band. Additionally, the ability to tailor the PSD via filtering allows us to mimic real-world conditions where the noise is inherently band-limited.

Definition 2.3.4 (Random phase multisine)**.** A random phase multisine is a periodic signal constructed as the sum of several sine wave components whose frequencies are harmonically related:

$$u(t) = \frac{1}{\sqrt{N}} \sum_{i=-N}^{N} U_i \, e^{j\left(2\pi f_{\max} \frac{i}{N} t + \phi_i\right)}, \tag{2.15}$$

where $t \in \mathbb{R}$ is the continuous-time variable, N is the number of frequency components, $f_{\max}$ is the maximum excitation frequency, U_i are the complex amplitudes (which can be chosen based on the user-defined power spectrum), and ϕ_i are independent random phases, typically uniformly distributed in $[0, 2\pi)$, such that $\mathbb{E}\{e^{j\phi_i}\} = 0$ (Schoukens et al., 2009a; Pintelon and Schoukens, 2012). For $u(t) \in \mathbb{R}$, we have that $\phi_{-i} = -\phi_i$.

For discrete-time signals, the equivalent formulation of the random phase multisine is given by:

$$u_k = \frac{1}{\sqrt{N}} \sum_{i=-N}^{N} U_i \, e^{j\left(2\pi \frac{i}{N} k + \phi_i\right)}, \tag{2.16}$$

where $k \in \mathbb{Z}$ is the discrete-time index and i indexes the frequency components.

The normalisation factor $1/\sqrt{N}$ ensures a constant RMS value, and the power of the multisine remains finite as N increases. Symmetry in the frequency domain is ensured by assuming $U_{-i} = U_i = U(f_{\max} \frac{i}{N})$. Theoretically, for a random multisine to be a Gaussian-distributed time-domain signal, $N \to \infty$, but in practice, $N > 20$ excited lines are already enough for smoothly varying amplitude distributions (Pintelon and Schoukens, 2012; Schoukens et al., 2005).

Example 4. Consider a random phase multisine of $N = 10{,}000$ samples, with flat power spectrum (constant U_i) and $f_{\max} = 0.25$ Hz. The amplitude of the random phase multisine is normalised to an RMS value of one, to compare with the Gaussian noise example above. From the time domain representation in Figure 2.4, it is hard to distinguish between the random phase multisine and Gaussian noise; however, from the frequency content plot, it is clear that the random phase multisine input has a flat power spectrum within the fixed band limit.

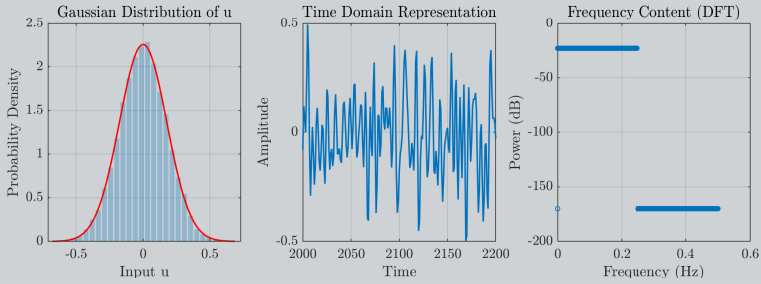


Figure 2.4: A random phase multisine input signal in the time domain and frequency domain, along with its distribution.

With the help of a user-defined flat power spectrum, SNR variations across all excited frequencies can be avoided when using a Random phase multisine as the input signal. It can be further customised to understand the nonlinear behaviour of the SUS. The

periodicity in the multisines enables averaging the measurement noise, thereby clearly distinguishing measurement noise from nonlinear distortions.

This research uses two types of random phase multisine signals: 'full' and 'odd'. The multisine 'full' refers to the excited frequency lines, i.e., all the frequency lines in the band limit are excited. The multisine 'odd' denotes that only the odd frequency lines are excited, which is useful in detecting even nonlinearities. For the even nonlinearities in the system, the 'odd' multisine excitation gives a contribution only at the even (non-excited) frequencies.

2.3.2 Basics of BLA

The BLA of a nonlinear system exists under the following conditions (Schetzen, 1980):

- The system is bounded-input, bounded-output (BIBO) stable.
- The current output of the system depends on a finite number of past inputs.
- The system is causal and time-invariant.

Any system that satisfies these conditions is called a Wiener system (Schetzen, 1980) (not to be confused with a Wiener block-oriented system in Section 1.3.1). The response of these systems to a harmonic input is harmonic, with the same period as the input. Hence, they are also referred to as Periodic Input, Same Periodic Output (PISPO) systems. Some nonlinear phenomena, including sub-harmonics, chaos, and hysteresis, cannot be approximated using the Wiener class of systems.

A nonlinear system can be represented using four components: excitation u, best linear approximation G_{BLA}, an unmodelled nonlinear source y_s of the system, and output y (Enqvist and Ljung, 2005; Schoukens et al., 2005), as depicted in Figure 2.5 for a SISO system. Here, y_s is uncorrelated with the excitation u, though it is dependent on it. In practice, an additional noise source y_n is added to account for the measurement noise. The output y in the frequency domain is given as

$$Y(j\omega_i) = G_{\mathrm{BLA}}(j\omega_i)\, U(j\omega_i) + Y_s(j\omega_i). \tag{2.17}$$

The unmodelled nonlinear contribution $Y_s(j\omega_i)$ is also called a nonlinear noise source, because if the input is stochastic, each realisation can be different, making the nonlinear contributions also stochastic. This makes it difficult to distinguish nonlinear contributions from other noise sources.

In general, $G_{\mathrm{BLA}}(j\omega_i)$ is estimated non-parametrically using multiple input realisations and averaging the measured FRF as follows:

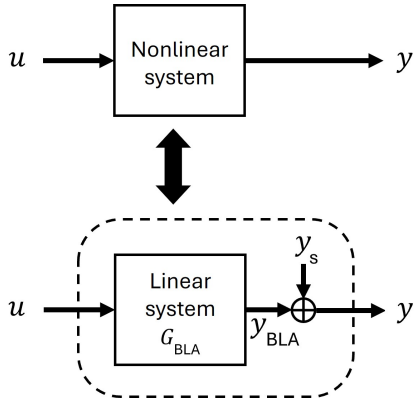


Figure 2.5: A SISO nonlinear system and its equivalent BLA representation.

1. Excite the system with multiple realisations of a stochastic input, where each realisation $u_m(t)$ for $m = 1, \ldots, M$, should be designed to have the same PSD, ensuring consistent input energy spread across frequencies.
2. Measure the output $y_m(t)$, for each input realisation $u_m(t)$.
3. Compute the FRF for each realisation:

$$G_m(j\omega_i) = \frac{Y_m(j\omega_i)}{U_m(j\omega_i)}, \tag{2.18}$$

   where $G_m(j\omega_i)$ is an estimate of system response for the mth realisation.
4. The $G_{\text{BLA}}(j\omega_i)$ is then obtained by averaging $G_m(j\omega_i)$ from M experiments. This procedure helps to reduce the effect of $Y_s(j\omega_i)$, although it varies with each realisation, but the BLA is consistent.

2.3.3 Properties of BLA

Robustness: In Example 1 and Example 2, it was illustrated that the linear approximation of a system is affected by the characteristics of excitation. However, BLA remains the same for all excitations within the same class of signals, provided they have the same power spectrum (Schoukens et al., 2009a). In this case, the relevant class is Gaussian with a fixed power spectrum (e.g., excitations mentioned in Section 2.3.1, Gaussian noise, and random phase multisine). This is proved in Schoukens et al. (2009a) using Riemann integral arguments that when the discrete power spectrum of a periodic signal, like a random phase multisine, asymptotically approaches the continuous spectrum of a non-periodic signal (Gaussian noise), they result in the same BLA.

Even nonlinearities do not contribute to BLA: Even nonlinearities (like u^2) create output frequency components at frequencies different from those of the input. These new components do not align with the fundamental frequency components of the input. Since they are stochastic, they disappear when averaged over multiple realisations of the FRFs. Therefore, their impact on FRF measurement is similar to random noise. BLA is thus unaffected by even nonlinearities, except for an increase in variance (i.e., more uncertainty in measurements). A random phase multisine contains frequency components at both odd and even frequencies. Hence, exciting a system with an odd multisine results in generating frequency content at even frequencies by even nonlinearities. Since the system's response at odd frequencies is recorded, the effect of even nonlinearities is no longer present in the FRF measurement (Pintelon and Schoukens, 2012; Schoukens et al., 2012). Consequently, the variance of BLA is reduced. However, the mean of it remains unchanged (Schoukens et al., 2005).

2.3.4 Non-parametric Estimation of the BLA for Periodic Data

The estimation procedure differs based on the excitation applied to the system. This thesis focuses on SISO systems. The extension to Multiple Inputs, Multiple Outputs (MIMO) systems is given in Paduart (2008).

To obtain the BLA for periodic excitation, the measurement procedure given in Figure 2.6 is adopted (D'haene et al., 2005). This distinguishes nonlinear distortions from the

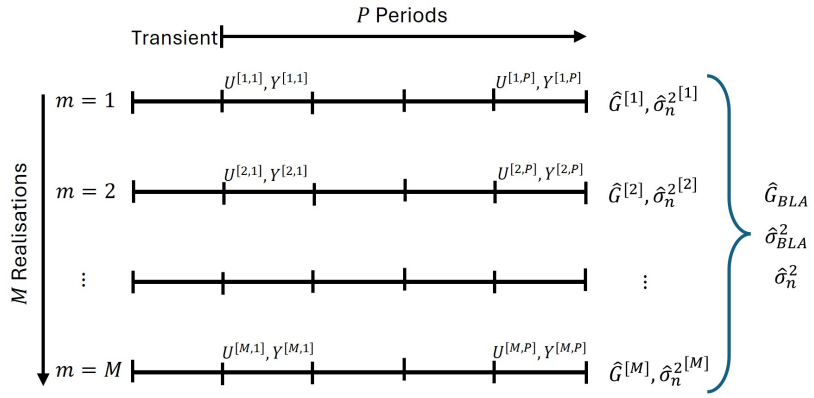


Figure 2.6: Design of experiments to estimate BLA for SISO periodic excitation. P steady-state periods of M realisations of excitation are measured.

measurement noise on the BLA due to the periodic data. In general, M realisations of random phase multisine are applied to the SUS, as shown on the vertical axis in

Figure 2.6. Each realisation consists of P steady-state periods on the horizontal axis, and the recording is started after some transient fade-out. The variations over the P periods are caused by measurement noise, and variations over M realisations are due to the combined effect of measurement noise and the stochastic nonlinear behaviour (Lauwers, 2011).

For the p-th period in realisation m, the input spectra $U(i)^{[m,p]}$ and output spectra $Y(i)^{[m,p]}$ are calculated using the DFT, where i is the frequency index. The averages of spectra over P periods, for the mth realisation is denoted as $\hat{U}(i)^{[m]}$ and $\hat{Y}(i)^{[m]}$. Thus, the FRF is calculated as

$$\hat{G}(i)^{[m]} = \frac{\hat{Y}(i)^{[m]}}{\hat{U}(i)^{[m]}}. \tag{2.19}$$

Note: $U(i)$ and $Y(i)$ denote the DFT coefficients of the input and output signals at the ith frequency bin, corresponding to angular frequency $\omega_i = 2\pi i/N$. These are equivalent to $U(j\omega_i)$ and $Y(j\omega_i)$, respectively. The simplified notation $U(i)$ is adopted for compactness when working with discrete Fourier-domain quantities.

The above equation is equivalent to (2.14) for periodic excitations. The variance $\hat{\sigma}^2_{\text{n}}(i)^{[m]}$ of $\hat{G}(i)^{[m]}$ is then given by

$$\hat{\sigma}^2_{\text{n}}(i)^{[m]} = \frac{1}{P(P-1)} \sum_{p=1}^{P} |\hat{G}(i)^{[m,p]} - \hat{G}(i)^{[m]}|^2, \tag{2.20}$$

where $\hat{G}(i)^{[m,p]}$ is calculated using the FRF of period p as below:

$$\hat{G}(i)^{[m,p]} = \frac{Y(i)^{[m,p]}}{U(i)^{[m,p]}}. \tag{2.21}$$

The BLA is obtained by averaging the FRF estimates in (2.19) over the M realisations:

$$\hat{G}_{\text{BLA}}(i) = \frac{1}{M} \sum_{m=1}^{M} \hat{G}(i)^{[m]}. \tag{2.22}$$

Therefore, the variance $\hat{\sigma}^2_{\text{n}}(i)^{[m]}$ is also averaged by considering the $\sqrt{N}$-law to represent the effect of measurement noise $\hat{\sigma}^2_{\text{n}}(i)$ as below:

$$\hat{\sigma}^2_{\text{n}}(i) = \frac{1}{M^2} \sum_{m=1}^{M} \hat{\sigma}^2_{\text{n}}(i)^{[m]}. \tag{2.23}$$

The variance $\hat{\sigma}^2_{\text{BLA}}$ on $\hat{G}_{\text{BLA}}(i)$ is given by

$$\hat{\sigma}^2_{\text{BLA}} = \frac{1}{M(M-1)} \sum_{m=1}^{M} |\hat{G}(i)^{[m]} - \hat{G}_{\text{BLA}}(i)|^2, \tag{2.24}$$

denoting the combined effect of measurement noise and stochastic nonlinear behaviour:

$$\sigma^2_{\mathrm{BLA}}(i) = \sigma^2_{\mathrm{n}}(i) + \sigma^2_{\mathrm{nl}}(i). \tag{2.25}$$

Thus, the variance due to the stochastic nonlinearity σ^2_{nl} can be estimated based on the estimated variances σ^2_{BLA} and σ^2_{n}.

2.3.5 Non-parametric Estimation of the BLA for Non-periodic Data

Exciting a system with a periodic signal enables the separation of linear response, nonlinear distortions, and measurement noise through harmonic analysis. However, for non-periodic signals, the distinction between nonlinear distortion and measurement noise cannot be made due to their stochastic nature, and they appear similar in the output spectrum. Zhang et al. (2010) proposed a solution to use a non-stationary excitation signal to detect nonlinear distortions. The idea is to assume that the measurement noise is stationary. Then, if nonlinear distortions exist, the output noise (the part of the output that a linear model cannot capture) will be non-stationary. A periodic excitation is still required to separate the nonlinear distortions from the measurement noise, because it allows averaging over periods to remove noise and identify harmonics to separate nonlinear distortions.

For non-periodic excitations, the measured input and output signals are split into M smaller blocks. Each block is treated as an independent realisation of the system response. For each block, the DFT is computed, and then the spectra are computed using Welch's method:

$$\begin{aligned} \hat{S}_{uu}(i) &= \frac{1}{M}\sum_{m=1}^{M} U_m(i)U_m^*(i), \\ \hat{S}_{yu}(i) &= \frac{1}{M}\sum_{m=1}^{M} Y_m(i)U_m^*(i), \end{aligned} \tag{2.26}$$

where $U_m(i)$ and $Y_m(i)$ are the DFT of the input and output signals in block m, and $*$ denotes the complex conjugate. The BLA is then given by:

$$\hat{G}_{\mathrm{BLA}}(i) = \frac{\hat{S}_{yu}(i)}{\hat{S}_{uu}(i)}. \tag{2.27}$$

This gives an estimate of the linear transfer function of the system at each frequency.

Another important issue when using non-periodic excitation is leakage. A common practice to reduce leakage error is applying a windowing function to the time-domain

signal before computing the DFT (Schoukens et al., 2006). A Hanning window or a diff window (Schoukens et al., 2006) is widely used. The windowing function affects the variance on $\hat{G}_{\mathrm{BLA}}(i)$. It is shown in Paduart (2008), that the variance on $\hat{G}_{\mathrm{BLA}}(i)$ using diff window is given by:

$$\hat{\sigma}^2_{\mathrm{BLA}}(i) = \frac{1}{M} \frac{\hat{\sigma}^2_{\mathrm{Y}}(i)}{\hat{S}_{uu}(i)}, \tag{2.28}$$

where $\hat{\sigma}^2_{\mathrm{Y}}(i)$ is the estimated variance of the output spectrum, defined as:

$$\hat{\sigma}^2_{\mathrm{Y}}(i) = \frac{M}{2(M-1)} \left(\hat{S}_{yy}(i) - \hat{G}_{\mathrm{BLA}}(i)\, \hat{S}_{uy}(i) \right). \tag{2.29}$$

This expression shows that increasing the number of blocks M in Welch's method reduces the variance of the BLA estimate, making it less sensitive to random fluctuations. However, this variance reduction comes at the cost of using shorter input-output data segments for each block. Shorter blocks increase the leakage effect and reduce frequency resolution. Therefore, the choice of M involves a trade-off between lowering the variance of $\hat{G}_{\mathrm{BLA}}(i)$ and maintaining good frequency resolution.

2.3.6 Non-parametric BLA to Parametric BLA

Traditionally, the BLA is used as a starting point for identifying nonlinear systems, serving both as a benchmark and a basis for initial parameter estimation in more advanced nonlinear modelling approaches. Thus, this section explains how the non-parametric estimate $\hat{G}_{\mathrm{BLA}}(i)$ obtained in the previous section is transformed into a parametric model of the LSS form (1.14). The $\hat{G}_{\mathrm{BLA}}(i)$ available in FRF form has been extracted from the input-output data. This step offers several benefits: enhanced SNR, enabling the focus on the frequency band of interest, and, for periodic data, the measurement noise and the effect of nonlinear behaviour can be separated. The procedure for the transformation of non-parametric to parametric BLA has been detailed in Paduart (2008).

Applying the DFT to the SISO LSS model (1.14) gives

$$\begin{aligned} z_i \boldsymbol{X}(i) &= \boldsymbol{A}\boldsymbol{X}(i) + \boldsymbol{b}U(i) + z_i \boldsymbol{x}_I, \\ Y(i) &= \boldsymbol{c}^\top \boldsymbol{X}(i) + dU(i), \end{aligned} \tag{2.30}$$

for $i = 1, \ldots, F$, where F is the number of excited frequency lines. The quantity $z_i = e^{j\omega_i T_{\mathrm{s}}}$ represents the complex exponential associated with the angular frequency ω_i (corresponding to the frequency response of the system) and T_{s} is the sampling time. The term $z_i \boldsymbol{x}_I$ represents the transient term, ensuring periodicity, where

$\boldsymbol{x_I}$ accounts for the initial and final states of the system:

$$\boldsymbol{x_I} = \frac{1}{\sqrt{N}}(\boldsymbol{x}(0) - \boldsymbol{x}(N)), \tag{2.31}$$

with N the period length of the DFT. Neglecting the transient term, and rewriting (2.30) as FRF for SISO system, by letting $U(i) = 1$:

$$\begin{aligned} z_i \boldsymbol{X}(i) &= \boldsymbol{A}\boldsymbol{X}(i) + \boldsymbol{b}, \\ G(i) &= \boldsymbol{c}^\top \boldsymbol{X}(i) + d, \end{aligned} \tag{2.32}$$

where the state vector $\boldsymbol{X}(i) \in \mathbb{C}^M$, $G(i) \in \mathbb{C}$ and M is the order of the model.

Multiplying and expanding the output equation of (2.32) by z_i^p and substituting the state equation $z_i X(i)$ intermittently:

$$\begin{aligned} z_i^p G(i) &= z_i^{p-1}(\boldsymbol{c}^\top z_i \boldsymbol{X}(i) + z_i d), \\ &= z_i^{p-1}(\boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{X}(i) + \boldsymbol{c}^\top \boldsymbol{b} + z_i d), \\ &= z_i^{p-2}(\boldsymbol{c}^\top \boldsymbol{A}^2 \boldsymbol{X}(i) + \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} + z_i \boldsymbol{c}^\top \boldsymbol{b} + z_i^2 d), \\ &\vdots \end{aligned} \tag{2.33}$$

Generalising after $p - 1$ substitutions, we get:

$$z_i^p G(i) = \boldsymbol{c}^\top \boldsymbol{A}^p \boldsymbol{X}(i) + \boldsymbol{c}^\top \boldsymbol{A}^{p-1}\boldsymbol{b} + z_i \boldsymbol{c}^\top \boldsymbol{A}^{p-2}\boldsymbol{b} + \cdots + z_i^{p-1}\boldsymbol{c}^\top \boldsymbol{b} + z_i^p d. \tag{2.34}$$

For $p = 0, \ldots, r-1$ and $r > M$:

$$\begin{aligned} G(i) &= \boldsymbol{c}^\top \boldsymbol{X}(i) + d, \\ z_i G(i) &= \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{X}(i) + \boldsymbol{c}^\top \boldsymbol{b} + z_i d, \\ &\vdots \\ z_i^{r-1} G(i) &= \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \boldsymbol{X}(i) + \boldsymbol{c}^\top \boldsymbol{A}^{r-2}\boldsymbol{b} + \cdots + z_i^{r-2}\boldsymbol{c}^\top \boldsymbol{b} + z_i^{r-1} d. \end{aligned} \tag{2.35}$$

From the above set of r equations, the extended observability matrix $\boldsymbol{\Gamma}_r$ and the matrix $\boldsymbol{S}_r$ containing Markov parameters in Toeplitz form can be defined as:

$$\boldsymbol{\Gamma}_r = \begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \end{bmatrix} \in \mathbb{R}^{r\times M}, \qquad \boldsymbol{S}_r = \begin{bmatrix} d & 0 & \dots & \dots & 0 \\ \boldsymbol{c}^\top \boldsymbol{b} & d & 0 & & 0 \\ \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{b} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-2}\boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}^{r-3}\boldsymbol{b} & \dots & \boldsymbol{c}^\top \boldsymbol{b} & d \end{bmatrix} \in \mathbb{R}^{r\times r}, \tag{2.36}$$

and for convenience, let $\boldsymbol{w}_r(i) \in \mathbb{C}^r$ be a vector containing the shift operators:

$$\boldsymbol{w}_r(i) = \begin{bmatrix} 1 & z_i & \dots & z_i^{r-1} \end{bmatrix}^\top. \tag{2.37}$$

Using definitions (2.36), (2.37) in (2.35), the following relation is obtained:

$$\boldsymbol{G} = \boldsymbol{\Gamma}_r \boldsymbol{X} + \boldsymbol{S}_r \boldsymbol{I}, \tag{2.38}$$

where the matrices $\boldsymbol{G} \in \mathbb{C}^{r\times F}, \boldsymbol{X} \in \mathbb{C}^{M\times F}, \boldsymbol{I} \in \mathbb{C}^{r\times F}$ are defined for SISO system as:

$$\begin{aligned} \boldsymbol{G} &= \begin{bmatrix} \boldsymbol{w}_r(1) \otimes G(1) & \dots & \boldsymbol{w}_r(F) \otimes G(F) \end{bmatrix}, \\ \boldsymbol{X} &= \begin{bmatrix} \boldsymbol{X}_1 & \dots & \boldsymbol{X}_F \end{bmatrix}, \\ \boldsymbol{I} &= \begin{bmatrix} \boldsymbol{w}_r(1) & \dots & \boldsymbol{w}_r(F) \end{bmatrix}. \end{aligned} \tag{2.39}$$

Rewriting the above complex equation into a real equation by introducing $\boldsymbol{G}^{\mathrm{re}} = \begin{bmatrix} \mathrm{Re}(\boldsymbol{G}) & \mathrm{Im}(\boldsymbol{G}) \end{bmatrix}$ and so for $\boldsymbol{X}^{\mathrm{re}}$ and $\boldsymbol{I}^{\mathrm{re}}$:

$$\boldsymbol{G}^{\mathrm{re}} = \boldsymbol{\Gamma}_r \boldsymbol{X}^{\mathrm{re}} + \boldsymbol{S}_r \boldsymbol{I}^{\mathrm{re}}. \tag{2.40}$$

Under the noise assumption,

$$G(i) = G_0(i) + N_G(i), \tag{2.41}$$

where $N_G(i)$ is a complex Gaussian noise term, that is independent across i, with zero mean

$$\mathbb{E}\{N_G(i)\} = 0, \tag{2.42}$$

and variance $C_G(i)$ (covariance in case of MIMO systems):

$$C_G(i) = \mathbb{E}\{N_G(i) N_G(i)^*\}, \tag{2.43}$$

(2.40) becomes:

$$\boldsymbol{G}^{\mathrm{re}} = \boldsymbol{\Gamma}_r \boldsymbol{X}^{\mathrm{re}} + \boldsymbol{S}_r \boldsymbol{I}^{\mathrm{re}} + \boldsymbol{N}_G^{\mathrm{re}}, \tag{2.44}$$

with $\boldsymbol{N}_G \in \mathbb{C}^{r \times F}$ defined as:

$$\boldsymbol{N}_G = \begin{bmatrix} \boldsymbol{w}_r(1) \otimes N_G(1) & \dots & \boldsymbol{w}_r(F) \otimes N_G(F) \end{bmatrix}. \tag{2.45}$$

Another assumption made to derive the parametric form of the BLA is that $(\boldsymbol{A}, \boldsymbol{c})$ is observable and $(\boldsymbol{A}, \boldsymbol{b})$ is controllable. These assumptions ensure that the system dynamics are both fully reconstructable from the measured outputs and completely influenced by the inputs. For completeness, the concepts of observability and controllability are briefly defined below.

Definition 2.3.5 (Observability)**.** A system is said to be *observable* if its internal state can be uniquely determined from knowledge of its outputs over a finite time interval. Mathematically, the pair $(\boldsymbol{A}, \boldsymbol{c})$ is observable if and only if the observability matrix $\boldsymbol{\Gamma}_M$ has full column rank equal to the model order M, where

$$\boldsymbol{\Gamma}_M = \begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{M-1} \end{bmatrix} \in \mathbb{R}^{M \times M}. \tag{2.46}$$

Definition 2.3.6 (Controllability)**.** A system is said to be *controllable* if it can be driven from any initial state to any desired final state within a finite time interval by means of an appropriate input. Mathematically, the pair $(\boldsymbol{A}, \boldsymbol{b})$ is controllable if and only if the controllability matrix,

$$\boldsymbol{\Delta}_M = \begin{bmatrix} \boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \cdots & \boldsymbol{A}^{M-1}\boldsymbol{b} \end{bmatrix} \in \mathbb{R}^{M \times M},$$

has a full row rank equal to the model order M.

The state-space form is derived from (2.44) using the frequency domain subspace identification (McKelvey et al., 1996) Algorithm 1 summarised for the SISO system.

Algorithm 1 Frequency-domain subspace identification

Input: G_i and $C_G(i)$
Output: Extended observability matrix $\boldsymbol{\Gamma}_r$

1: Set $r > M$ and form:

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{I}^{\mathrm{re}} \\ \boldsymbol{G}^{\mathrm{re}} \end{bmatrix} \in \mathbb{R}^{2r \times 2F}, \tag{2.47}$$

and

$$\boldsymbol{C}_N = \mathrm{Re}\left(\sum_{i=1}^{F} \boldsymbol{w}_r(i) \boldsymbol{w}_r^*(i) \otimes C_G(i) \right) \in \mathbb{R}^{r \times r}. \tag{2.48}$$

2: Eliminate the input $\boldsymbol{I}^{\mathrm{re}}$ from $\boldsymbol{Z}$ using the QR decomposition of $\boldsymbol{Z}^\top$:

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{I}^{\mathrm{re}} \\ \boldsymbol{G}^{\mathrm{re}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{11}^\top & \boldsymbol{0} \\ \boldsymbol{R}_{12}^\top & \boldsymbol{R}_{22}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_1^\top \\ \boldsymbol{Q}_2^\top \end{bmatrix}, \tag{2.49}$$

where the block matrices $\boldsymbol{R}_{11}^\top, \boldsymbol{R}_{12}^\top, \boldsymbol{R}_{22}^\top$ are each of size $r \times r$.

3: Remove the noise influence from (2.44) by using whitening and singular value decomposition (SVD) of

$$\boldsymbol{C}_N^{-1/2} \boldsymbol{R}_{22}^\top = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^\top, \tag{2.50}$$

and estimate $\boldsymbol{\Gamma}_r$ as (in MATLAB notation)

$$\hat{\boldsymbol{\Gamma}}_r = \boldsymbol{C}_N^{-1/2} \boldsymbol{U}(:, 1 : M). \tag{2.51}$$

4: Using shift property of $\boldsymbol{\Gamma}_r$, estimate $\boldsymbol{A}$ and $\boldsymbol{c}$ from $\hat{\boldsymbol{\Gamma}}_r$:

$$\hat{\boldsymbol{A}} = (\hat{\boldsymbol{\Gamma}}_r^{\downarrow})^{\dagger} \hat{\boldsymbol{\Gamma}}_r^{\uparrow}, \tag{2.52}$$

where $\hat{\boldsymbol{\Gamma}}_r^{\downarrow}$ and $\hat{\boldsymbol{\Gamma}}_r^{\uparrow}$ are matrices with last and first rows removed, and

$$\hat{\boldsymbol{c}}^\top = \hat{\boldsymbol{\Gamma}}_r(1, :). \tag{2.53}$$

5: Estimate $\boldsymbol{b}$ and d using $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{c}}$: minimize the cost V_{ss} with respect to $\boldsymbol{b}$ and d:

$$V_{ss} = \sum_{i=1}^{F} \epsilon^*(i) C_G^{-1}(i) \epsilon(i), \tag{2.54}$$

with

$$\epsilon(i) = G_{ss}(\hat{\boldsymbol{A}}, \boldsymbol{b}, \hat{\boldsymbol{c}}, d, i) - G(i), \tag{2.55}$$

$$G_{ss}(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, i) = \boldsymbol{c}^\top (z_i \boldsymbol{I}_M - \boldsymbol{A})^{-1} \boldsymbol{b} + d. \tag{2.56}$$

2.4 Overview of Subspace-based State-Space Identification

The LSS model for a system is identified directly from the input-output data using the approaches from Van Overschee and De Moor (1994). The subspace state-space form was used for identifying the combined deterministic-stochastic system, which in theory is similar to the alternate representation in Figure 2.5. The linear part is the deterministic component that includes the LTI system. The stochastic part of the system introduces some randomness or uncertainty, including noise and unmodelled dynamics. A common way to represent the output of such systems is:

$$y_k = H^d(u_k) + H^s(w_k), \tag{2.57}$$

where $H^d(u_k)$ is the deterministic response to the input u_k and $H^s(w_k)$ is the stochastic response due to noise w_k. Alternatively, in state-space form:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}u_k + \boldsymbol{w}_k, \\ y_k &= \boldsymbol{c}^\top \boldsymbol{x}_k + du_k + v_k, \end{aligned} \tag{2.58}$$

where $\boldsymbol{w}_k \in \mathbb{R}^M$ and $v_k \in \mathbb{R}$ represent process noise (stochastic part of the system) and measurement noise, respectively, both assumed to have WGN properties. The parameters $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d$ define the deterministic dynamics. The covariance structure of the noise terms $\boldsymbol{w}_k$ and v_k in the combined deterministic-stochastic system (2.58) is given by:

$$\mathbb{E}\left[\begin{pmatrix} \boldsymbol{w}_k \\ v_k \end{pmatrix} \begin{pmatrix} \boldsymbol{w}_\ell^\top & v_\ell \end{pmatrix}\right] = \begin{bmatrix} \boldsymbol{Q}^s & \boldsymbol{s}^s \\ (\boldsymbol{s}^s)^\top & r^s \end{bmatrix} \delta_{k\ell} \geq 0, \tag{2.59}$$

where $\boldsymbol{Q}^s \in \mathbb{R}^{M\times M}$ represents the process noise covariance matrix, $\boldsymbol{s}^s \in \mathbb{R}^M$ represents the cross-covariance between process noise and measurement noise and $r^s \in \mathbb{R}$ is the variance of the measurement noise v_k. The system is assumed to be observable in the Kalman sense, i.e., the pair $(\boldsymbol{A}, \boldsymbol{c})$ forms an observable system with an observability matrix of full rank M. Controllability is assumed concerning both the deterministic input and the stochastic excitation, modelled via the pair $(\boldsymbol{A}, [\boldsymbol{b}\ (\boldsymbol{Q}^s)^{1/2}])$, where $(\boldsymbol{Q}^s)^{1/2}$ denotes the square root (e.g., Cholesky factor) of the process noise covariance matrix. This augmented input matrix accounts for the fact that process noise can also excite the system's internal states. The resulting controllability matrix is assumed to be of full rank, ensuring that the system is controllable under both input and noise excitation.

The system in (2.58) is split into deterministic ($.^d$) and stochastic parts ($.^s$) as below:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}^d_{k+1} + \boldsymbol{x}^s_{k+1} \\ &= (\boldsymbol{A}\boldsymbol{x}^d_k + \boldsymbol{b}u_k) + (\boldsymbol{A}\boldsymbol{x}^s_k + \boldsymbol{w}_k), \\ y_k &= y^d_k + y^s_k \\ &= (\boldsymbol{c}^\top \boldsymbol{x}^d_k + du_k) + (\boldsymbol{c}^\top \boldsymbol{x}^s_k + v_k). \end{aligned} \tag{2.60}$$

The modes (i.e., eigenvalues of $\boldsymbol{A}$) associated with the pair $\{\boldsymbol{A}, \boldsymbol{b}\}$ may be either stable or unstable, depending on whether the input u_k excites unstable dynamics. In contrast, the pair $\{\boldsymbol{A}, (\boldsymbol{Q}^s)^{1/2}\}$ is assumed to be stable, meaning that the stochastic part of the system does not excite any unstable modes of $\boldsymbol{A}$. Although the matrix $\boldsymbol{A}$ is shared by both the deterministic and stochastic subsystems, stability of each depends on how its respective input, either u_k or the process noise, excites the modes. This assumption ensures that the stochastic state and output components, $\boldsymbol{x}^s_k$ and y^s_k, remain bounded in a mean-square sense, i.e., their second-order moments (such as variance and covariance) remain finite for all k. This condition guarantees that the noise-driven dynamics do not diverge over time, allowing for stable statistical behaviour and meaningful stochastic analysis. On the other hand, the deterministic part (i.e., $\boldsymbol{x}^d_k$ and y^d_k) is allowed to exhibit quasi-stationary behaviour, including the presence of unstable modes. It is assumed that the pair $\{\boldsymbol{A}, \boldsymbol{b}\}$ is controllable, so that such unstable modes can be influenced or stabilised through the input u_k.

To maintain meaningful system behaviour, particularly during identification or filtering, the process noise should not destabilise the system. Therefore, if $\boldsymbol{A}$ contains unstable modes, the noise covariance $\boldsymbol{Q}^s$ must be such that these modes are not excited. In contrast to the input-driven subsystem, the pair $\{\boldsymbol{A}, (\boldsymbol{Q}^s)^{1/2}\}$ is not necessarily assumed to be controllable. As a result, the deterministic and stochastic subsystems may share common modes (when control and noise affect the same dynamics) or may be partially or fully decoupled (when noise and control influence disjoint mode subspaces).

Thus, the system identification problem is to identify $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{Q}^s, \boldsymbol{s}^s, r^s$ from the given input-output measurements generated by an unknown combined deterministic-stochastic model. The MATLAB implementation of Numerical Algorithms for Subspace-based State-Space Identification (N4SID) and its variants, such as the Multivariable Output Error State-Space (MOESP) algorithm (Verhaegen and Dewilde, 1992a,b) and Canonical Variate Analysis (CVA) algorithm (Larimore, 1990), are much more straightforward and user-friendly. Thus, the following sections detail the MATLAB implementation of these subspace-based algorithms.

2.4.1 Preliminaries

The state-space form in (2.58) for the noise-free case can be rewritten as:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ y_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{c}^\top & d \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ u_k \end{bmatrix}, \qquad \boldsymbol{Y_k} = \boldsymbol{\Theta}\boldsymbol{\phi_k}, \tag{2.61}$$

where all the unknown parameters are grouped in the matrix $\boldsymbol{\Theta}$. In the noisy case,

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ y_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{c}^\top & d \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ u_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_k \\ v_k \end{bmatrix}, \tag{2.62}$$

the parameter matrix is thus estimated by least-squares approximation as follows:

$$\begin{aligned} \hat{\boldsymbol{\Theta}}_{\mathrm{LS}} &= \begin{bmatrix} \hat{\boldsymbol{A}} & \hat{\boldsymbol{b}} \\ \hat{\boldsymbol{c}}^\top & \hat{d} \end{bmatrix} \\ &= \left[\sum_{k=0}^{N-1} \begin{pmatrix} x_{k+1} \\ y_k \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_k \\ u_k \end{pmatrix}^\top \right] \left[\sum_{k=0}^{N-1} \begin{pmatrix} x_k \\ u_k \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_k \\ u_k \end{pmatrix}^\top \right]^{-1}. \end{aligned} \tag{2.63}$$

The solution to (2.63) is unique if the regressor covariance is of full rank, i.e.,

$$\operatorname{rank}\left\{ \begin{bmatrix} \boldsymbol{x}_0 & \dots & \boldsymbol{x}_{N-1} \\ u_0 & \dots & u_{N-1} \end{bmatrix} \right\} = M + 1, \tag{2.64}$$

where M is the order. The key question here is how to obtain the states from the input-output data. This is the typical system realisation problem. The output y in the deterministic case can be expanded using the state equation as follows:

$$\begin{aligned} y_k &= \boldsymbol{c}^\top \boldsymbol{x}_k + d u_k \\ y_{k+1} &= \boldsymbol{c}^\top \boldsymbol{x}_{k+1} + d u_{k+1} = \boldsymbol{c}^\top \boldsymbol{A} \boldsymbol{x}_k + \boldsymbol{c}^\top \boldsymbol{b} u_k + d u_{k+1} \\ &\vdots \\ y_{k+r-1} &= \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \boldsymbol{x}_k + \boldsymbol{c}^\top \boldsymbol{A}^{r-2} \boldsymbol{b} u_k + \cdots + \boldsymbol{c}^\top \boldsymbol{b} u_{k+r-2} + d u_{k+r-1}. \end{aligned} \tag{2.65}$$

In matrix form, the above equations are given as:

$$\begin{bmatrix} y_k \\ y_{k+1} \\ y_{k+2} \\ \vdots \\ y_{k+r-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \boldsymbol{c}^\top \boldsymbol{A}^2 \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} d & & & & \\ \boldsymbol{c}^\top \boldsymbol{b} & d & & & \\ \boldsymbol{c}^\top \boldsymbol{A} \boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{b} & d & & \\ \vdots & \ddots & \ddots & \ddots & \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-2} \boldsymbol{b} & \dots & \dots & \boldsymbol{c}^\top \boldsymbol{b} & d \end{bmatrix} \begin{bmatrix} u_k \\ u_{k+1} \\ u_{k+2} \\ \vdots \\ u_{k+r-1} \end{bmatrix}. \tag{2.66}$$

Succinctly,

$$\boldsymbol{y}_r[k] = \boldsymbol{\Gamma}_r \boldsymbol{x}[k] + \boldsymbol{\Psi}_r \boldsymbol{u}_r[k], \tag{2.67}$$

where $\boldsymbol{\Gamma}_r \in \mathbb{R}^{r\times M}$ is the extended observability matrix and $\boldsymbol{\Psi}_r \in \mathbb{R}^{r\times r}$ is a Toeplitz structured matrix containing the Markov parameters.

Let us introduce the following matrix definitions for some N:

$$\begin{aligned}
\boldsymbol{U}_{0|r-1} &= \begin{bmatrix} \boldsymbol{u}_r[0] & \boldsymbol{u}_r[1] & \dots & \boldsymbol{u}_r[N-1] \end{bmatrix} = \begin{bmatrix} u_0 & u_1 & \dots & u_{N-1} \\ u_1 & u_2 & \dots & u_N \\ \vdots & \vdots & \ddots & \vdots \\ u_{r-1} & u_r & \dots & u_{r+N-2} \end{bmatrix} \in \mathbb{R}^{r\times N}, \\
\boldsymbol{Y}_{0|r-1} &= \begin{bmatrix} \boldsymbol{y}_r[0] & \boldsymbol{y}_r[1] & \dots & \boldsymbol{y}_r[N-1] \end{bmatrix} = \begin{bmatrix} y_0 & y_1 & \dots & y_{N-1} \\ y_1 & y_2 & \dots & y_N \\ \vdots & \vdots & \ddots & \vdots \\ y_{r-1} & y_r & \dots & y_{r+N-2} \end{bmatrix} \in \mathbb{R}^{r\times N}, \\
\boldsymbol{X}_0 &= \begin{bmatrix} \boldsymbol{x}[0] & \boldsymbol{x}[1] & \dots & \boldsymbol{x}[N-1] \end{bmatrix} \in \mathbb{R}^{M\times N}.
\end{aligned} \tag{2.68}$$

Rewriting (2.67) using the matrix definitions:

$$\boldsymbol{Y}_{0|r-1} = \boldsymbol{\Gamma}_r \boldsymbol{X}_0 + \boldsymbol{\Psi}_r \boldsymbol{U}_{0|r-1}, \tag{2.69}$$

Certain assumptions about the data must hold to ensure the correct identification of the state-space matrices.

Assumption 1. Reachability Condition: $\operatorname{rank}(\boldsymbol{X}_0) = M$.
The state trajectory matrix $\boldsymbol{X}_0$ is assumed to be of full rank to ensure that the state vector is sufficiently excited and the system is reachable. If the matrix is rank-deficient, then the system states do not fully evolve, and essential system dynamics may be missed in the identification process.

Assumption 2. Persistent Excitation of Inputs: $\operatorname{rank}(\boldsymbol{U}_{0|r-1}) = r$.
The Hankel matrix of inputs $\boldsymbol{U}_{0|r-1}$ is assumed to be of full rank to ensure that the input sequence is persistently exciting of order r, i.e., the input contains sufficient frequency components to excite all system modes. A poor choice of excitation signal might cause incorrect model estimation.

Assumption 3. No collinearity between inputs and states: $\operatorname{rank}\left\{ \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{X}_0 \end{bmatrix} \right\} = r + M$.
The input and state matrices are assumed not to span the same subspace (i.e., they are

independent). In practical terms, it prevents linear dependence between the input and the state vector.

Under these assumptions, an arbitrary input-output data matrix,

$$\boldsymbol{W}_{0|r-1} = \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{Y}_{0|r-1} \end{bmatrix}, \tag{2.70}$$

can be transformed into a particular form, from which the state-space parameters are determined. To understand what this particular form of coordinate transformation looks like, let us consider the following example.

Example 5. Suppose an impulse input $\boldsymbol{u} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & \dots \end{bmatrix}^\top$ is applied at instance $k = 3$ to an LTI system, the output $\boldsymbol{y}$ is given as $\boldsymbol{y} = \begin{bmatrix} 0 & 0 & 0 & h_0 & h_1 & \dots \end{bmatrix}^\top$. The Figure 2.7 shows the input-output response.

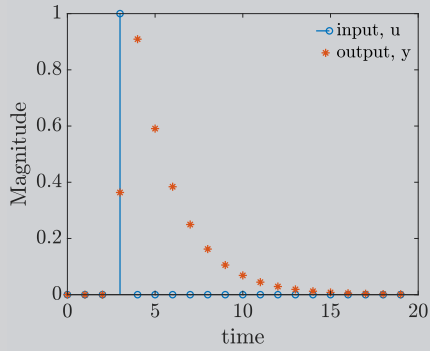


Figure 2.7: Zero-input response

Let $r = 4$ and $N = 8$, the data matrix $\boldsymbol{W}_{0|r-1}$ is given by:

$$\boldsymbol{W}_{0|3} = \begin{bmatrix} \boldsymbol{U}_{0|3} \\ \boldsymbol{Y}_{0|3} \end{bmatrix} = \left[\begin{array}{cccc:cccc} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 \\ 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & h_6 \\ h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 \end{array}\right], \tag{2.71}$$

where the top right block shows zero input to the system, and the bottom right block is a Hankel matrix showing the response to zero input.

From Ho-Kalman's method for system realisation, the impulse response coefficient matrix in Hankel form (say $\boldsymbol{H}$) can be written as:

$$\boldsymbol{H} = \begin{bmatrix} h_1 & h_2 & \dots & h_r \\ h_2 & h_3 & & h_{r+1} \\ \vdots & & \ddots & \vdots \\ h_r & h_{r+1} & \dots & h_{2r-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}^\top \boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} & \dots & \boldsymbol{c}^\top \boldsymbol{A}^{r-1}\boldsymbol{b} \\ \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}^2\boldsymbol{b} & & \boldsymbol{c}^\top \boldsymbol{A}^r\boldsymbol{b} \\ \vdots & & \ddots & \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1}\boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}^r\boldsymbol{b} & \dots & \boldsymbol{c}^\top \boldsymbol{A}^{2r-2}\boldsymbol{b} \end{bmatrix}. \tag{2.72}$$

Interestingly, the matrix $\boldsymbol{H}$ can be decomposed into the extended observability and the extended controllability matrices as below:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \dots & \boldsymbol{A}^{r-1}\boldsymbol{b} \end{bmatrix} = \boldsymbol{\Gamma}_r \boldsymbol{\Delta}_r. \tag{2.73}$$

Thus, SVD of the matrix $\boldsymbol{H}$ results in the factors $\boldsymbol{\Gamma}_r$ and $\boldsymbol{\Delta}_r$, as shown below:

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \begin{bmatrix} \boldsymbol{U}_1 & \boldsymbol{U}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1^\top \\ \boldsymbol{V}_2^\top \end{bmatrix} \approx \boldsymbol{U}_1 \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^\top. \tag{2.74}$$

Since the matrices $\boldsymbol{\Gamma}_r$ and $\boldsymbol{\Delta}_r$ are of full rank, a balanced realisation is possible. Hence,

$$\boldsymbol{\Gamma}_r = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1^{1/2}, \qquad \boldsymbol{\Delta}_r = \boldsymbol{\Sigma}_1^{1/2} \boldsymbol{V}_1^\top. \tag{2.75}$$

In general, this decomposition is feasible up to some similarity transformation with a non-singular matrix $\boldsymbol{T}$,

$$\boldsymbol{\Gamma}_r = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1^{1/2} \boldsymbol{T}, \qquad \boldsymbol{\Delta}_r = \boldsymbol{T}^{-1} \boldsymbol{\Sigma}_1^{1/2} \boldsymbol{V}_1^\top. \tag{2.76}$$

From the above generalised matrices, the minimal realisation system parameters can be estimated (in MATLAB notation) as below:

$$\boldsymbol{c} = \boldsymbol{\Gamma}_r(1,:)^\top \in \mathbb{R}^M, \qquad \boldsymbol{b} = \boldsymbol{\Delta}_r(:,1) \in \mathbb{R}^M. \tag{2.77}$$

The matrix $\boldsymbol{A}$ is obtained using the shift property of $\boldsymbol{\Gamma}_r$. Post-multiplying $\boldsymbol{A}$ to $\boldsymbol{\Gamma}_r$:

$$\begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \end{bmatrix} \boldsymbol{A} = \begin{bmatrix} \boldsymbol{c}^\top \boldsymbol{A} \\ \boldsymbol{c}^\top \boldsymbol{A}^2 \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^r \end{bmatrix} \tag{2.78}$$

$$\boldsymbol{\Gamma}_r \boldsymbol{A} = \boldsymbol{\Gamma}_{r+1}^{'}$$

$$\boldsymbol{A} = (\boldsymbol{\Gamma}_r^\top \boldsymbol{\Gamma}_r)^{-1} \boldsymbol{\Gamma}_r^\top \boldsymbol{\Gamma}_{r+1}^{'}.$$

Thus, the idea presented in Example 5 can be generalised to any input-output data. When the input signal contains a segment of zero input, the corresponding data matrix $\boldsymbol{W}_{0|r-1}$ can be structured to separate the input and zero-input response components. Specifically, it can be reorganised into the following block lower-triangular form:

$$\boldsymbol{W}_{0|r-1} = \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{Y}_{0|r-1} \end{bmatrix} \rightsquigarrow \begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22} \end{bmatrix}, \tag{2.79}$$

where the top-right block corresponds to the known zero-input segment, and $\boldsymbol{L}_{22}$ contains the system's zero-input (impulse) response. The reordering or structuring shown here follows from aligning the data matrix as in Example 5.

Lemma 1. Let the Assumptions 1, 2, 3 hold and $\mathrm{rank}(\boldsymbol{\Gamma}_r) = M$ for the data matrix $\boldsymbol{W}_{0|r-1}$, then $\mathrm{rank}(\boldsymbol{W}_{0|r-1}) = r + M$.

Proof. The Lemma is similar to the Assumption 3. Thus, the assumption can be used to prove. Rearranging (2.69), we get

$$\boldsymbol{W}_{0|r-1} = \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{Y}_{0|r-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_r & \boldsymbol{0} \\ \boldsymbol{\Psi}_r & \boldsymbol{\Gamma}_r \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{X}_0 \end{bmatrix}. \tag{2.80}$$

The identity matrix $\boldsymbol{I}_r$ is of $\mathrm{rank} - r$ and the extended observability matrix $\boldsymbol{\Gamma}_r$ is $\mathrm{rank} - M$. Thus, the matrices on the right side of the equation are $\mathrm{rank} - (r + M)$, which proves the rank of the matrix $\boldsymbol{W}_{0|r-1}$ is also $r + M$. □

This lemma implies that eliminating row vectors in $\boldsymbol{Y}_{0|r-1}$ that are dependent on row vectors in $\boldsymbol{U}_{0|r-1}$, there remain exactly M independent row vectors in $\boldsymbol{Y}_{0|r-1}$.

Lemma 2. Any input-output pair of length r that satisfies

$$\boldsymbol{y}_r(0) = \boldsymbol{\Gamma}_r \boldsymbol{x}(0) + \boldsymbol{\Psi}_r \boldsymbol{u}_r(0), \tag{2.81}$$

can be expressed as a linear combination of the column vectors of data matrix $\boldsymbol{W}_{0|r-1}$, which satisfies Assumptions 1, 2, 3. This implies that there exists a vector $\boldsymbol{\zeta} \in \mathbb{R}^N$, such that

$$\begin{bmatrix} \boldsymbol{u}_r(0) \\ \boldsymbol{y}_r(0) \end{bmatrix} = \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{Y}_{0|r-1} \end{bmatrix} \boldsymbol{\zeta}. \tag{2.82}$$

Proof. For an arbitrary input-output pair $\boldsymbol{u}_r(0)$ and $\boldsymbol{y}_r(0)$, there exists an initial state vector $\boldsymbol{x}(0)$ that satisfies the equation (2.81). Using the assumption 3,

$$\begin{bmatrix} \boldsymbol{u}_r(0) \\ \boldsymbol{x}(0) \end{bmatrix} = \begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{X}_0 \end{bmatrix} \boldsymbol{\zeta}. \tag{2.83}$$

Rearranging (2.69), we get

$$\begin{aligned}\begin{bmatrix}\boldsymbol{u}_r(0)\\ \boldsymbol{y}_r(0)\end{bmatrix} &= \begin{bmatrix}\boldsymbol{I}_r & \boldsymbol{0}\\ \boldsymbol{\Psi}_r & \boldsymbol{\Gamma}_r\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_r(0)\\ \boldsymbol{x}(0)\end{bmatrix}\\ &= \begin{bmatrix}\boldsymbol{I}_r & \boldsymbol{0}\\ \boldsymbol{\Psi}_r & \boldsymbol{\Gamma}_r\end{bmatrix}\begin{bmatrix}\boldsymbol{U}_{0|r-1}\\ \boldsymbol{X}_0\end{bmatrix}\boldsymbol{\zeta} \qquad (2.84)\\ &= \begin{bmatrix}\boldsymbol{U}_{0|r-1}\\ \boldsymbol{Y}_{0|r-1}\end{bmatrix}\boldsymbol{\zeta}.\end{aligned}$$

□

The Lemmas are used to transform the data matrix $\boldsymbol{W}_{0|r-1}$ to a zero-input response form:

$$\exists \boldsymbol{\zeta} \in \mathbb{R}^N, \quad \text{s.t.} \quad \boldsymbol{W}_{0|r-1}\boldsymbol{\zeta} = \begin{bmatrix}0\\ \vdots\\ 0\\ \tilde{y}(0)\\ \vdots\\ \tilde{y}(r-1)\end{bmatrix}. \qquad (2.85)$$

The vector $\boldsymbol{\zeta}$ is a linear combination of the columns of the data matrix. The right-hand side of (2.85) represents the zero-input response of the system. Repeating this process multiple times, a Hankel matrix that contains sufficient information about the system dynamics can be constructed. From this Hankel matrix, the state-space parameters $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ can be extracted in a structured way, as shown in Example 5. LQ decomposition can be applied to transform the data matrix into a lower triangular matrix, which is the desired zero-input response form as shown below:

$$\begin{bmatrix}\boldsymbol{U}_{0|r-1}\\ \boldsymbol{Y}_{0|r-1}\end{bmatrix}\underbrace{\begin{bmatrix}\boldsymbol{\zeta}^1 & \dots & \boldsymbol{\zeta}^{2r}\end{bmatrix}}_{\boldsymbol{Q}} = \begin{bmatrix}\boldsymbol{L}_{11} & \boldsymbol{0}\\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22}\end{bmatrix}. \qquad (2.86)$$

By choosing multiple different vectors $\boldsymbol{\zeta}^1, \boldsymbol{\zeta}^2, \dots, \boldsymbol{\zeta}^{2r}$, multiple zero-input responses can be constructed. $\boldsymbol{\zeta}$ vectors can be made orthogonal using Householder

transformations, forming an orthonormal matrix $\boldsymbol{Q}$

$$\begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{Y}_{0|r-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_1 & \boldsymbol{Q}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22} \end{bmatrix}$$
$$\begin{bmatrix} \boldsymbol{U}_{0|r-1} \\ \boldsymbol{Y}_{0|r-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_1^\top \\ \boldsymbol{Q}_2^\top \end{bmatrix}, \tag{2.87}$$

where $\boldsymbol{L}_{11}, \boldsymbol{L}_{22} \in \mathbb{R}^{r \times r}$ are lower triangular matrices and $\boldsymbol{Q}_1, \boldsymbol{Q}_2 \in \mathbb{R}^{N \times r}$ are orthogonal matrices. Hence, $\boldsymbol{Q}\boldsymbol{Q}^\top = \boldsymbol{I}$.

2.4.2 Multivariable Output Error State-Space Method

The simple Multivariable Output Error State-Space (MOESP) algorithm (Verhaegen and Dewilde, 1992a,b; Verhaegen and Verdult, 2007) is a linear subspace identification method that identifies system parameters, expanding the idea presented in Section 2.4.1.

From (2.87), we get

$$\begin{aligned} \boldsymbol{U}_{0|r-1} &= \boldsymbol{L}_{11}\boldsymbol{Q}_1^\top, \\ \boldsymbol{Y}_{0|r-1} &= \boldsymbol{L}_{21}\boldsymbol{Q}_1^\top + \boldsymbol{L}_{22}\boldsymbol{Q}_2^\top. \end{aligned} \tag{2.88}$$

Rewriting (2.69) using the above definitions, we get

$$\boldsymbol{L}_{21}\boldsymbol{Q}_1^\top + \boldsymbol{L}_{22}\boldsymbol{Q}_2^\top = \boldsymbol{\Gamma}_r \boldsymbol{X}_0 + \boldsymbol{\Psi}_r \boldsymbol{L}_{11}\boldsymbol{Q}_1^\top. \tag{2.89}$$

As seen from Example 5, the terms containing the extended observability matrix $\boldsymbol{\Gamma}_r$ and the zero-input response form $\boldsymbol{L}_{22}$ are of interest to us. Thus, eliminating the other terms by post-multiplying $\boldsymbol{Q}_2$, we get

$$\boldsymbol{L}_{22} = \boldsymbol{\Gamma}_r \boldsymbol{X}_0 \boldsymbol{Q}_2. \tag{2.90}$$

Taking SVD of $\boldsymbol{L}_{22}$, we get

$$\boldsymbol{\Gamma}_r \boldsymbol{X}_0 \boldsymbol{Q}_2 = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^\top, \tag{2.91}$$

and the observability matrix is given by:

$$\boldsymbol{\Gamma}_r = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1^{1/2}. \tag{2.92}$$

From $\boldsymbol{\Gamma}_r$, parameters $\boldsymbol{A}$ and $\boldsymbol{c}$ are obtained as explained in (2.78) and (2.77) respectively. Next, the parameters $\boldsymbol{b}$ and d are determined from $\boldsymbol{\Psi}_r$, using the estimates $\boldsymbol{A}$ and $\boldsymbol{c}$.

The parameter d is directly available from $\boldsymbol{\Psi}_r$ as first element and the parameter $\boldsymbol{b}$ is obtained by solving the following least squares problem:

$$\boldsymbol{\Psi}_r(2:\text{end},1) = \begin{bmatrix} \boldsymbol{c}^\top \boldsymbol{b} \\ \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-2}\boldsymbol{b} \end{bmatrix} = \boldsymbol{\Gamma}_r \boldsymbol{b}. \tag{2.93}$$

In the presence of noise, the modified versions of MOESP called Past Input - MOESP (PI-MOESP) and Past Output - MOESP (PO-MOESP) from Verhaegen and Verdult (2007) are used for the system parameter estimation.

2.4.3 Numerical Algorithms for State-Space Subspace Identification

The Numerical Algorithms for State-Space Subspace Identification (N4SID) method also uses LQ decomposition and SVD. However, it differs from the MOESP method. In this method, the data is split into past (subscript p) and future (subscript f) sets as shown in Figure 2.8

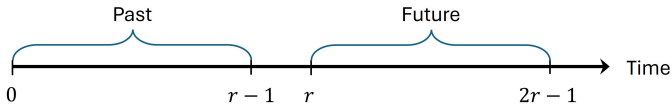


Figure 2.8

and the following matrix definitions hold:

$$\boldsymbol{U}_p = \boldsymbol{U}_{0|r-1} = \begin{bmatrix} u_0 & \dots & u_{N-1} \\ \vdots & \ddots & \vdots \\ u_{r-1} & \dots & u_{r+N-2} \end{bmatrix}, \quad \boldsymbol{U}_f = \boldsymbol{U}_{r|2r-1} = \begin{bmatrix} u_r & \dots & u_{r+N-1} \\ \vdots & \ddots & \vdots \\ u_{2r-1} & \dots & u_{2r+N-2} \end{bmatrix},$$

$$\boldsymbol{Y}_p = \boldsymbol{Y}_{0|r-1} = \begin{bmatrix} y_0 & \dots & y_{N-1} \\ \vdots & \ddots & \vdots \\ y_{r-1} & \dots & y_{r+N-2} \end{bmatrix}, \quad \boldsymbol{Y}_f = \boldsymbol{Y}_{r|2r-1} = \begin{bmatrix} y_r & \dots & y_{r+N-1} \\ \vdots & \ddots & \vdots \\ y_{2r-1} & \dots & y_{2r+N-2} \end{bmatrix},$$

$$\boldsymbol{X}_p = \boldsymbol{X}_0 = \begin{bmatrix} \boldsymbol{x}[0] & \dots & \boldsymbol{x}[N-1] \end{bmatrix}, \quad \boldsymbol{X}_f = \boldsymbol{X}_r = \begin{bmatrix} \boldsymbol{x}[r] & \dots & \boldsymbol{x}[r+N-1] \end{bmatrix}.$$

For the above definition of output matrices, the relation (2.69) holds. Hence,

$$\boldsymbol{Y}_p = \boldsymbol{\Gamma}_r \boldsymbol{X}_p + \boldsymbol{\Psi}_r \boldsymbol{U}_p, \qquad (2.94) \qquad\qquad \boldsymbol{Y}_f = \boldsymbol{\Gamma}_r \boldsymbol{X}_f + \boldsymbol{\Psi}_r \boldsymbol{U}_f. \qquad (2.95)$$

The Assumptions 1, 2 and 3, made earlier in Section 2.4.1, are extended for past and future dataset splits as below:

Assumption 4. Reachability Condition:

$$\operatorname{rank}(\boldsymbol{X}_p) = \operatorname{rank}(\boldsymbol{X}_f) = M.$$

Assumption 5. Persistent Excitation of Inputs:

$$\operatorname{rank}(\boldsymbol{U}_{0|2r-1}) = \operatorname{rank}\left\{ \begin{bmatrix} \boldsymbol{U}_p \\ \boldsymbol{U}_f \end{bmatrix} \right\} = 2r.$$

Assumption 6. No collinearity between inputs and states:

$$\operatorname{rank}\left\{ \begin{bmatrix} \boldsymbol{U}_{0|2r-1} \\ \boldsymbol{X}_p \end{bmatrix} \right\} = \operatorname{rank}\left\{ \begin{bmatrix} \boldsymbol{U}_{0|2r-1} \\ \boldsymbol{X}_f \end{bmatrix} \right\} = 2r + M.$$

Similar to Lemma 2, for a data matrix satisfying Assumptions 4, 5, and 6, any input-output response can be expressed as a linear combination of the column vectors of the data matrix:

$$\exists \boldsymbol{\zeta} \in \mathbb{R}^N, \quad \text{s.t.} \quad \begin{bmatrix} \boldsymbol{u}_f \\ \boldsymbol{u}_p \\ \boldsymbol{y}_p \\ \boldsymbol{y}_f \end{bmatrix} = \begin{bmatrix} \boldsymbol{U}_f \\ \boldsymbol{U}_p \\ \boldsymbol{Y}_p \\ \boldsymbol{Y}_f \end{bmatrix} \boldsymbol{\zeta}, \qquad (2.96)$$

where the arrangement of past and future data sets is recommended for notational simplicity in later stages. To attain the zero-input response form, we use the same method as seen in (2.86), and extend the Lemma 2 for the new data matrix:

$$\begin{bmatrix} \boldsymbol{U}_f \\ \boldsymbol{U}_p \\ \boldsymbol{Y}_p \\ \boldsymbol{Y}_f \end{bmatrix} \underbrace{\begin{bmatrix} \boldsymbol{\zeta}^1 & \dots & \boldsymbol{\zeta}^{4r} \end{bmatrix}}_{\boldsymbol{Q}} = \begin{bmatrix} \boldsymbol{L}_{11} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{L}_{21} & \boldsymbol{L}_{22} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{L}_{31} & \boldsymbol{L}_{32} & \boldsymbol{L}_{33} & \boldsymbol{0} \\ \boldsymbol{L}_{41} & \boldsymbol{L}_{42} & \boldsymbol{L}_{43} & \boldsymbol{L}_{44} \end{bmatrix}, \qquad (2.97)$$

where each block matrices are of dimension $r \times r$ and the block matrices on the diagonal $\boldsymbol{L}_{11}, \dots, \boldsymbol{L}_{44}$ are lower triangular. Interestingly, the block matrix $\boldsymbol{L}_{44}$ is a zero matrix, since all the inputs of both past and future times, and the past outputs are zero. Replacing

$\begin{bmatrix} \boldsymbol{U}_p \\ \boldsymbol{Y}_p \end{bmatrix}$ with $\boldsymbol{W}_p$ and rewriting the above equation:

$$\begin{bmatrix} \boldsymbol{U}_f \\ \boldsymbol{W}_p \\ \boldsymbol{Y}_f \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} & \boldsymbol{0} \\ \boldsymbol{R}_{31} & \boldsymbol{R}_{32} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_1^\top \\ \boldsymbol{Q}_2^\top \\ \boldsymbol{Q}_2^\top \end{bmatrix}, \tag{2.98}$$

where $\boldsymbol{R}_{11} = \boldsymbol{L}_{11}, \boldsymbol{R}_{31} = \boldsymbol{L}_{41}$,
$\boldsymbol{R}_{21} = \begin{bmatrix} \boldsymbol{L}_{21} \\ \boldsymbol{L}_{31} \end{bmatrix}, \boldsymbol{R}_{22} = \begin{bmatrix} \boldsymbol{L}_{22} & \boldsymbol{0} \\ \boldsymbol{L}_{32} & \boldsymbol{L}_{33} \end{bmatrix}$ and $\boldsymbol{R}_{32} = \begin{bmatrix} \boldsymbol{L}_{42} & \boldsymbol{L}_{43} \end{bmatrix}$. The matrix $\boldsymbol{Q} \in \mathbb{R}^{N \times 4r}$ is orthogonal and therefore $\boldsymbol{Q}_i^\top \boldsymbol{Q}_j = 0, i \neq j$.

With the decomposition of the data matrix, the following system of equations can be obtained:

$$\begin{aligned} \boldsymbol{U}_f &= \boldsymbol{R}_{11} \boldsymbol{Q}_1^\top, \\ \boldsymbol{W}_p &= \boldsymbol{R}_{21} \boldsymbol{Q}_1^\top + \boldsymbol{R}_{22} \boldsymbol{Q}_2^\top, \\ \boldsymbol{Y}_f &= \boldsymbol{R}_{31} \boldsymbol{Q}_1^\top + \boldsymbol{R}_{32} \boldsymbol{Q}_2^\top. \end{aligned} \tag{2.99}$$

Since $\boldsymbol{U}_f$ is full rank, $\boldsymbol{R}_{11}$ is also full rank and non-singular, therefore

$$\boldsymbol{Q}_1^\top = \boldsymbol{R}_{11}^{-1} \boldsymbol{U}_f. \tag{2.100}$$

From Lemma 1, the $\mathrm{rank}(\boldsymbol{W}_p) = r + M$. Therefore, by definition, the $\mathrm{rank}(\boldsymbol{R}_{22}) = \mathrm{rank}\left\{ \begin{bmatrix} \boldsymbol{L}_{22} & \boldsymbol{0} \\ \boldsymbol{L}_{32} & \boldsymbol{L}_{33} \end{bmatrix} \right\} = r + M$, indicating that $\boldsymbol{R}_{22}$ is rank deficit. Hence, $\boldsymbol{Q}_2^\top$ is given by:

$$\boldsymbol{Q}_2^\top = \boldsymbol{R}_{22}^\dagger (\boldsymbol{W}_p - \boldsymbol{R}_{21} \boldsymbol{Q}_1^\top). \tag{2.101}$$

By substituting the definitions of $\boldsymbol{Q}_1^\top, \boldsymbol{Q}_2^\top$ in $\boldsymbol{Y}_f$ of (2.99), the following equation is obtained:

$$\begin{aligned} \boldsymbol{Y}_f &= \boldsymbol{R}_{31} \boldsymbol{R}_{11}^{-1} \boldsymbol{U}_f + \boldsymbol{R}_{32} \boldsymbol{R}_{22}^\dagger (\boldsymbol{W}_p - \boldsymbol{R}_{21} \boldsymbol{R}_{11}^{-1} \boldsymbol{U}_f) \\ &= (\boldsymbol{R}_{31} - \boldsymbol{R}_{32} \boldsymbol{R}_{22}^\dagger \boldsymbol{R}_{21}) \boldsymbol{R}_{11}^{-1} \boldsymbol{U}_f + \boldsymbol{R}_{32} \boldsymbol{R}_{22}^\dagger \boldsymbol{W}_p. \end{aligned} \tag{2.102}$$

Equation (2.102) is of the form (2.95) and the following observations can be derived from (2.102): $\boldsymbol{Y}_p$ is spanned by $\boldsymbol{X}_p$ and $\boldsymbol{U}_p$. This means that the row vectors in $\boldsymbol{Y}_p$ are linear combinations of the row vectors in $\boldsymbol{X}_p$ and $\boldsymbol{U}_p$, leading to:

$$\mathrm{span}(\boldsymbol{W}_p) = \mathrm{span}\left(\begin{bmatrix} \boldsymbol{U}_p \\ \boldsymbol{Y}_p \end{bmatrix} \right) = \mathrm{span}\left(\begin{bmatrix} \boldsymbol{U}_p \\ \boldsymbol{X}_p \end{bmatrix} \right). \tag{2.103}$$

Assumption 5 implies no overlap between the bases of $\boldsymbol{U}_p$ and $\boldsymbol{U}_f$. Therefore,

$$\operatorname{span}(\boldsymbol{U}_p) \cap \operatorname{span}(\boldsymbol{U}_f) = \varnothing. \tag{2.104}$$

Similarly, from Assumption 6, we have:

$$\operatorname{span}(\boldsymbol{X}_p) \cap \operatorname{span}(\boldsymbol{U}_f) = \varnothing. \tag{2.105}$$

Thus, from the above two relations, it can be concluded that:

$$\operatorname{span}(\boldsymbol{W}_p) \cap \operatorname{span}(\boldsymbol{U}_f) = \varnothing. \tag{2.106}$$

Therefore, Equation (2.102) represents $\boldsymbol{Y}_f$ as the sum of two terms that exist in two subspaces having no overlap in their bases (direct sum property). The same direct sum property can also be confirmed between $\boldsymbol{X}_f$ and $\boldsymbol{U}_f$ in (2.95).

With the relationships established between $\boldsymbol{W}_p$ and $\boldsymbol{U}_f$, and $\boldsymbol{X}_f$ and $\boldsymbol{U}_f$, we have to check how $\boldsymbol{W}_p$ and $\boldsymbol{X}_f$ are related. The state evolution equation can be written as:

$$\begin{aligned} \boldsymbol{x}_{r+i} &= \boldsymbol{A}^r \boldsymbol{x}_i + \boldsymbol{A}^{r-1}\boldsymbol{b}u_i + \boldsymbol{A}^{r-2}\boldsymbol{b}u_{i+1} + \cdots + \boldsymbol{b}u_{i+r-1} \\ &= \boldsymbol{A}^r \boldsymbol{x}_i + \underbrace{\begin{bmatrix} \boldsymbol{A}^{r-1}\boldsymbol{b} & \boldsymbol{A}^{r-2}\boldsymbol{b} & \dots & \boldsymbol{b} \end{bmatrix}}_{\bar{\boldsymbol{\Delta}}_r} \begin{bmatrix} u_i \\ u_{i+1} \\ \vdots \\ u_{i+r-1} \end{bmatrix}, \end{aligned} \tag{2.107}$$

where $\bar{\boldsymbol{\Delta}}_r$ is column-reversed Controllability matrix $\boldsymbol{\Delta}_r$. By repeating the above relationship for $i = 0, \dots, N-1$ and rewriting in matrix form, we get:

$$\boldsymbol{X}_f = \boldsymbol{A}^r \boldsymbol{X}_p + \bar{\boldsymbol{\Delta}}_r \boldsymbol{U}_p. \tag{2.108}$$

Using the relation (2.94), the matrix $\boldsymbol{X}_p$ can be written as:

$$\boldsymbol{X}_p = \boldsymbol{\Gamma}_r^{\dagger}(\boldsymbol{Y}_p - \boldsymbol{\Psi}_r \boldsymbol{U}_p). \tag{2.109}$$

Substituting this into (2.108), we get:

$$\begin{aligned} \boldsymbol{X}_f &= \boldsymbol{A}^r \boldsymbol{\Gamma}_r^{\dagger}(\boldsymbol{Y}_p - \boldsymbol{\Psi}_r \boldsymbol{U}_p) + \bar{\boldsymbol{\Delta}}_r \boldsymbol{U}_p \\ &= \boldsymbol{A}^r \boldsymbol{\Gamma}_r^{\dagger} \boldsymbol{Y}_p + (\bar{\boldsymbol{\Delta}}_r - \boldsymbol{A}^r \boldsymbol{\Gamma}_r^{\dagger} \boldsymbol{\Psi}_r) \boldsymbol{U}_p. \end{aligned} \tag{2.110}$$

This implies that the above equation is a direct sum and $\boldsymbol{X}_f$ is spanned by $\boldsymbol{U}_p$ and $\boldsymbol{Y}_p$, i.e.,

$$\boldsymbol{X}_f \in \operatorname{span}(\boldsymbol{W}_p). \tag{2.111}$$

Therefore, from equations (2.95) and (2.102), it can be observed that they represent the same quantity ($\boldsymbol{Y}_f$), their corresponding terms must be equal in subspaces. Hence equating the terms of interest, we get:

$$\boldsymbol{\Gamma}_r \boldsymbol{X}_f = \boldsymbol{R}_{32} \boldsymbol{R}_{22}^{\dagger} \boldsymbol{W}_p. \tag{2.112}$$

Applying SVD to the relationship:

$$\boldsymbol{R}_{32} \boldsymbol{R}_{22}^{\dagger} \boldsymbol{W}_p = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^{\top} = \begin{bmatrix} \boldsymbol{U}_1 & \boldsymbol{U}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1^{\top} \\ \boldsymbol{V}_2^{\top} \end{bmatrix} = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^{\top}. \tag{2.113}$$

Splitting Between $\boldsymbol{\Gamma}_r$ and $\boldsymbol{X}_f$:

$$\boldsymbol{\Gamma}_r = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1^{1/2} \boldsymbol{T}, \qquad \boldsymbol{X}_f = \boldsymbol{T}^{-1} \boldsymbol{\Sigma}_1^{1/2} \boldsymbol{V}_1^{\top}. \tag{2.114}$$

From the extended observability matrix $\boldsymbol{\Gamma}_r$, the system parameters $\boldsymbol{A}$ and $\boldsymbol{c}$ can be obtained, and from $\boldsymbol{\Psi}_r$, the parameters d and $\boldsymbol{b}$ can be obtained, as explained in Section 2.4.2. However, in the presence of noise, we will rely on the $\boldsymbol{X}_f$ relation. Consider the following definitions:

$$\begin{aligned} \bar{\boldsymbol{X}}_r &= \begin{bmatrix} \boldsymbol{x}[r] & \dots & \boldsymbol{x}[r+N-2] \end{bmatrix} \in \mathbb{R}^{M \times N-1}, \\ \bar{\boldsymbol{X}}_{r+1} &= \begin{bmatrix} \boldsymbol{x}[r+1] & \dots & \boldsymbol{x}[r+N-1] \end{bmatrix} \in \mathbb{R}^{M \times N-1}, \\ \bar{\boldsymbol{u}}_{r|r} &= \begin{bmatrix} u_r & \dots & u_{r+N-2} \end{bmatrix} \in \mathbb{R}^{1 \times N-1}, \\ \bar{\boldsymbol{y}}_{r|r} &= \begin{bmatrix} y_r & \dots & y_{r+N-2} \end{bmatrix} \in \mathbb{R}^{1 \times N-1}. \end{aligned} \tag{2.115}$$

These quantities define the state and measurement equations:

$$\begin{bmatrix} \bar{\boldsymbol{X}}_{r+1} \\ \bar{\boldsymbol{y}}_{r|r} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{c}^{\top} & d \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{X}}_r \\ \bar{\boldsymbol{u}}_{r|r} \end{bmatrix}. \tag{2.116}$$

The matrices $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d$ can now be extracted from this equation as below:

$$\begin{bmatrix} \hat{\boldsymbol{A}} & \hat{\boldsymbol{b}} \\ \hat{\boldsymbol{c}}^{\top} & \hat{d} \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{X}}_{r+1} \\ \bar{\boldsymbol{y}}_{r|r} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{X}}_r \\ \bar{\boldsymbol{u}}_{r|r} \end{bmatrix}^{\top} \left\{ \begin{bmatrix} \bar{\boldsymbol{X}}_r \\ \bar{\boldsymbol{u}}_{r|r} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{X}}_r \\ \bar{\boldsymbol{u}}_{r|r} \end{bmatrix}^{\top} \right\}^{-1}. \tag{2.117}$$

2.4.4 MATLAB Commands

The system identification methods discussed in this chapter, such as subspace identification techniques including N4SID and MOESP, are well-established in the

literature and can be implemented from first principles. However, to generate baseline models used for performance comparison in this work, we used the implementations available in MATLAB's System Identification Toolbox. This choice allows for consistency with standard practices and facilitates reproducibility. To enable others to replicate the baseline results, the corresponding MATLAB commands are provided below.

Create an `iddata` object containing a time-domain output signal `y` and input signal `u`. `Ts` specifies the sample time of the experimental data:

```
data = iddata(y,u,Ts);
```

Set the initial state during estimation to zero:

```
opt = n4sidOptions('InitialState','zero');
```

Set the required Subspace algorithm:

```
opt = n4sidOptions('N4weight','MOESP');
```

Enforce stability of the estimated model:

```
opt = n4sidOptions('EnforceStability','true');
```

Estimate a discrete-time state-space model `sys` of order `M` using all the input and output signals in `data`. Additionally, supply some name-value arguments:

```
sys = n4sid(data,M,opt,'Feedthrough',1,...
'DisturbanceModel','none','Form','canonical');
A = sys.A; b = sys.B; c = sys.C; d = sys.D;
```

These implementations serve as reference models throughout the thesis to evaluate the effectiveness of the proposed identification methods introduced in later chapters. The estimated linear model of a nonlinear system is optimised for complete identification, using the initial parameter values of the estimated linear model. The optimisation procedure is discussed in Section 4.3.5.

2.5 Conclusion

This chapter provided a foundational overview of linear system identification techniques, including the BLA for obtaining a linear representation of nonlinear systems, followed by frequency-domain subspace identification methods applied to the estimated BLA,

as well as time-domain approaches such as N4SID and MOESP. These methods are used to generate baseline models that will later serve as benchmarks for evaluating the nonlinear identification strategies proposed in this thesis. Starting from classical linear modelling principles, the chapter introduced how linear models can approximate system behaviour under specific excitation conditions. The subspace-based methods discussed here form a solid reference for assessing the improvements achieved by the kernel-based nonlinear frameworks in the subsequent chapters.

Chapter 3

Case Study: Subspace-based Estimation of Multiple Poles

Abstract: This chapter presents a case study on identifying autonomous linear systems containing poles with multiplicity greater than one, using a subspace-based framework. It introduces a generalised signal model that can represent repeated poles and outlines techniques to estimate parameters of exponential polynomials, including pole locations, multiplicity structure, and complex amplitudes. A systematic search-based algorithm is proposed for determining the pole multiplicity from a fixed model order. The chapter emphasises both theoretical insights and practical performance, validating the proposed method through numerical simulations, including scenarios with noise and real-valued signals. The proposed strategy is particularly effective for low-order signals, where classical methods often fail to resolve repeated poles, but it also accommodates the general case of signals with both repeated and simple poles. Comparative analysis with existing techniques highlights the strengths and limitations of the approach, notably its robustness in signal reconstruction and sensitivity to noise and high-multiplicity poles.

This chapter is based on the following publication:

• P. Vijayan, P. Dreesen, I. Markovsky and M. Ishteva, "Parameter Estimation of Multiple Poles by Subspace-Based Method," in *Proceedings of the 2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2023, pp. 1948–1953. DOI: 10.1109/CoDIT58514.2023.10284047

The research was carried out by the PhD candidate under the guidance of the co-authors, who acted as promoters and co-promoters. I. Markovsky was the former promoter at the time of publication.

3.1 Background and Problem Statement

In fields such as communications, biomedical engineering and signal processing, the retrieval of information from measured data is vital. Noise in the data prevents the deterministic extraction of signal parameters. A widely used approach is to model signals as sums of damped complex exponentials, which correspond to systems with distinct (simple) poles. However, in many practical scenarios, closely spaced or repeated dynamics lead to repeated poles, requiring a generalisation of the model to exponential polynomials. This case study builds on that generalisation, which applies to both simple and repeated poles and focuses on estimating key signal characteristics: pole locations, multiplicities, and complex amplitudes.

Repeated poles arise in several application domains, where system responses exhibit closely spaced or repeated dynamics. In electromagnetic pulse analysis, especially in lossy or dispersive media, repeated poles may appear due to degenerate propagation paths or near-identical reflection delays (Sircar, 2016). In nuclear magnetic resonance spectroscopy, while the standard model assumes simple exponential decay, repeated poles can emerge in cases of overlapping resonances, or non-ideal relaxation effects (Barkhuijsen et al., 1987; Papy et al., 2009). In mechanical systems, repeated natural frequencies due to structural symmetries or boundary conditions can lead to multiple poles in the vibrational response (Meirovitch, 1967; Seyranian et al., 1994). Electrical circuits, such as RLC networks or filters, may exhibit pole multiplicities when component values induce degenerate resonance frequencies (Antoniou, 2006). In acoustics, repeated modes can result from architectural symmetries in reverberant spaces (Kuttruff, 2009), and in biomedical signals like ECG or EEG, repeated poles can be used to model sustained oscillatory phenomena or to capture dynamics where simple exponential models fail (Van Huffel and Vandewalle, 2004; Fira et al., 2021). In all these cases, exponential polynomial models allow for a more accurate representation of signals where standard sum of exponentials signals (with only simple poles) are insufficient.

Various statistical approaches have historically been used for estimating the parameters of signals. Linear prediction methods, such as Prony (1795); Pisarenko (1973), have long been used to determine poles and signal amplitudes. More recent estimation techniques include maximum likelihood estimation (Stoica and Sharmen, 1990), matrix pencil techniques (Hua and Sarkar, 1990), and subspace-based approaches such as Kung's realisation (Kung et al., 1983), the Rotational Invariance Technique (Roland

et al., 2006), and Hankel Total Least Squares (Morren et al., 2001; Papy et al., 2009), which are especially popular for their accuracy in pole estimation.

The aforementioned estimation methods are effective in accurately estimating simple poles. Estimating poles with multiplicity is known to result in a loss of accuracy (Zeng, 2005). To address this challenge, the sum-of-exponentials model must be extended (generalised). The generalised formulation considered here accommodates both simple and repeated poles. The ideas proposed in this chapter, by improving the identification of repeated poles and associated system dynamics, can support data-driven control design procedures of linear systems, such as those proposed by De Persis and Tesi (2020), where accurate identification of system dynamics, including repeated poles, is critical for constructing reliable models used in controller synthesis.

This study examines the classical problem of estimating signal parameters in an exponential polynomial model. This chapter introduces a generalised signal model that incorporates both simple and repeated poles, and proposes a systematic search-based algorithm designed to determine the pole multiplicity structure (i.e., number of times each pole is repeated) for a given model order. This algorithm utilises pole and complex amplitude estimates obtained from existing subspace-based methods to identify the pole multiplicity structure. The schematic in Fig. 3.1 summarises the full procedure. The approach constructs signal data into a Hankel matrix form for parameter estimation. The Hankel data matrix is decomposed using SVD, and the shift-invariance property is then applied to the unitary matrix of SVD to extract the poles. Once the poles are identified, the complex amplitudes are estimated. These estimated parameters are input into a systematic search-based algorithm that reconstructs the signal and infers the multiplicity of each pole.

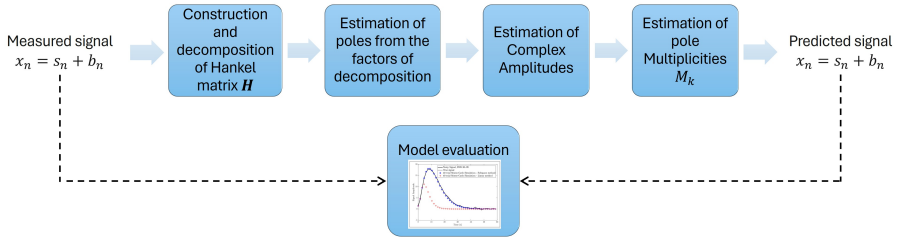


Figure 3.1: Schematic of the proposed method: Starting from a noisy measured signal assumed to follow an exponential polynomial model with known order, the method constructs a Hankel data matrix, estimates poles and complex amplitudes using subspace techniques, infers the multiplicity structure, reconstructs the signal, and compares it with the original to assess estimation accuracy.

The contributions of this case study are twofold:

- **Generalised Exponential Polynomial Model**: An extended exponential polynomial model is introduced that incorporates multiple poles, inspired by the structure of differential equations. Existing subspace-based methods are utilised to estimate the pole locations and complex amplitudes of the underlying signal.
- **Multiplicity Structure Estimation**: Building on the estimated pole locations, a systematic search-based algorithm is developed to determine the multiplicity structure when poles coincide. The effectiveness of the proposed approach is evaluated through a series of numerical experiments, assessing the performance of the systematic search-based method.

3.2 Mathematical Preliminaries

This section reviews essential concepts from linear systems theory that support the signal model developed in this chapter. It begins with a summary of homogeneous linear difference equations and their general solution, which describe the evolution of autonomous LTI systems. This is followed by an overview of how system dynamics are governed by the eigenstructure of the state-transition matrix, including cases involving repeated or complex eigenvalues. This leads naturally to the role of diagonalisation and Jordan canonical form in characterising the system's behaviour and motivating the form of the signal model.

3.2.1 Homogeneous Linear Difference Equation

The signal model considered in this chapter can be viewed as the output of a discrete-time autonomous LTI system, excited by initial conditions and affected by noise. Specifically, such systems can be described by a homogeneous linear difference equation:

$$a_0 x_n + a_1 x_{n-1} + a_2 x_{n-2} + \cdots + a_R x_{n-R} = 0, \quad \text{for } n \geq R, \tag{3.1}$$

where a_i are constant coefficients and R is the order of the equation. The right-hand side being zero makes the equation homogeneous, and the system response is entirely determined by the initial conditions $x_0, x_1, \ldots, x_{R-1}$.

The general solution to (3.1) is determined by the roots (poles) of its characteristic polynomial:

$$a_0 \lambda^R + a_1 \lambda^{R-1} + \cdots + a_{R-1}\lambda + a_R = 0 \tag{3.2}$$

If all roots $\lambda_1, \ldots, \lambda_R$ are distinct, the solution is a linear combination of exponentials:

$$x_n = C_1 \lambda_1^n + C_2 \lambda_2^n + \cdots + C_R \lambda_R^n. \tag{3.3}$$

If some roots are repeated, the solution takes the form of polynomially modulated exponentials:

$$x_n = (D_0 + D_1 n + \cdots + D_{m-1} n^{m-1})\lambda_1^n + (E_0 + E_1 n + \cdots + E_{p-1} n^{p-1})\lambda_2^n + \ldots. \quad (3.4)$$

where λ_1, λ_2 are the roots with multiplicities m and p, respectively. This behaviour forms the mathematical basis of the exponential polynomial signal model used in this work (Section 3.4).

Example 6. Consider the following second-order homogeneous linear difference equations with initial conditions $x_0 = 4,\ x_1 = 9$:

	Distinct Roots	**Repeated Roots**
Equation:	$x_n - 2x_{n-1} - 8x_{n-2} = 0$	$x_n + 4x_{n-1} + 4x_{n-2} = 0$
Characteristic equation:	$\lambda^2 - 2\lambda - 8 = 0$	$\lambda^2 + 4\lambda + 4 = 0$
Roots:	$\lambda = -2,\ 4$	$\lambda = -2,\ -2$
General solution:	$x_n = -2^n C_1 + 4^n C_2$	$x_n = (-2)^n (D_0 + D_1 n)$
Note:	C_1, C_2 from initial values	D_0, D_1 from initial values

The initial conditions provide the information needed to solve for the unknown constants. This structure directly parallels the exponential and exponential-polynomial components of the signal models considered later in this chapter (Section 3.3 and Section 3.4).

3.2.2 Eigenstructure of Linear Systems: Diagonalisation and Jordan Canonical Form

Building on the homogeneous linear difference equation introduced in the previous section, this section explores how the solution structure is determined by the eigenvalues of the underlying system matrix. This connection arises naturally when the system is expressed in the state-space form of an autonomous LTI system.

Without loss of generality, divide the recurrence relation (3.1) by $a_0 \neq 0$:

$$x_n = -a_1' x_{n-1} - a_2' x_{n-2} - \cdots - a_R' x_{n-R}, \quad \text{where } a_k' = \frac{a_k}{a_0}. \quad (3.5)$$

Define the state vector $\boldsymbol{x}_n \in \mathbb{R}^R$ to collect the current and past $(R-1)$ values of the sequence:

$$\boldsymbol{x}_n = \begin{bmatrix} x_n & x_{n-1} & x_{n-2} & \ldots & x_{n-R+1} \end{bmatrix}^\top. \quad (3.6)$$

The evolution of the system can then be described by the first-order linear state equation:

$$\boldsymbol{x}_{n+1} = \boldsymbol{A}\boldsymbol{x}_n, \tag{3.7}$$

where the state transition matrix $\boldsymbol{A} \in \mathbb{R}^{R\times R}$ has companion matrix form:

$$\boldsymbol{A} = \begin{bmatrix} -a'_1 & -a'_2 & \dots & -a'_{R-1} & -a'_R \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}. \tag{3.8}$$

This matrix structure shifts the state vector down and computes the new leading value using the recurrence relation.

To extract the observable scalar output, define:

$$y_n = \boldsymbol{c}^\top \boldsymbol{x}_n, \quad \text{with } \boldsymbol{c} = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}^\top. \tag{3.9}$$

which selects the first entry of the state vector, i.e., $y_n = x_n$. The solution behaviour of y_n is fully determined by the eigenstructure of $\boldsymbol{A}$.

Even when the characteristic roots (poles) are repeated, the underlying recurrence relation (3.5) and its associated companion matrix form (3.8) remain unchanged. However, the solution structure of the system, governed by the eigenstructure of the companion matrix, transitions from a sum of exponentials to polynomially modulated exponentials when the matrix is not diagonalisable.

If $\boldsymbol{A}$ is diagonalisable, its dynamics are simple to describe: the output y_n becomes a linear combination of complex exponentials of the form λ^n, where λ is an eigenvalue of $\boldsymbol{A}$. However, if $\boldsymbol{A}$ has repeated eigenvalues and does not admit a full set of linearly independent eigenvectors, the system cannot be diagonalised. In this case, solutions contain polynomially modulated exponentials, such as $n^{\ell-1}\lambda^n$, where ℓ is the size of the Jordan block associated with λ. This behaviour is fundamental to the signal models adopted in this chapter.

Diagonalisation and Generalised Eigenvectors:
Matrix $\boldsymbol{A}$ is diagonalisable if it can be written as:

$$\boldsymbol{A} = \boldsymbol{T}\boldsymbol{D}\boldsymbol{T}^{-1}, \tag{3.10}$$

where $\boldsymbol{T}$ is a matrix of linearly independent eigenvectors and $\boldsymbol{D}$ is a diagonal matrix of eigenvalues. Diagonalisability requires that the algebraic multiplicity (number of repetitions of an eigenvalue) equals the geometric multiplicity (dimension of the eigenspace) for all eigenvalues.

Example 7. To illustrate diagonalisability, we use two simple 2×2 matrices (not in companion form) with the same repeated eigenvalue $\lambda = 1$:

Diagonalisable:

$\boldsymbol{A}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Two linearly independent eigenvectors:

$\boldsymbol{t}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\boldsymbol{t}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Diagonalisable via $\boldsymbol{T}^{-1}\boldsymbol{A}_1\boldsymbol{T}$

Not Diagonalisable:

$\boldsymbol{A}_2 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$

One eigenvector only:

$\boldsymbol{t}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ because $\mathrm{Null}(\boldsymbol{A}_2 - \lambda\boldsymbol{I}) = 1$

Requires generalised eigenvector:

$\boldsymbol{t}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ from $\mathrm{Null}(\boldsymbol{A}_2 - \lambda\boldsymbol{I})^2$

Leads to Jordan block form.

Complex and Repeated Eigenvalues:
If the state transition matrix $\boldsymbol{A} \in \mathbb{R}^{R\times R}$ has complex eigenvalues, they appear in conjugate pairs. For instance, if $\lambda = a + ib$ and $\bar{\lambda} = a - ib$, the system exhibits oscillatory behaviour. In the Jordan canonical form, these pairs are represented using 2×2 real blocks of the form:

$$\begin{bmatrix} a & b \\ -b & a \end{bmatrix}.$$

Repeated eigenvalues may or may not lead to a diagonalisable system, depending on the dimension of the corresponding eigenspaces. The following canonical forms illustrate the different possibilities. Note that these are not the original state matrix $\boldsymbol{A}$, but rather its Jordan-equivalent representations:

1. Fully Diagonalisable (algebraic multiplicity = geometric multiplicity):

$$\boldsymbol{A} = \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix} \tag{3.11}$$

Here, $\mathrm{rank}(\boldsymbol{A} - \lambda\boldsymbol{I}) = 0$; the null space has dimension 3, therefore three unique eigenvectors can be obtained.

2. Partially Diagonalisable:

$$\boldsymbol{A} = \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix} \tag{3.12}$$

Now, $\operatorname{rank}(\boldsymbol{A} - \lambda \boldsymbol{I}) = 1$; the null space has dimension 2. One generalised eigenvector is needed.

3. Minimally Diagonalisable:

$$\boldsymbol{A} = \begin{bmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{bmatrix} \tag{3.13}$$

Here, $\operatorname{rank}(\boldsymbol{A} - \lambda \boldsymbol{I}) = 2$, only one eigenvector exists, and two generalised eigenvectors are required.

Jordan Canonical Form:
The Jordan canonical form provides a structured way to represent any square matrix up to similarity. For a $R \times R$ matrix with distinct real eigenvalues $\lambda_1, \ldots, \lambda_m$, a complex pair $\lambda \pm i\omega$, a repeated eigenvalue μ with full eigenspace and a repeated eigenvalue γ with deficient eigenspace, leading to a Jordan form:

$$\begin{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_m \end{bmatrix} & & & & \\ & \begin{bmatrix} \lambda & \omega \\ -\omega & \lambda \end{bmatrix} & & & \\ & & \begin{bmatrix} \mu & 0 \\ 0 & \mu \end{bmatrix} & & \\ & & & \begin{bmatrix} \gamma & 1 \\ 0 & \gamma \end{bmatrix} \end{bmatrix} \tag{3.14}$$

Therefore, although it is not possible to completely diagonalise the matrix, it is possible to attain a block diagonal form.

3.3 Harmonic Retrieval of Simple Poles

Building on the mathematical preliminaries, we now turn to classical parametric signal models. These models often represent a measurement sequence x_n as a sum of damped complex exponentials corrupted by additive WGN. Notably, the structure of such

models aligns with the general solution for linear difference equations with distinct characteristic roots, as described in (3.3):

$$x_n = \sum_{k=1}^{R} (a_k e^{j\phi_k})(e^{\{(-\alpha_k + j2\pi f_k)\Delta t\}n}) + b_n, \tag{3.15}$$

where, j is the imaginary unit, R is a positive integer denoting the model order, $n = 0, \ldots, N-1$ represents the time index with N samples, Δt is the sampling time interval and b_n is WGN. Each exponential component is defined by the amplitude $a_k \in \mathbb{R}_+$, phase $\phi_k \in \mathbb{R}$, frequency $f_k \in \mathbb{R}$, and damping factor $\alpha_k \in \mathbb{R}_+$.

Letting $c_k = a_k e^{j\phi_k} \in \mathbb{C}$ and $z_k = e^{\{(-\alpha_k + j2\pi f_k)\Delta t\}} \in \mathbb{C}\backslash\{0\}$ denote the complex amplitude and pole, respectively, the model simplifies to:

$$x_n = \sum_{k=1}^{R} c_k z_k^n + b_n, \quad n = 0, \ldots, N-1. \tag{3.16}$$

This form is widely used in spectral estimation, modal analysis, and system identification applications. A common assumption is that the poles z_k are distinct, which ensures that the solution can be represented as a linear combination of linearly independent exponential terms.

Subspace-based methods (Kung et al., 1983; Papy et al., 2009) typically operate on a 'Hankel-structured' data matrix $\boldsymbol{H}$, which is constructed using N data samples of x_n from (3.16), where each pole $z_k \in \mathbb{C}$ is distinct (i.e., multiplicity 1), $c_k \in \mathbb{C}$ are complex amplitudes, and b_n is an optional additive noise term. The number of distinct poles is equal to the total model order, i.e., $K = R$. The Hankel matrix $\boldsymbol{H} \in \mathbb{R}^{L \times M}$ is defined as:

$$\boldsymbol{H} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{M-1} \\ x_1 & \iddots & \iddots & x_M \\ \vdots & \iddots & \iddots & \vdots \\ x_{L-1} & \cdots & x_{N-2} & x_{N-1} \end{bmatrix}, \tag{3.17}$$

where both $L, M > R$, with L serving as a hyperparameter and $M = N - L + 1$.

The rationale for using a Hankel matrix is its inherent shift-invariance property, which reveals the exponential structure of the signal and facilitates parameter estimation through matrix factorisation techniques.

Noiseless Case: In the absence of noise ($b_n = 0, \forall n$), $\boldsymbol{H}$ is rank-R and admits a Vandermonde decomposition (VDMD):

$$\boldsymbol{H} = \boldsymbol{V}_L \boldsymbol{\Lambda} \boldsymbol{V}_M^{\top}, \tag{3.18}$$

where:$\boldsymbol{V}_L \in \mathbb{C}^{L\times R}$ and $\boldsymbol{V}_M \in \mathbb{C}^{M\times R}$ are Vandermonde matrices formed from the poles $z_k, k = 1, \ldots, K$:

$$\boldsymbol{V}_L = \begin{bmatrix} 1 & \ldots & 1 \\ z_1 & \ldots & z_K \\ z_1^2 & \ldots & z_K^2 \\ \vdots & \vdots & \vdots \\ z_1^{L-1} & \ldots & z_K^{L-1} \end{bmatrix}, \qquad \boldsymbol{V}_M = \begin{bmatrix} 1 & \ldots & 1 \\ z_1 & \ldots & z_K \\ z_1^2 & \ldots & z_K^2 \\ \vdots & \vdots & \vdots \\ z_1^{M-1} & \ldots & z_K^{M-1} \end{bmatrix} \tag{3.19}$$

and $\boldsymbol{\Lambda} = \mathrm{diag}(c_1, c_2, \ldots, c_K)$ contains the complex amplitudes.

This factorisation is foundational for subspace-based methods, as the shift-invariance property of $\boldsymbol{V}_L$ enables recovery of the poles via matrix pencil or eigenvalue decomposition techniques. Let $\boldsymbol{V}_\downarrow$ and $\boldsymbol{V}_\uparrow$ denote matrices obtained by removing the bottom and top row of $\boldsymbol{V}$, respectively. Then, the shift-invariance relation is:

$$\boldsymbol{V}_\downarrow \boldsymbol{Z} = \boldsymbol{V}_\uparrow, \tag{3.20}$$

where $\boldsymbol{Z} = \mathrm{diag}(z_1, z_2, \ldots, z_K)$ is a diagonal matrix of poles. This property forms the basis for many pole estimation techniques, allowing recovery of z_k via eigenvalue decomposition.

Noisy Case: In practical settings, the observed signal (3.16) is contaminated by additive noise ($b_n \neq 0$), and the decomposition (3.18) no longer holds exactly. In this case, the Hankel matrix can be expressed as:

$$\boldsymbol{H} = \boldsymbol{V}_L \boldsymbol{\Lambda} \boldsymbol{V}_M^\top + \boldsymbol{E}, \tag{3.21}$$

where $\boldsymbol{E}$ represents a perturbation due to noise. The matrix $\boldsymbol{H}$ is now of full rank, but its dominant subspace still approximates the column space of $\boldsymbol{V}$.

Subspace-based methods address this by computing a low-rank approximation of $\boldsymbol{H}$ using the SVD. Let $\widehat{\boldsymbol{U}}$ contain the leading K left singular vectors of $\boldsymbol{H}$. Then, even in the presence of noise, $\widehat{\boldsymbol{U}}$ approximately spans the same subspace as the Vandermonde matrix $\boldsymbol{V}$. As a result, a shift-invariance relation similar to (3.20) can be formulated:

$$\widehat{\boldsymbol{U}}_\uparrow = \widehat{\boldsymbol{U}}_\downarrow \widetilde{\boldsymbol{Z}}, \tag{3.22}$$

where $\widetilde{\boldsymbol{Z}}$ is a matrix whose eigenvalues are used as pole estimates $\hat{z}_k$. This relation is typically solved in a total least-squares (TLS) sense, and the Hankel Total Least Squares (HTLS) algorithm (Papy et al., 2009; Morren et al., 2001) is a well-known method that leverages this structure.

Finally, if the signal model includes damped or complex exponentials of the form $x_n \sim e^{\{(-\alpha_k + j2\pi f_k)\Delta t\}n}$, the estimated poles can be mapped to physical parameters via:

$$\lambda_k = \exp\{(-\hat{\alpha}_k + j2\pi\hat{f}_k)\Delta t\}, \tag{3.23}$$

where $\hat{\alpha}_k$ and $\hat{f}_k$ denote the estimated damping factors and frequencies, respectively.

3.4 Generalised Signal Model

The assumption that the poles are distinct (simple), as in (3.16), is often too restrictive in practical scenarios where repeated poles naturally arise, for example, in critically damped systems or structures exhibiting repeated vibrational modes. In such cases, the solution includes additional polynomial modulation terms, leading to a more general exponential polynomial model (Smith, 2007; Niculescu et al., 2021).

A key contribution of this study is the generalisation of the classical signal model (3.16) to explicitly account for repeated poles. Notably, this generalised model reduces to the standard sum-of-exponentials form when all poles are simple, thereby unifying both simple and repeated pole scenarios within a single framework.

To capture this richer class of behaviours, we extend (3.16) to the exponential polynomial model, which corresponds to the general solution of linear difference equations with repeated characteristic roots, as described in (3.4):

$$x_n = \sum_{k=1}^{K} \sum_{\ell=1}^{m_k} n^{\ell-1} c_{k\ell} z_k^n + b_n, \quad n = 0, \ldots, N-1, \tag{3.24}$$

where $z_k \in \mathbb{C}$ is a distinct pole with multiplicity $m_k \in \mathbb{N}$, $c_{k\ell} \in \mathbb{C}$ is the coefficient for the ℓth repeated mode, K is the number of distinct poles, and the total model order is $R = \sum_{k=1}^{K} m_k$.

This structure directly corresponds to the general solution of homogeneous linear difference equations with constant coefficients, as developed in Section 3.2.1. Specifically, if a characteristic polynomial

$$P(z) = \sum_{r=0}^{R} a_r z^{R-r}$$

has a root z_k of multiplicity m_k, then the general solution includes terms of the form $n^{\ell-1} z_k^n$ for $\ell = 1, \ldots, m_k$ (Zill and Cullen, 2008).

Example 8. Consider a second-order homogeneous difference equation: $x_n - 1.6x_{n-1} + 0.64x_{n-2} = 0$. The characteristic polynomial is $z^2 - 1.6z + 0.64 = (z - 0.8)^2$, which has a repeated root at $z = 0.8$ with multiplicity 2. Therefore, the general solution is $x_n = (c_{11} + c_{12}n) \cdot 0.8^n$, which fits exactly into the exponential polynomial model given in (3.24), where repeated poles require additional terms involving powers of n.

Figure 3.2 illustrates the time-domain signal responses (top row) and the corresponding pole locations on the complex plane (bottom row) for four different third-order signal models, each showcasing a unique pole configuration. For comparison, the amplitudes are set to one for all cases.

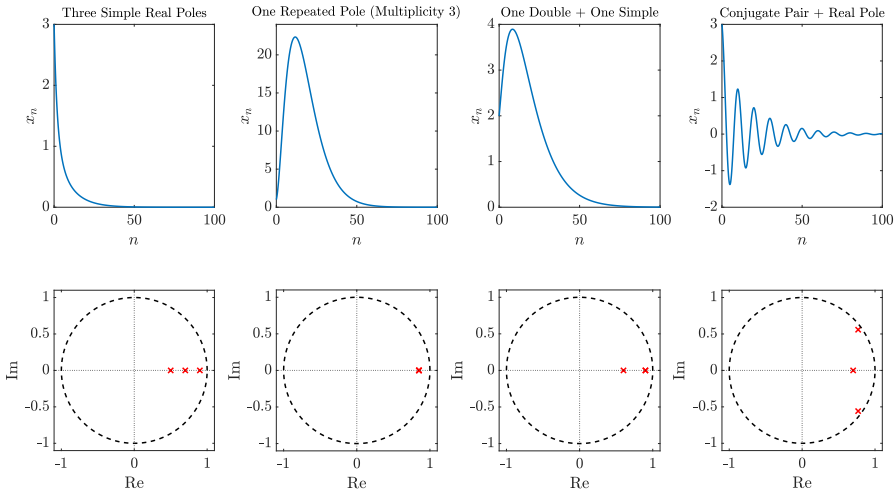


Figure 3.2: The figure includes signal sequences obtained from a third-order signal model that considers different combinations of poles. The corresponding pole locations inside the unit circle are marked in ×.

• Case 1: This signal consists of three distinct real poles $\{0.9, 0.7, 0.5\}$. The resulting sequence is a weighted sum of decaying exponentials with different rates. As shown in the plot, the signal decays quickly and monotonically to zero. In the pole plot, all poles lie on the real axis inside the unit circle.

• Case 2: Here, a single real pole at 0.85 is repeated three times, leading to an exponential polynomial of the form $x_n = (c_{11} + c_{12}n + c_{13}n^2)z^n$, where $c_{11} = c_{12} = c_{13} = 1$.

The signal exhibits a strong transient effect, with an initial rise due to the n-weighted terms, followed by a decay. In the pole plot, only one pole appears, but with algebraic multiplicity three, manifesting as a single point on the real axis.

• Case 3: This case uses a simple pole at 0.9 and a double pole at 0.6 (multiplicity 2). The signal combines a fast (0.6) and a slower (0.9) decaying component, with mild non-monotonic behaviour due to the polynomial weighting of the repeated term. The pole plot shows two distinct points on the real axis, with the repeated pole contributing twice to the model order.

• Case 4: This configuration includes a complex conjugate pair $0.95e^{\pm j2\pi f}$ with $f = 0.1$, and a real pole at 0.7. The signal exhibits oscillatory decay due to the complex and is modulated by the exponential decay of both real and complex components. The pole plot clearly shows a symmetric complex conjugate pole pair and a real pole, all inside the unit circle, indicative of a stable and oscillatory signal.

Note that the pole-zero maps in Figure 3.2 display only the poles of the signal models, with no visible zeros. This is because the signals under consideration are generated by autonomous systems (systems without external input), where the signal evolves according to the system's internal dynamics. In such cases, there is no input-output relationship, and hence, the concept of zeros (as defined in the transfer function relating input to output) does not apply. Poles are intrinsic to the system, determined by the roots of the characteristic polynomial, and govern the natural modes of the system's evolution. Zeros, on the other hand, are extrinsic. They characterise how an input is shaped by the system to produce an output. For example, the vibrations of a piano string after being struck are governed by its natural frequencies, the poles. Once the initial impulse (the strike) has occurred, the system is autonomous, and the resulting free vibration contains no notion of input-output zeros. Thus, in the context of autonomous signal models, zeros are not defined or relevant.

An alternative but equivalent perspective for the signal model (3.24) comes from the inverse z-transform of rational transfer functions. If the z-domain representation of a system output is:

$$X(z) = \frac{N(z)}{D(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{\prod_{k=1}^{K} (1 - z_k^{-1} z)^{m_k}},$$

where $N(z)$ and $D(z)$ are the numerator and denominator polynomials respectively, $z_k \in \mathbb{C}$ are the poles of the system, $m_k \in \mathbb{N}$ is the multiplicity of each pole z_k. Let us now analyse the inverse z-transform of $X(z)$, which yields the time-domain signal x_n.

Simple Poles: If all poles z_k are distinct and have multiplicity one (i.e., $m_k = 1, \forall k \geq 1$), then the partial fraction expansion of $X(z)$ leads to terms of the form:

$$\frac{c_k}{1 - z_k^{-1} z} \quad \longleftrightarrow \quad x_n^{(k)} = c_k z_k^n,$$

in the time domain. This directly corresponds to a sum of decaying or oscillatory exponentials.

Repeated Poles: If some poles have multiplicity $m_k > 1$, then the partial fraction expansion includes higher-order terms:

$$\sum_{\ell=1}^{m_k} \frac{c_{k\ell}}{(1 - z_k^{-1} z)^\ell} \quad \longleftrightarrow \quad x_n^{(k)} = \sum_{\ell=1}^{m_k} c_{k\ell} n^{\ell-1} z_k^n.$$

Each repeated pole contributes a *polynomial-in-n* term multiplied by an exponential z_k^n. This is precisely the structure described in the exponential polynomial model of (3.24).

Complete Signal: Summing over all poles z_k, each with their respective multiplicity m_k, the full signal takes the form:

$$x_n = \sum_{k=1}^{K} \sum_{\ell=1}^{m_k} c_{k\ell} n^{\ell-1} z_k^n,$$

which matches exactly the expression in (3.24), confirming that the exponential polynomial model is nothing but the inverse z-transform of a rational function with repeated poles.

This relationship is fundamental because it links two key perspectives (Oppenheim and Schafer, 2009):

- The *system-theoretic viewpoint*, where the dynamics are governed by poles and zeros of transfer functions, and
- The *parametric signal modelling viewpoint*, where the observed sequence is decomposed into known functional bases (here, exponential-polynomial terms).

The exponential polynomial signal structures naturally arise in various applications: high-resolution spectral estimation where closely spaced or repeated modes appear, structural health monitoring and modal analysis in mechanical systems, and multi-exponential decay modelling in MRI or fluorescence signals. The exponential polynomial model generalises the classical sum of complex exponentials, allowing richer dynamics and forming the basis for the system identification methods developed in the following sections.

3.5 Matrix Form of Generalised Signal Model

This section discusses the matrix-based framework used in subspace-based methods for signal parameter estimation, for the general case of repeated poles. The distinction

between noiseless and noisy cases is made explicit, as the underlying decomposition structure depends heavily on whether the observed signal contains perturbations.

To handle repeated poles, the signal model x_n is generalised to the exponential polynomial form (3.24). The Hankel matrix $\boldsymbol{H} \in \mathbb{R}^{L\times M}$ is constructed as before (3.17).

Noiseless Case: In the noiseless case, the Hankel matrix $\boldsymbol{H}$ admits a generalised VDMD, also called confluent VDMD, which serves as the theoretical foundation for estimating the signal's poles and amplitudes:

$$\boldsymbol{H} = \boldsymbol{T}_L \boldsymbol{D} \boldsymbol{T}_M^{\top}, \tag{3.25}$$

This decomposition arises from the fact that the proposed signal model is a generalised sum of damped exponentials modulated by polynomials, as required for representing repeated poles. This structure naturally leads to a structured low-rank factorisation of $\boldsymbol{H}$. The left and right factor matrices, $\boldsymbol{T}_L$ and $\boldsymbol{T}_M$, are confluent Vandermonde matrices containing monomial-exponential terms tied to the poles and their multiplicities. The central matrix $\boldsymbol{D}$ encodes the polynomial coefficients of the amplitude functions. When all poles are simple (i.e., multiplicity one), the model simplifies to the classical sum-of-exponentials form (3.16), and the factor matrices reduce to standard Vandermonde matrices (3.19) and a diagonal matrix $\boldsymbol{\Lambda}$.

This structure allows us to decouple the pole locations from the amplitude parameters and provides a clear algebraic path to recover the model parameters through eigenvalue decompositions or structured matrix pencil methods. It also justifies the use of subspace techniques, which seek a basis for the column space (or row space) of $\boldsymbol{H}$ in terms of Vandermonde-like components.

The decomposition serves as the foundation for theoretically estimating the complex amplitudes and poles of the signal for $k = 1, \ldots, K$ and $\ell = 1, \ldots, M_k$. Here, $\boldsymbol{D}$ is a $R \times R$ block-diagonal matrix, consisting of K blocks. The kth block $\boldsymbol{D}_k$ of $\boldsymbol{D}$ has dimensions $m_k \times m_k$ and is structured as follows:

$$\boldsymbol{D}_k = \begin{bmatrix} c_{k1} & c_{k2} & c_{k3} & c_{k4} & \ldots & c_{km_k} \\ c_{k2} & 2c_{k3} & 3c_{k4} & \iddots & \iddots & 0 \\ c_{k3} & 3c_{k4} & \iddots & \iddots & \iddots & \vdots \\ c_{k4} & \iddots & \iddots & \iddots & & \vdots \\ \vdots & \iddots & \iddots & & & \vdots \\ c_{km_k} & 0 & \ldots & \ldots & \ldots & 0 \end{bmatrix},$$

$$[\boldsymbol{D}_k]_{ij} = \begin{cases} \binom{i+j-2}{i-1} c_{kl}, & \text{if } i+j \leq m_k+1, \\ 0, & \text{if } i+j > m_k+1, \end{cases} \tag{3.26}$$

where $\forall k \in \{1, 2, \ldots, K\}$, $\ell = 1, \ldots, m_k$, and $i, j = 1, \ldots, m_k$, are local positional indices.

Note: When all poles are simple (i.e., $m_k = 1, \forall k$), each block $\boldsymbol{D}_k$ reduces to a scalar $[c_{k1}]$, and the overall decomposition in (3.25) simplifies to the classical Vandermonde-based representation (3.18) used in the standard sum-of-exponentials model.

The matrices $\boldsymbol{T}_L$ and $\boldsymbol{T}_M$ in (3.25) are confluent Vandermonde matrices with dimensions $L \times R$ and $M \times R$, respectively, each containing K confluent blocks. These matrices arise from the derivatives of exponential terms in the presence of pole multiplicities, ensuring that all linearly independent contributions from repeated poles are captured. The kth confluent blocks of Vandermonde matrices are given by:

$$[\boldsymbol{T}_L]_k = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ z_k & z_k & z_k & \ldots & z_k \\ z_k^2 & 2z_k^2 & 4z_k^2 & \ldots & 2^{m_k-1} z_k^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ z_k^{(L-1)} & (L-1) z_k^{(L-1)} & (L-1)^2 z_k^{(L-1)} & \ldots & (L-1)^{m_k-1} z_k^{(L-1)} \end{bmatrix},$$

$$[\boldsymbol{T}_M]_k = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ z_k & z_k & z_k & \ldots & z_k \\ z_k^2 & 2z_k^2 & 4z_k^2 & \ldots & 2^{m_k-1} z_k^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ z_k^{(M-1)} & (M-1) z_k^{(M-1)} & (M-1)^2 z_k^{(M-1)} & \ldots & (M-1)^{m_k-1} z_k^{(M-1)} \end{bmatrix} \tag{3.27}$$

Again, in the case of simple poles ($m_k = 1$), the confluent blocks reduce to standard Vandermonde columns, with each pole contributing a single exponential term. This illustrates that the generalised model seamlessly includes the classical case as a special instance.

A crucial aspect of the decomposition is the shift-invariance property, which is used to estimate the pole locations. This property is expressed in the form:

$$\boldsymbol{T}_{\uparrow} = \boldsymbol{T}_{\downarrow} \boldsymbol{Z}, \tag{3.28}$$

which indirectly estimates the signal parameters in (Morren et al., 2001) for the simple poles case (3.20). However, the same property also holds for repeated poles estimation. Here, $\boldsymbol{T}_{\uparrow}$ and $\boldsymbol{T}_{\downarrow}$ are matrices formed by removing the first and last row of $\boldsymbol{T}$, respectively.

$\boldsymbol{Z}$ is an $R \times R$ block-diagonal matrix, where each block corresponds to the multiplicity of a pole. This relationship implies that the column space of the Hankel matrix exhibits a predictable transformation under shifting, which is encoded in the matrix $\boldsymbol{Z}$.

The kth block $\boldsymbol{Z}_k$ is an upper triangular Pascal matrix of dimension $m_k \times m_k$, defined as:

$$\boldsymbol{Z}_k = \begin{bmatrix} z_k & z_k & z_k & \dots & z_k \\ 0 & z_k & 2z_k & \dots & \vdots \\ \vdots & \ddots & z_k & \dots & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & z_k \end{bmatrix},$$

$$[\boldsymbol{Z}_k]_{ij} = \begin{cases} \binom{j-1}{i-1} z_k, & \text{if } i \leq j, \\ 0, & \text{if } i > j, \end{cases} \tag{3.29}$$

where $i, j = 1, \dots, m_k$, are local positional indices.

In the simple pole case, $\boldsymbol{Z}_k$ again reduces to the scalar z_k, and the overall block-diagonal matrix $\boldsymbol{Z}$ becomes a diagonal matrix (3.20). This reveals how the generalised framework accounts for both repeated and simple poles in a structurally consistent manner.

The use of Pascal matrices stems from the recurrence relations between shifted monomials and directly encodes the way repeated poles influence the temporal evolution of the signal. Solving for $\boldsymbol{Z}$ from (3.28) enables indirect recovery of the poles z_k, and their multiplicities m_k are reflected in the size of the blocks.

Noisy Case: In the presence of noise, the decomposition in (3.25) holds only approximately:

$$\boldsymbol{H} = \boldsymbol{T}_L \boldsymbol{D} \boldsymbol{T}_M^\top + \boldsymbol{E}, \tag{3.30}$$

where $\boldsymbol{E}$ is the noise perturbation. As with the simple poles case, subspace-based methods apply SVD to $\boldsymbol{H}$ to estimate the dominant signal subspace, from which the pole structure can be indirectly inferred.

To this end, the singular value decomposition (SVD) is applied to the Hankel matrix $\boldsymbol{H}$, yielding a rank-R approximation:

$$\boldsymbol{H} \approx \boldsymbol{H}_R = \boldsymbol{U}_R \boldsymbol{\Sigma}_R \boldsymbol{V}_R^H, \tag{3.31}$$

where $\boldsymbol{U}_R \in \mathbb{C}^{L \times R}$ and $\boldsymbol{V}_R \in \mathbb{C}^{M \times R}$ contain the leading R left and right singular vectors, and $\boldsymbol{\Sigma}_R \in \mathbb{R}^{R \times R}$ is a diagonal matrix of the dominant singular values. The

accuracy of this approximation, and hence of the parameter estimates, depends on the choice of L and the signal-to-noise ratio.

To estimate the pole locations, the truncated left singular matrix $\boldsymbol{U}_R$ is exploited via its shift-invariance property. Specifically, a linear system of the form:

$$\boldsymbol{U}_{R\downarrow}\widetilde{\boldsymbol{Z}} = \boldsymbol{U}_{R\uparrow}, \tag{3.32}$$

is solved for $\widetilde{\boldsymbol{Z}} \in \mathbb{C}^{R\times R}$, where $\boldsymbol{U}_{R\uparrow}$ and $\boldsymbol{U}_{R\downarrow}$ are obtained by removing the first and last rows of $\boldsymbol{U}_R$, respectively. The eigenvalues of $\widetilde{\boldsymbol{Z}}$ give the estimated poles $\hat{z}_k$, $k = 1, \ldots, R$, which encode both the pole locations and their multiplicities. In practice, repeated poles manifest as clusters of closely spaced eigenvalues. These can be grouped and averaged to estimate the true (distinct) poles z_k, as described in Section 3.7.

Once the poles are estimated, the complex amplitude coefficients $\hat{c}_{k\ell}$, for $k = 1, \ldots, K$ and $\ell = 1, \ldots, m_k$, are computed using the SVD-based approximation and the confluent Vandermonde structure of the model. The estimated Hankel matrix can be written as:

$$\boldsymbol{H} \approx \boldsymbol{U}_R\boldsymbol{\Sigma}_R\boldsymbol{V}_R^H = (\boldsymbol{T}_L\boldsymbol{B}_1)\boldsymbol{\Sigma}_R(\boldsymbol{B}_2\boldsymbol{T}_M^\top), \tag{3.33}$$

where $\boldsymbol{T}_L$ and $\boldsymbol{T}_M$ are confluent Vandermonde matrices constructed using the estimated average poles, and $\boldsymbol{B}_1 = \boldsymbol{T}_L^\dagger\boldsymbol{U}_R$, $\boldsymbol{B}_2 = (\boldsymbol{T}_M^\dagger\boldsymbol{V}_R)^\top$ are transformation matrices.

The central product $\boldsymbol{D} = \boldsymbol{B}_1\boldsymbol{\Sigma}_R\boldsymbol{B}_2$ yields a block-diagonal matrix with structure defined in (3.26). Each block corresponds to a repeated pole, and from this structure, the individual amplitude coefficients $\hat{c}_{kl}$ are extracted. This process allows the complete recovery of the exponential polynomial model, even in the presence of moderate noise.

3.6 Proposed Model vs. Polynomial Amplitude Complex Exponentials Model

The problem of estimating poles with multiplicities greater than one has also been addressed using the Polynomial Amplitude Complex Exponentials (PACE) model (Roland et al., 2006). This model adopts a different parametrisation based on a polynomial amplitude structure and is expressed as:

$$x_n = \sum_{k=1}^{K}\sum_{\ell=1}^{m_k} F_{\ell-1}(n)c'_{k\ell}z_k^{n-\ell+1} + b_n, \quad n = 0, \ldots, N-1, \tag{3.34}$$

where $c'_{k\ell}$ denotes the complex amplitude of the PACE model and $F_{\ell-1}(n)$ is a polynomial of degree $(\ell-1)$, defined by:

$$F_{\ell-1}(n) = \begin{cases} 1, & \text{if } \ell = 1, \\ \dfrac{1}{(\ell-1)!} \displaystyle\prod_{i=0}^{\ell-2} (n-i), & \text{if } \ell > 1. \end{cases} \tag{3.35}$$

Apart from the polynomial basis functions and amplitude parameterisation, the remaining terms in (3.34) are structurally identical to those in the proposed model (3.24). The primary distinction between the two models lies in how they represent the polynomial modulation of repeated poles through their complex amplitudes. While both models share the same set of poles $\{z_k\}$ and multiplicities $\{m_k\}$, their amplitude parameters are related through a transformation that arises from the difference in basis representation.

Since both models describe the same signal sequence, a relationship can be established between their parameter sets through matrix equivalence. Specifically, for each kth pole, the complex amplitudes are related by the following transformation:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ & z_k & z_k & z_k & \dots & z_k \\ & & 2!z_k^2 & 6z_k^2 & \dots & \vdots \\ & & & 3!z_k^3 & \dots & \vdots \\ & & & & \ddots & \vdots \\ & & & & & (m_k-1)!z_k^{(m_k-1)} \end{bmatrix} \begin{bmatrix} c_{k1} \\ c_{k2} \\ \vdots \\ c_{km_k} \end{bmatrix} = \begin{bmatrix} c'_{k1} \\ c'_{k2} \\ \vdots \\ c'_{km_k} \end{bmatrix}, \tag{3.36}$$

This transformation highlights that the two models are algebraically equivalent in representing repeated poles, though they adopt different basis functions and parametrisation strategies.

3.7 Estimation of Pole Multiplicity

Following the estimation of poles and complex amplitudes as discussed in Section 3.5, this section focuses on identifying the multiplicity structure associated with the estimated poles. This is a critical step when modelling signals with repeated poles, where the true system may include both simple and repeated modes whose individual multiplicities are unknown. To ensure numerical stability, the subspace-based method introduced earlier is used within a systematic search-based framework, as outlined in Algorithm 2. An illustrative example demonstrating its effectiveness is provided in Section 3.8.1.

We assume that the model order R is known, i.e., the signal comprises a total of R exponential components, which may include repeated poles. The goal is to determine the distribution of these R components among K distinct poles, each potentially with a multiplicity m_k such that $\sum_{k=1}^{K} m_k = R$.

The signal samples x_n are first arranged into a Hankel matrix $\boldsymbol{H}$, and poles are estimated using the subspace-based method with SVD and the shift-invariance property (refer to Section 3.5). However, in this section, the primary objective is not to refine the pole estimates, but rather to identify the correct multiplicity structure that best matches the observed data.

Given R estimated poles, there are a total of 2^{R-1} possible groupings that represent different multiplicity structures. These include the case where all poles are distinct, all poles are identical, or any combination in between.

Example 9. For $R = 3$, the possible multiplicity structures are:

- Three distinct simple poles: $M_k = [1, 1, 1]$;
- One double pole and one simple pole: $M_k = [2, 1]$ or $[1, 2]$;
- A single pole with multiplicity three: $M_k = [3]$.

In practice, repeated poles may not appear consecutively in the estimated list. To ensure consistency in how multiplicities are assigned, the estimated poles are sorted based on their magnitude. This heuristic ensures that repeated poles appear adjacent in the array, which reduces the number of candidate multiplicity structures S and enables an efficient search. However, sorting by magnitude may fail when repeated poles share the same magnitude but differ in phase. Such cases arise when repeated poles lie symmetrically on a circle in the complex plane. To address this, a clustering strategy such as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is applied to group estimated poles based on spatial proximity in the complex plane, not just magnitude.

Definition 3.7.1 (DBSCAN)**.** DBSCAN is a non-parametric clustering algorithm particularly suited for grouping poles in the complex plane. DBSCAN forms clusters by identifying core points with a sufficient number of neighbouring points within a specified distance ϵ. Points within ϵ of a core point are included in the cluster, even if they are not themselves core points.

This approach effectively groups closely located poles, while naturally treating outliers as noise. It does not require specifying the number of clusters in advance, which is

particularly advantageous in this context.

Each candidate multiplicity structure in the set S is evaluated using the following steps:

1. Group and average the pole estimates according to the current multiplicity structure.
2. Construct the corresponding confluent Vandermonde matrix using the averaged poles and multiplicities.
3. Estimate the complex amplitudes via SVD and least-squares approximation.
4. Reconstruct the signal from these estimates and compute the MSE against the original signal data.

The multiplicity structure that yields the lowest MSE is selected as the most likely configuration, and the corresponding pole and amplitude estimates are retained.

The number of multiplicity structures in S grows exponentially with R, making exhaustive search impractical for large R. Clustering helps alleviate this by reducing the search space to only those structures that are supported by observable clusters. This principle is illustrated in the following two examples:

- **Case 1:** Consider true real-valued poles at 0.89 and 0.9 with $M_k = [2, 2]$. Suppose the estimated poles are $\{0.89 \pm 0.1i,\ 0.9 \pm 0.1i\}$. The following multiplicity groupings are possible:
  - Sorting the poles by their magnitude gives two groups centred at approximately 0.8956 and 0.9055, yielding $M_k = [2, 2]$, which reflects the true multiplicity structure.
  - DBSCAN with small ϵ treats all estimated poles as simple poles, resulting in $M_k = [1, 1, 1, 1]$.
  - DBSCAN with significantly large ϵ incorrectly groups based on imaginary part (e.g., top vs. bottom half of complex plane), resulting in $M_k = [2, 2]$, but with incorrect pole locations (e.g., $0.895 \pm 0.1i$), leading to a signal reconstruction mismatch.

  In this case, magnitude-based grouping preserves true structure.

- **Case 2:** Consider complex-conjugate poles at $0.9985 \pm 0.0540i$ with $M_k = [3, 3]$. Suppose the estimated poles are: $\{0.995 \pm 0.052i, 0.998 \pm 0.058i, 1.003 \pm 0.054i\}$. The following cases are possible:
  - Sorting based on magnitude gives three clusters at $0.9964, 0.9997, 1.0045$ with $M_k = [2, 2, 2]$, which misrepresents the true structure.

- DBSCAN with small ϵ treats all poles as distinct.
- DBSCAN with appropriate ϵ gives two clusters $0.9987 \pm 0.0547i$ of $M_k = [3, 3]$, which matches the ground truth and reconstructs the signal most accurately.

In this case, clustering-based multiplicity estimation preserves the true structure.

In practice, the true multiplicity structure is unknown, and relying solely on magnitude or clustering heuristics can lead to incorrect interpretations. The two examples above demonstrate that neither sorting nor clustering alone is sufficient in all scenarios. Hence, the proposed algorithm incorporates both magnitude-based and clustering-based grouping strategies and evaluates each resulting multiplicity structure using signal reconstruction error (e.g., MSE). This allows the algorithm to adaptively select the best structure based on data fidelity, without prior knowledge of the true multiplicity. This complete procedure is outlined in Algorithm 2.

Algorithm 2 Subspace-based Method for the Estimation of Multiplicity Structure, Pole Locations and Complex Amplitudes

Input: Signal data x_n of length N, model order R
Output: Poles z_k, complex amplitudes $c_{k\ell}$, multiplicity structure M_k

1: Set Hankel matrix dimensions $L \leftarrow \lfloor N/2 \rceil, M \leftarrow N - L + 1$
2: Construct Hankel matrix $\boldsymbol{H}$ of size $L \times M$
3: Estimate poles using SVD and shift-invariance on $\boldsymbol{H}$
4: Generate candidate multiplicity structures S using:
   - Sorting poles by magnitude
   - Clustering poles using DBSCAN with radius ϵ
5: **for** $i = 1 :$ length(S) **do**
6: Average poles within each group in $S(i)$ to obtain $\hat{z}_k(i, :)$
7: Construct confluent Vandermonde matrix from $\hat{z}_k(i, :)$ and $S(i)$
8: Estimate $\hat{c}_{k\ell}(i, :)$ using SVD and least-squares
9: Reconstruct signal from $\hat{z}_k(i, :), \hat{c}_{k\ell}(i, :)$
10: Compute and store MSE with the original signal
11: **end for**
12: Select index p of minimal MSE
13: **return** $M_k \leftarrow S(p), z_k \leftarrow \hat{z}_k(p, :), c_{k\ell} \leftarrow \hat{c}_{k\ell}(p, :)$

This approach provides an accurate and efficient method for determining the correct pole multiplicities, especially when combined with a suitable clustering strategy.

3.8 Numerical Simulations

This section presents the numerical experiments designed to evaluate the accuracy and robustness of the proposed parameter estimation framework. A key factor influencing the performance of subspace-based methods is the choice of dimensions for the Hankel matrix $\boldsymbol{H}$ constructed from the input signal x_n.

The Hankel matrix $\boldsymbol{H} \in \mathbb{C}^{L\times M}$ is constructed such that $L + M - 1 = N$, where N is the length of the observed signal. For a fixed signal length N, the rank of $\boldsymbol{H}$ (in the noise-free case) is equal to the number of distinct exponential components R present in the signal. However, in the presence of noise, the effective numerical rank and the conditioning of the subspaces used in pole estimation are heavily influenced by the aspect ratio of $\boldsymbol{H}$.

Empirically, it is well-established that using a nearly square Hankel matrix, i.e., choosing L such that $\frac{N}{3} \leq L \leq \frac{2N}{3}$, provides a good balance between numerical stability and estimation accuracy. The maximum achievable rank occurs when $L = \frac{N}{2}$, where the matrix is closest to being square, maximising subspace resolution and minimising ill-conditioning during SVD.

Therefore, all simulations presented here consider $L = \lfloor N/2 \rfloor$ and $M = N - L + 1$ unless otherwise specified. While other values of L within the recommended range may also work well, the estimation accuracy remains largely stable across this interval, with marginal variations. Fixing L to $\frac{N}{2}$ thus provides a reliable and consistent configuration for the Hankel matrix across all scenarios.

3.8.1 Signal Reconstruction using the Estimated Parameters

Several alternative methods for pole estimation were discussed in Section 3.1. Here, the subspace-based method is compared with classical linear prediction techniques, including Prony (1795) and Pisarenko (1973). In the absence of noise, the signal sequence (3.24) satisfies a homogeneous linear recurrence relation of order R. The linear prediction approach first estimates the R coefficients of this recurrence using least squares. These coefficients define a companion matrix whose eigenvalues yield the estimated poles.

The comparison is conducted for several discrete-time signal sequences x_n defined by (3.24), each with a sample size $N = 40$ and a sampling interval $\Delta t = 1.5$. The signals are corrupted by circularly symmetric WGN b_n with an SNR of 30 dB. The ground truth parameters are given in Table 3.1. A Monte-Carlo simulation is performed with $N_{\text{runs}} = 40$ independent trials.

Table 3.1: True parameters of the signals

Case	No. of distinct poles, K	Multiplicity structure, M_k	Pole(s), z_k	Complex amplitudes, c_{kl}
1	1	[3]	$z_1 = 0.8$	$c_{11} = 1.0$, $c_{12} = 1.3$, $c_{13} = 1.5$
2	2	[2,1]	$z_1 = 0.8$, $z_2 = 0.9$	$c_{11} = 1.0$, $c_{12} = 1.3$, $c_{21} = 1.5$

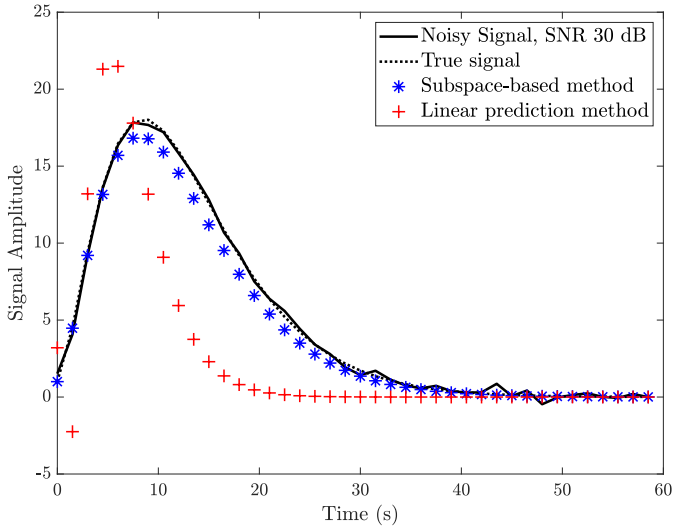


Figure 3.3: Comparison of true, noisy, and reconstructed signals for Case 1 in Table 3.1 using subspace-based and linear prediction methods over 40 Monte Carlo trials. The results highlight the superior performance of the subspace-based method due to its numerically stable SVD approach. The mean relative error in the reconstructed signals (averaged over 40 trials) is approximately 10^{-2} for the subspace-based method and 10^{-1} for the linear prediction method.

The simulation results, shown in Figure 3.3, demonstrate that the subspace-based method yields significantly higher accuracy in pole estimation than the linear prediction method. This improvement is attributed to the robustness of the SVD-based subspace projection. Once the poles are estimated, the corresponding complex amplitudes are computed and used to reconstruct the signal. The reconstruction from the subspace-based estimates closely approximates the true signal, even under moderate noise conditions.

3.8.2 Estimation of True Multiplicity Structure

To evaluate the effectiveness of the proposed multiplicity estimation approach, simulations were conducted for signal cases 1 and 2 from Table 3.1, under varying SNR levels. The results are illustrated in Figures 3.4 and 3.5 for case 1, and in Figure 3.6 for case 2.

For signal case 1, which involves a single pole of multiplicity three, accurate estimation of the true multiplicity structure $M_k = [3]$ is achieved even at low SNR levels, as shown in Figure 3.4. In this setup, all 2^{R-1} possible multiplicity structures (with $R = 3$) are evaluated for SNR values ranging from 10 to 100 dB. A single trial run is used in each case to estimate the poles and complex amplitudes, which are then used for signal reconstruction. The results show that $M_k = [3]$ consistently yields the lowest reconstruction error, confirming it as the most accurate multiplicity structure.

Figure 3.5 further supports this observation by comparing reconstructed signals from all multiplicity candidates at SNR 30 dB. It confirms that the reconstruction with $M_k = [3]$ most closely matches the true signal. Interestingly, even at high SNRs (e.g., 100 dB), the MSE does not reach machine precision, highlighting the sensitivity of pole locations in signal reconstruction, especially for higher-order or repeated poles.

Signal case 2, shown in Table 3.1, is more complex, featuring two distinct poles with a multiplicity structure of $M_k = [2, 1]$. As expected, accurate estimation becomes more challenging in this scenario, especially at lower SNRs. As shown in Figure 3.6, the true multiplicity structure is only reliably recovered when the SNR exceeds 40 dB. This underscores a key limitation: the sensitivity of multiplicity estimation increases with noise and signal complexity.

3.8.3 Real-valued Signal with Noise

This example, adapted from Roland et al. (2006), illustrates the estimation of poles and complex amplitudes for a higher-order real-valued signal using the subspace-based method. The signal reconstruction is then carried out using the proposed model (3.24),

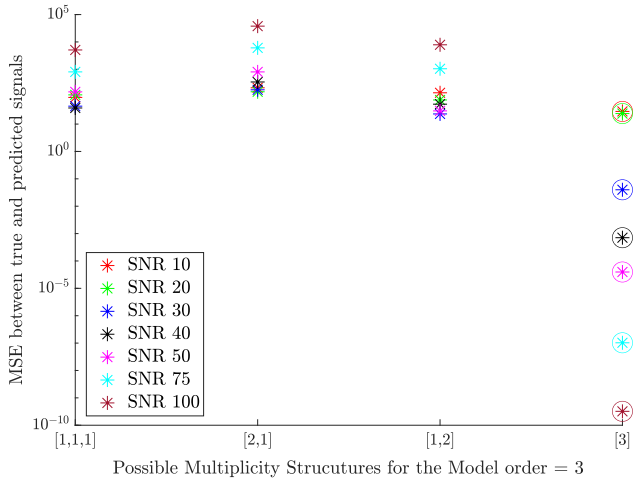


Figure 3.4: MSE between the true and predicted signals across SNR levels for model order $R = 3$. All 2^{R-1} possible multiplicity structures are evaluated, with the estimated structure marked by ◯. Case 1 from Table 3.1 is used, where the true multiplicity is $M_k = [3]$. Accurate identification is achieved even at an SNR of 10 dB.

and the result is compared against the original signal to assess the accuracy of the parameter estimation.

When the prediction polynomial has real coefficients, its roots, i.e., the poles, appear either as real values or in complex conjugate pairs, with matching multiplicities. In the signal described by Roland et al. (2006), the true parameters are: two complex conjugate poles, each repeated three times, resulting in a total of six exponentials. This yields a model order of $R = 6$ with $K = 2$ distinct poles and a multiplicity structure $M_k = [3, 3]$.

Due to the high order and complexity of this signal, accurate parameter estimation becomes challenging under noise. As observed in Section 3.8.2, estimation error increases with multiplicity and noise level. Thus, the SNR is set to 40 dB in this experiment to ensure reasonable accuracy while maintaining the complexity of the problem.

Figure 3.7 shows the pole estimates from the subspace-based method (△) for the noisy signal at 40 dB SNR. The true poles (■) correspond to $z_{1,2} = e^{\pm j2\pi f}$, with $f = 0.0086$. Due to the perturbation introduced by noise, the repeated poles do not align exactly and instead form tightly clustered groups around the true values. This is consistent with the first-order perturbation analysis described in Roland et al. (2006), where repeated

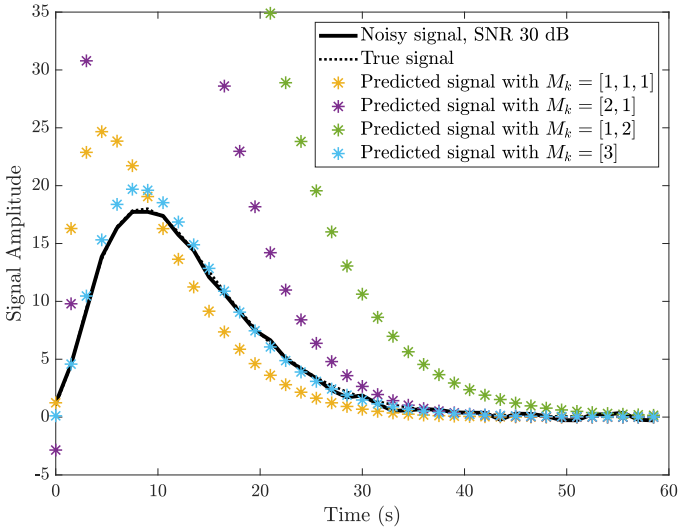


Figure 3.5: Comparison of true, noisy, and predicted signals using the subspace-based method for case 1 in Table 3.1, with multiplicity $M_k = [3]$ and model order $R = 3$: Evaluating pole estimation accuracy. Signals were predicted for various multiplicity scenarios with a fixed model order of $R = 3$. The subspace-based method provided a fairly accurate reconstruction of the true signal when all three poles were identical.

poles appear as distinct due to noise, even if they share the same origin. To address this, the estimated poles are grouped using DBSCAN clustering with a small radius parameter ϵ, and the centroid of each resulting cluster is taken as the representative pole estimate (▲). These cluster-averaged poles are then used to estimate the corresponding complex amplitudes under the assumed multiplicity structure $M_k = [3, 3]$.

Using these estimated poles and amplitudes, the signal is reconstructed according to model (3.24). As seen in the signal comparison plot in Figure 3.7, the reconstructed signal closely follows the true signal, validating the estimation strategy at 40 dB SNR. However, for SNRs lower than 40 dB, the estimated poles deviate more significantly from their true values, leading to a substantial degradation in the reconstruction. This sensitivity to noise justifies limiting the experiments in this example to 40 dB.

3.9 Conclusion

This study presents a generalised model for damped exponential signals with multiple poles, incorporating physically interpretable parameters. In addition, an extension to the

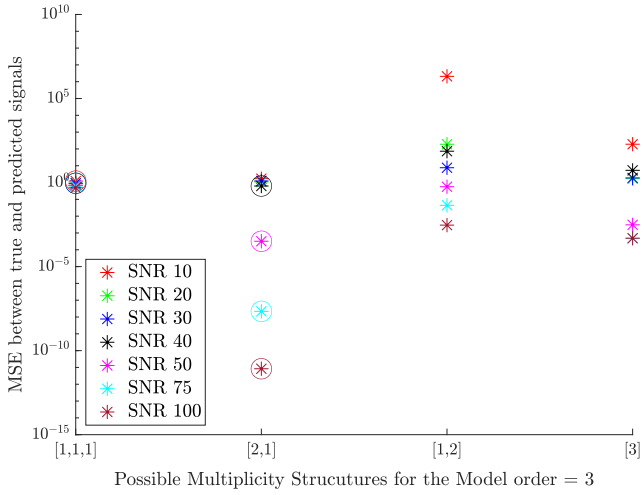


Figure 3.6: MSE between true and reconstructed signals for model order $R = 3$. Each point corresponds to a candidate multiplicity structure; the chosen structure at each SNR is marked by ◯. The true structure $M_k = [2, 1]$ (from case 2 in Table 3.1) is only correctly identified when the SNR is 40 dB or higher.

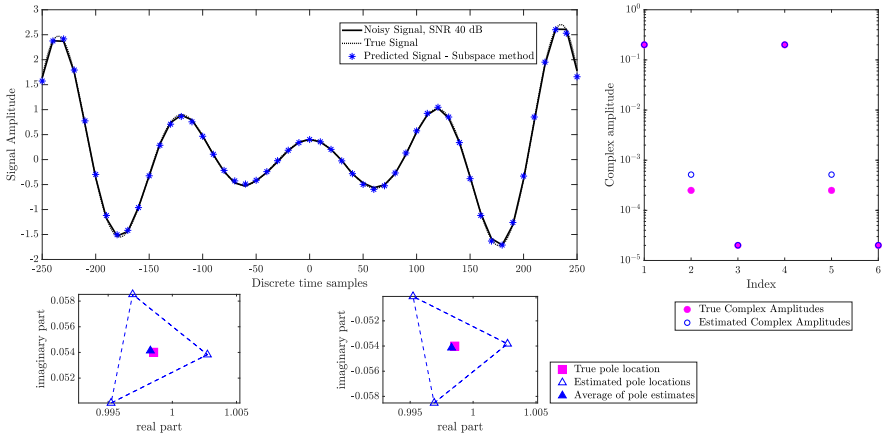


Figure 3.7: Pole estimation using the subspace-based method (△) for an SNR of 40 dB. Also shown are the mean of the estimated poles (▲) and the true pole locations (■). The estimated complex amplitudes (○) are compared with the true amplitudes (●). The predicted signal, reconstructed using the estimated parameters, is shown alongside the true signal. The relative error in the predicted signal is approximately 10^{-1}.

subspace-based method is introduced, enabling direct estimation of poles and complex amplitudes through SVD applied to the data matrix and confluent Vandermonde matrix. A systematic search-based algorithm for estimating the multiplicity structure of poles is also proposed and validated via extensive numerical simulations.

The estimation procedure leverages both magnitude-based sorting and spatial clustering via DBSCAN to identify candidate multiplicity structures. While sorting by magnitude is effective when poles differ significantly in radius, it may fail when repeated poles share the same magnitude but differ in phase. In such cases, DBSCAN, used with a carefully chosen small radius parameter, proves to be effective by clustering poles based on their spatial proximity in the complex plane. This dual strategy allows for a flexible and data-driven approach to group repeated poles under various noise conditions.

The core estimation algorithm evaluates all candidate multiplicity structures (derived from sorting or clustering) and selects the one that yields the lowest reconstruction error (MSE) between the observed and reconstructed signals. Simulations show that this strategy accurately identifies true multiplicity even in the presence of moderate noise.

Numerical Experiments demonstrate that the subspace-based method consistently yields more accurate pole and amplitude estimates compared to classical linear prediction methods, owing to its numerical stability via SVD. Case studies included both synthetic signals with known pole structures and a real-valued signal example taken from the literature. In particular, the latter example highlighted the importance of clustering in scenarios with repeated complex-conjugate poles, where DBSCAN-based averaging improved the pole estimates significantly.

Estimating the correct multiplicity structure serves as a crucial preprocessing step in harmonic retrieval. It constrains the signal model to match the underlying structure and improves parameter accuracy. Incorporating this structure into the subspace-based estimation framework offers a promising direction for improved performance. Further, analytical derivation of Cramér–Rao bounds for the proposed model could offer additional insight into the interplay between estimation variance, model order, and multiplicity.

Chapter 4

Nonlinear Models for System Identification

Abstract: After a detailed introduction of linear modelling approaches and a case study in the previous chapters, we now delve into introducing some of the nonlinear modelling approaches that are more complex yet closer to the real-world systems. In this chapter, an in-depth exploration of nonlinear models commonly used in system identification is presented, focusing on block-oriented structures, polynomial state-space representations and Volterra series. It begins with an overview of Hammerstein, Wiener, and WH models, highlighting their structural features and identification challenges. The chapter then introduces the PNLSS model, with a detailed discussion on how nonlinearity is included and how it affects the stability of the model. The Volterra series is presented as a foundational framework, bridging block-oriented and polynomial models. A key emphasis is placed on the identification of WH systems in the literature, including methods to decouple subsystems, pole allocation strategies using the BLA, and the structural equivalence between PNLSS and block-oriented models. Finally, advanced approaches based on diagonal Volterra kernel representation of the WH system and tensor decompositions are discussed, providing interpretable and efficient tools for identifying complex nonlinear systems.

Note on Chapter Content: This chapter provides a foundational overview of nonlinear models commonly used in system identification tasks. These models serve as the lifeblood of this research, where the identification task is performed by exploiting their strengths. While the content presented here is not an original contribution, it is

compiled and explained based on existing literature, with interpretations and illustrative examples drawn from the author's understanding.

4.1 Foundation and Focus

A key step in nonlinear system identification is selecting an appropriate model structure to represent the system. Numerous options exist, including Nonlinear FIR, NARX models, block-oriented models, NLSS models, neural networks, etc., each with distinct advantages and limitations. Their performance can be assessed by applying them to the same nonlinear system using identical input-output data. This chapter provides a detailed revisit of the nonlinear models introduced in Section 1.3, focusing on their role in system identification.

The identification scope must be narrowed since no single model structure universally represents all nonlinear systems. The primary objective of the thesis is to develop methodologies for identifying nonlinear dynamical systems, particularly those that adhere to block-structured representations. The block-oriented nonlinear models considered in this work consist of linear dynamic and static nonlinear blocks arranged in series. Some widely studied configurations, such as the WH system (Figure 1.5), have specific names. The following sections define the fundamental building blocks of block-oriented models and describe common configurations.

To establish a strong foundation, this chapter also explores relevant works in the literature that address the identification of block-oriented models, highlighting key approaches and challenges.

4.2 Block-oriented Models

As the name suggests, block-oriented models are built with two types of blocks: a linear dynamic block and a static nonlinear block. The linear dynamic block-oriented models the LTI property of a system (refer to Definitions 2.1.1 and 2.1.2). The nonlinearity considered in this research is static and is defined as below:

Definition 4.2.1 (Static nonlinear system)**.** If a system's output is a nonlinear function of the current input, then it is called a static nonlinear system.

For the given input $u(t)$, the corresponding output $y(t)$ is given by:

$$y(t) = \phi(u(t)), \tag{4.1}$$

where $\phi(\cdot)$ is a nonlinear function. The schematic of a static nonlinear block is shown in Figure 4.1.

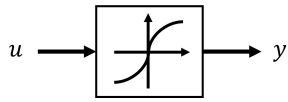


Figure 4.1: A static nonlinear system with input u and output y.

The linear dynamic and static nonlinear blocks can be connected in various fashions, including series, parallel, and feedback connections, leading to several model structures. This thesis is restricted to studying three popular block-oriented structures, namely Hammerstein, Wiener, and WH. These models, though highly structured and parsimonious, do not offer generalisability, unlike other nonlinear models such as the Volterra series, NARX or NLSS models. This lack of generalisability, however, is not a limitation but rather a defining characteristic of these models. Their structured nature allows for efficient identification, interpretability, and reduced computational complexity, making them well-suited for practical applications. Many real-world systems inherently exhibit the block-structured dynamics captured by these models, making them the most appropriate choice for identification and analysis.

For instance, Hammerstein models are widely used in actuator dynamics and power electronics, where a static nonlinearity (such as saturation or dead-zone effects) is followed by a linear system. Wiener models effectively describe biomedical systems, chemical processes, and communication channels, where the system response involves linear dynamics followed by a nonlinear observation process. The WH structure is particularly relevant in audio and RF amplifiers, biological signal processing, and mechanical systems, where nonlinearities occur between two dynamic stages.

By taking advantage of the inherent structure of these models, it is possible to achieve a parsimonious yet highly effective representation of many physical systems. Unlike fully general models, which require a large number of parameters and may suffer from overfitting, these models strike a balance between accuracy, efficiency, and interpretability, making them an essential tool in nonlinear system identification.

4.2.1 Hammerstein Model

The Hammerstein model consists of a static nonlinear block followed in series by a linear dynamic block (see Figure 1.5). The input v is first transformed by a static nonlinear

function, producing an intermediate or internal signal q, which is then processed by the LTI block to generate the final output y. The intermediate signal q is not directly measurable, making system identification more challenging.

Several identification algorithms have been developed for Hammerstein models (Narendra and Gallman, 1966; Crama and Schoukens, 2001; Bai, 2010a,b; Vörös, 1997; Pawlak, 1991; Greblicki, 1996; Schoukens et al., 2007; Hasiewicz and Mzyk, 2009). The Hammerstein model is widely used to represent linear dynamic processes influenced by nonlinear actuators, where actuator saturation effects introduce nonlinearities. This model has been applied in model predictive control for regulating micro-turbines (Wang and Henriksen, 1994; Jurado, 2006).

Many chemical processes can also be effectively described using a Hammerstein model, such as pH neutralisation process (Fruzzetti et al., 1997; Zou et al., 2003), distillation columns, and heat exchangers (Eskinat et al., 1991). In audio processing, Hammerstein models have been used to capture the nonlinearity of loudspeakers combined with a linear room impulse response (Shi et al., 2008).

In telecommunications, Hammerstein systems are commonly found in signal processing architectures, particularly in systems where a mixer (for up or down conversion) is followed by a filter (for band selection). Another example is a nonlinear sensor connected to a transmission line, where the sensor introduces nonlinear distortions before transmission.

The noiseless output y_k of the Hammerstein model at time k is given by:

$$\begin{aligned} y_k &= T(z)q_k, \\ q_k &= f(v_k), \end{aligned} \tag{4.2}$$

where the static nonlinear function $f(\cdot)$ is approximated as a linear combination of polynomial basis functions up to order D, with corresponding coefficients $p_1, \ldots, p_D$. Thus, the static nonlinearity, $q_k = f(v_k)$ is expressed as:

$$q_k = \sum_{j=1}^{D} p_j v_k^j, \tag{4.3}$$

The LTI subsystem is typically modelled using a rational transfer function of the backwards shift operator z^{-1}:

$$T(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{n_b} z^{-n_b}}{a_0 + a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a}}, \tag{4.4}$$

where b_i and a_i are coefficients grouped into the vectors $\boldsymbol{b} \in \mathbb{R}^{n_b+1}$ and $\boldsymbol{a} \in \mathbb{R}^{n_a+1}$, respectively. To ensure a consistent and unique system representation, the first non-zero

element of $\boldsymbol{a}$ is assumed to be positive, and its L^2-norm is normalised to 1. Without this constraint, the coefficients of the numerator and denominator could be arbitrarily scaled, resulting in multiple equivalent parametrisations of the same transfer function. Alternatively, the LTI block can be represented using impulse response coefficients $\boldsymbol{t}$ with a memory length of M_h, as given by:

$$y_k = \sum_{i=0}^{M_\mathrm{h}} t_i q_{k-i}. \tag{4.5}$$

This impulse response formulation is used in this thesis to study the Hammerstein model. Later, the impulse response coefficients are converted into transfer function parameters (4.4), which facilitates the transformation from a FIR representation of the LTI system to an IIR form, while also considerably reducing the number of parameters from M_h to $n_a + n_b + 2$.

4.2.2 Wiener Model

The Wiener model consists of a linear dynamic block followed by a nonlinear static block (see Figure 1.5). The input u is first processed by the LTI block, generating an unobservable intermediate signal v, which is then transformed by a static nonlinear function to generate the final output q.

Wiener systems are widely used in various applications. For instance, they effectively model a linear channel followed by a non-ideal sensor exhibiting static nonlinear effects. In the chemical process industry, Wiener systems have been applied for model-based control in several scenarios, including pH regulation in a continuous flow reactor (Pajunen, 1992), high-purity distillation column control (Zhu, 1999; Norquay et al., 1999a), and pH neutralisation experiments (Norquay et al., 1999b). Additionally, in microwave engineering, Wiener models have been used to describe wide-band power amplifiers (Clark et al., 1998). Several classical references discuss Wiener system identification, including Crama and Schoukens (2001); Hu and Chen (2005); Giri et al. (2009); Bai (2003); Vörös (2001); Greblicki (1992).

The noiseless output q_k of the Wiener model at time k is given by:

$$\begin{aligned} q_k &= f(v_k), \\ v_k &= S(z)u_k, \end{aligned} \tag{4.6}$$

where the static nonlinear function $f(\cdot)$ is approximated using polynomial basis function as shown in (4.3). The LTI subsystem is modelled using a rational transfer function of the backwards shift operator z^{-1}:

$$S(z) = \frac{d_0 + d_1 z^{-1} + \cdots + d_{n_d} z^{-n_d}}{c_0 + c_1 z^{-1} + \cdots + c_{n_c} z^{-n_c}}, \tag{4.7}$$

where d_i and c_i are coefficients grouped into the vectors $\boldsymbol{d} \in \mathbb{R}^{n_d+1}$ and $\boldsymbol{c} \in \mathbb{R}^{n_c+1}$, respectively. The first non-zero element of $\boldsymbol{c}$ is assumed to be positive, and its L^2-norm is normalised to 1. Alternatively, the LTI block can be represented using impulse response coefficients $\boldsymbol{s}$ with a memory length of M_w, as given by:

$$v_k = \sum_{i=0}^{M_\mathrm{w}} s_i u_{k-i}. \tag{4.8}$$

This impulse response formulation is used in this thesis to study the Wiener model, which can be converted to a transfer function model (4.7) with reduced parameters.

4.2.3 Wiener-Hammerstein Model

As the name suggests, the Wiener-Hammerstein (WH) model combines the Wiener and Hammerstein models. The static nonlinear block is sandwiched between two linear dynamic blocks (see Figure 1.5). The presence of two LTI blocks complicates the identification task. Some of the popular identification procedures are in Billings and Fakhouri (1978); Hunter and Korenberg (1986); Crama and Schoukens (2005); Lauwers et al. (2008); Schoukens et al. (2014a); Tan et al. (2012); Sjöberg and Schoukens (2012); Sjöberg et al. (2012); Westwick and Schoukens (2012); Schoukens et al. (2014b); Vanbeylen (2014); Tiels et al. (2014). WH models are used to model skeletal muscle dynamics (Bai et al., 2009).

The input u and the output y are the available signals for identification. The intermediate signals v and q are not directly measured. The noiseless output y_k of the WH model at time k is given by:

$$\begin{aligned} y_k &= T(z)q_k, \\ q_k &= f(v_k), \\ v_k &= S(z)u_k. \end{aligned} \tag{4.9}$$

The definitions and assumptions of the LTI subsystems $S(z), T(z)$ and the static nonlinear function $f(\cdot)$ given in Sections 4.2.1 and 4.2.2 also hold here.

4.2.4 Challenges in Identification

These models, though simple, interpretable, and parsimonious, also present several challenges in identification due to their structured yet nonlinear nature.

Configuration selection: One of the primary difficulties in identifying block-oriented models is selecting the appropriate configuration of linear dynamic and static nonlinear

blocks. The system's behaviour depends on the arrangement of these components and incorrect assumptions can lead to poor model accuracy or incorrect physical interpretations. Manually determining the best structure for a given system can be both time-consuming and computationally intensive. This work assumes that the structural configuration of the system is already known.

Hyperparameter selection: Once a structure is chosen, identifying the correct order of the linear block(s) and the appropriate complexity of the nonlinear functions is another challenge. If the model order is too low, the system dynamics may not be captured accurately, leading to biased estimates and poor generalisation. If the order is too high, the model becomes unnecessarily complex, increasing the risk of overfitting and making real-time implementation impractical. Finding an optimal trade-off between model complexity and accuracy requires systematic model selection techniques and data-driven optimisation methods. In this thesis, it is assumed that these hyperparameters are already available.

Identifiability issues: Block-oriented models suffer from identifiability issues, i.e. different parameter sets may produce similar input-output behaviour, making it difficult to uniquely determine model parameters.

For example, the linear and nonlinear blocks in Wiener and Hammerstein can only be determined up to an unknown gain α, i.e. scaling the nonlinear function $f(\cdot)$ by some factor α^{-1} and simultaneously scaling the linear system $T(z)$ (for Wiener model) or $S(z)$ (for Hammerstein model) by α does not change the overall system output. This inherent scaling ambiguity in these models results in multiple combinations of $\{T(z), f(\cdot)\}$ or $\{f(\cdot), S(z)\}$ representing the same system behaviour. Identifiability issues are further exacerbated when only limited input-output data is available, requiring careful excitation signal design to ensure all system dynamics are properly excited and distinguishable.

Similarly, in WH models, the presence of two linear dynamic blocks separated by a static nonlinearity allows gain exchange between different blocks. If the unknown gain factors α_1, α_2 are moved respectively from the first and second LTI blocks to the nonlinear block, then this results in arbitrary scaling ambiguity. Additionally, a phase shift (or time delay) observed in one of the LTI blocks can be exactly compensated by an inverse phase shift in the other LTI block, keeping the overall system response unchanged. Therefore, when identifying a WH model, the LTI blocks can be estimated with arbitrary scaling and shifts, making it difficult to determine the true system representation.

Optimisation: The estimation of block-oriented models often involves non-convex nonlinear optimisation. Many parameter estimation techniques rely on iterative algorithms, where convergence to local minima is common and heavily influenced by the initial parameter guesses and the choice of optimisation method. Moreover, Computational efficiency becomes a concern, particularly for large-scale systems or

real-time applications, where extensive optimisation routines may be impractical.

Noise: In real-world applications, external disturbances and measurement noise can further complicate the estimation, as they can obscure the true system behaviour, requiring robust filtering and preprocessing techniques to ensure reliable parameter estimation.

Input excitation: Poorly designed input signals may fail to sufficiently excite the relevant dynamics or nonlinear characteristics of the system, leading to biased or uninformative model estimates. This is particularly problematic in block-oriented structures, where the interplay between linear dynamic blocks and static nonlinearities can obscure certain internal behaviours unless the input spans a rich set of frequencies and amplitudes. Consequently, some identification methods impose restrictive assumptions on the excitation, such as the Gaussian nature of the signal, an independent and identically distributed (i.i.d.) signal, etc., to guarantee identifiability or to simplify the estimation process. However, such constraints may not be practical or realistic in real-world scenarios. In this work, the minimum requirement on the input signal is that it is sufficiently (persistently) exciting. This condition ensures that the input contains enough information to uncover the relevant system dynamics and nonlinearities.

4.3 Polynomial Nonlinear State-Space Model

The Polynomial Nonlinear State-Space (PNLSS) model provides a powerful framework for representing nonlinear dynamical systems by augmenting the classical linear state-space formulation with nonlinear functions of the states and inputs. This approach offers a flexible balance between interpretability and expressiveness, enabling the capture of both linear dynamics and complex nonlinear interactions within a unified structure. Moreover, its polynomial formulation allows for convenient analytical manipulation, efficient identification, and a clear connection with other nonlinear modelling approaches such as Volterra and block-oriented models.

The NLSS model introduced in Section 1.3.2 is extended here by introducing functional expansions of the functions z_1, z_2 in (1.15). The polynomial basis function is the popular choice and also appropriate for this work, since the nonlinearity in the system is assumed to be modelled with a polynomial function (4.3). Using the polynomial basis function (Paduart et al., 2010) in the NLSS model confers numerical advantages for optimisation and approximation when we aim to realise it through other polynomial models, like the Volterra model. Therefore, assuming the polynomial basis function and separating linear and nonlinear terms in z_1 and z_2, the discrete-time polynomial

NLSS (PNLSS) model (Paduart et al., 2010; Decuyper et al., 2021) is obtained:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}u_k + \boldsymbol{E}\boldsymbol{\zeta}(\boldsymbol{x}_k, u_k), \\ y_k &= \boldsymbol{c}^\top \boldsymbol{x}_k + du_k + \boldsymbol{f}^\top \boldsymbol{\eta}(\boldsymbol{x}_k, u_k), \end{aligned} \tag{4.10}$$

where $\boldsymbol{\zeta} \in \mathbb{R}^{n_\zeta}$ and $\boldsymbol{\eta} \in \mathbb{R}^{n_\eta}$ contain monomial basis functions of degrees 2 up to D in u_k and $\boldsymbol{x}_k$, where D is the maximum degree of nonlinearity. The matrix $\boldsymbol{E} \in \mathbb{R}^{M \times n_\zeta}$ and vector $\boldsymbol{f} \in \mathbb{R}^{n_\eta}$ contain the coefficients of monomials in $\boldsymbol{\zeta}$ and, $\boldsymbol{\eta}$ respectively. The parameters of the PNLSS model (4.10) are the elements of the parameters $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{E}$ and $\boldsymbol{f}$, denoted by $\boldsymbol{\theta}$ (assuming zero initial conditions $\boldsymbol{x}_0 = \boldsymbol{0}$ and $u_0 = 0$), so formally we have

$$\boldsymbol{\theta} = \begin{bmatrix} \mathrm{vec}(\boldsymbol{A})^\top & \boldsymbol{b}^\top & \boldsymbol{c}^\top & d & \mathrm{vec}(\boldsymbol{E})^\top & \boldsymbol{f}^\top \end{bmatrix}^\top. \tag{4.11}$$

The operator $\mathrm{vec}(\cdot)$ vectorises a matrix to a column vector.

In theory, the overall input-output behaviour of a PNLSS model does not depend on how its terms are categorised as linear or nonlinear. Whether a particular component is treated as part of the linear dynamics (e.g., parameters $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d$) or as part of the nonlinear mapping (e.g., parameters $\boldsymbol{E}, \boldsymbol{f}$), the total system behaviour remains the same, i.e., the output for a given input remains unchanged. This is due to the flexibility of the model, where the linear and nonlinear parts can compensate for each other.

In practice, however, separating linear and nonlinear parts is useful for the identification procedure. Traditionally, a linear approximation of the system is first estimated using techniques such as the BLA or N4SID as explained in Chapter 2. This initial linear model is used as a starting point for the second-stage nonlinear optimisation. Thus, while the separation does not affect the final model behaviour, it helps organise and simplify the identification process, facilitating a two-step identification approach.

In contrast, the approach proposed in this thesis does not rely on a two-step procedure. Instead, it uses the Volterra kernel representation to construct the PNLSS model directly. The parameters describing linear dynamics $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d$ are estimated from the system's impulse response, while the parameters of the nonlinear mapping $\boldsymbol{E}, \boldsymbol{f}$ are obtained from the Volterra kernel coefficients, including higher-order polynomial terms. This unified identification approach offers the following advantages:

- It does not require prior separation between linear and nonlinear behaviour.
- The model parameters are directly linked to the Volterra kernel, which has a clear physical interpretation.
- The method is less sensitive to model mismatch or poor initialisation, which can affect traditional two-step approaches.

The following concepts will be helpful to understand how the multivariable monomials are organised in the vectors $\boldsymbol{\zeta}(\boldsymbol{x}_k, u_k), \boldsymbol{\eta}(\boldsymbol{x}_k, u_k)$.

4.3.1 Multi-Index Framework for Multivariate Polynomial Expansion

The concept of a multi-index, an n-tuple of non-negative integers used to compactly express operations like $|\boldsymbol{\alpha}|$, $\boldsymbol{\alpha}!$, $\boldsymbol{x}^{\boldsymbol{\alpha}}$, and $\partial^{\boldsymbol{\alpha}}$, has long been established in multivariable calculus, partial differential equations theory, and functional analysis (Reed and Simon, 1972). In the context of PNLSS modelling, Paduart et al. (2010); Paduart (2008) adopted this notation to express multivariate polynomial expansions efficiently, along with the associated combinatorial structure for organising monomials. The representation is simplified for a SISO system in this thesis. A multi-index is defined as:

$$\boldsymbol{\alpha} = \Big[\underbrace{\alpha_1 \quad \alpha_2 \quad \dots \quad \alpha_M}_{\text{for } \boldsymbol{x}_k} \quad \underbrace{\alpha_{M+1}}_{\text{for } u_k}\Big], \tag{4.12}$$

where each α_i is a non-negative integer. The first M integers in $\boldsymbol{\alpha}$ represent the exponent of M variables in state vector $\boldsymbol{x}_k$, and the $(M+1)^{\text{th}}$ integer in $\boldsymbol{\alpha}$ represents the exponent of u_k in a multivariable monomial. Given a variable vector

$$\boldsymbol{\xi} = \begin{bmatrix}\xi_1 & \dots & \xi_M & \xi_{M+1}\end{bmatrix}^\top = \begin{bmatrix}\boldsymbol{x}_k(1) & \dots & \boldsymbol{x}_k(M) & u_k\end{bmatrix}^\top, \tag{4.13}$$

a monomial associated with the multi-index $\boldsymbol{\alpha}$ is written as:

$$\boldsymbol{\xi}^{\boldsymbol{\alpha}} = \prod_{i=1}^{M+1} \xi_i^{\alpha_i}. \tag{4.14}$$

The total degree of the monomial is then:

$$|\boldsymbol{\alpha}| = \sum_{i=1}^{M+1} \alpha_i, \tag{4.15}$$

and the multi-index factorial is defined as the product of the factorials of its components:

$$\boldsymbol{\alpha}! = \alpha_1!\alpha_2! \dots \alpha_{M+1}! \tag{4.16}$$

To enumerate monomials systematically, let $\boldsymbol{\xi}_{(D)}$ be a column vector of all distinct monomials of total degree D, formed from the elements of the vector $\boldsymbol{\xi}$. The number of such monomials is given by the binomial coefficient:

$$\binom{M+D}{D}. \tag{4.17}$$

If instead we consider all monomials of degree from 2 up to D (excluding degrees 0 and 1), the vector is still denoted using $\boldsymbol{\xi}_{\{2:D\}}$. The total number of such monomials is:

$$L_{M,D} = \binom{M+D+1}{D} - M - 2. \tag{4.18}$$

Using these definitions, the classical theorem is stated (Weisstein, 2005) below:

Theorem 1 (Multinomial Expansion Theorem). The multinomial expansion theorem provides an expression for the power of a sum in terms of monomials:

$$\left(\sum_{i=1}^{M+1} \xi_i\right)^k = \sum_{|\boldsymbol{\alpha}|=k} \frac{k!}{\boldsymbol{\alpha}!} \boldsymbol{\xi}^{\boldsymbol{\alpha}}. \tag{4.19}$$

4.3.2 Ordering Multivariate Monomials

To work with multivariate polynomials systematically, Paduart et al. (2010); Paduart (2008) uses the graded lexicographic (grlex) order, a consistent way to order the monomials. This ensures that monomials are arranged in a deterministic and logical way. The idea begins with choosing an order among the variables, such as:

$$\xi_1 \prec \xi_2 \prec \cdots \prec \xi_{M+1}, \tag{4.20}$$

where $\prec$ denotes the variable precedence relation used in the grlex order.

Using this ordering, any monomial (a product of powers of variables) is sorted by first grouping terms based on their total degree (i.e., the sum of powers), and then ordering within each group using the lexicographic rule. For instance, the unordered monomial $\xi_3^2\xi_2\xi_1$ should be written as $\xi_1\xi_2\xi_3^2$ under this rule. Monomials with the same total degree are compared like words in a dictionary:

$$\xi_1^2\xi_2 \prec \xi_1\xi_3^2 \prec \xi_2\xi_3^2. \tag{4.21}$$

Monomials with different total degrees are placed into separate groups, starting from the lower degrees to the higher ones. This grouping by degree, combined with lexicographic sorting, is what defines the grlex order. This ordering is essential when forming vectors that collect all monomials of a certain degree. The vector $\boldsymbol{\xi}_{\{2:D\}}$ collects all monomials of total degree 2 up to D. For instance, when $M = 1$ and $D = 3$,

$$\boldsymbol{\xi}_{\{2:3\}} = \begin{bmatrix} \boldsymbol{\xi}_{(2)} \\ \boldsymbol{\xi}_{(3)} \end{bmatrix} = \begin{bmatrix} \xi_1^2 & \xi_1\xi_2 & \xi_2^2 & \xi_1^3 & \xi_1^2\xi_2 & \xi_1\xi_2^2 & \xi_2^3 \end{bmatrix}^\top. \tag{4.22}$$

4.3.3 Polynomial Approximation and Vector Construction

To accurately represent a nonlinear system using polynomials, all relevant monomials up to a chosen degree D, i.e., every possible combination of the system's state and input variables and their products up to degree D, are included.

The vector defined in (4.13) simply stacks all the state variables $\boldsymbol{x}_k$ and input u_k at time k into a single column vector. To include the nonlinear effects, let us build a new vector

$\boldsymbol{\xi}_{\{2:D\}}(\boldsymbol{x}_k, u_k)$ that contains all monomials formed from the entries of $\boldsymbol{\xi}$ with degrees ranging from 2 up to D, arranged in grlex order:

$$\boldsymbol{\zeta}(\boldsymbol{x}_k, u_k) = \boldsymbol{\eta}(\boldsymbol{x}_k, u_k) = \boldsymbol{\xi}_{\{2:D\}}(\boldsymbol{x}_k, u_k). \tag{4.23}$$

This vector captures all nonlinear combinations used in the model.

The number of parameters in the PNLSS model depends on the number of such monomials. The total number of parameters of the PNLSS model is:

$$\left(\binom{M+D+1}{D} - 1\right)(M+1). \tag{4.24}$$

In practice, some parameters are not uniquely identifiable due to similarity transformations in the model. When all nonlinear combinations are included in $\boldsymbol{\zeta}(\boldsymbol{x}_k, u_k)$ and $\boldsymbol{\eta}(\boldsymbol{x}_k, u_k)$, the model remains invariant to such transformations, i.e., the number of effective parameters can be reduced by removing redundant ones:

$$\left(\binom{M+D+1}{D} - 1\right)(M+1) - M^2. \tag{4.25}$$

This compact representation helps manage the complexity of the model while still capturing rich nonlinear behaviour.

4.3.4 Stability Correction

The stability of a discrete-time PNLSS model (4.10) is determined by the state-update equation. In particular, the recursive nature of the model is governed by the evolution of the internal state. Hence, to analyse the stability of a general model of the form:

$$\boldsymbol{x}_{k+1} = z_1(\boldsymbol{x}_k, u_k), \tag{4.26}$$

it is sufficient to study how the state $\boldsymbol{x}_k$ evolves, given an initial state and an input signal. The output equation does not affect stability and is therefore not considered in this context.

A useful notion for analysing the stability of such systems is Input-to-State Stability (ISS) (Paduart, 2008). Informally, ISS means that the state of the system will remain bounded if the input remains bounded, and that the effect of the initial state fades away over time. If the input signal approaches zero, the state also tends to zero, regardless of how large the initial state was.

Let the system be given an initial condition and an input signal. If the input is zero, the state should eventually decay to zero (internal stability). If the input is small but not

zero, the state should remain small and not grow unbounded. Formally, a system is Input-to-State stable if the state magnitude at time k can be bounded by a function of the initial condition and a function of the input signal.

Example 10. Consider the following discrete-time nonlinear model with a single state and a single input:

$$x_{k+1} = 0.5x_k + u_k + 0.2x_k^2, \qquad y_k = x_k, \tag{4.27}$$

where x_k is the internal state and u_k is the external input. This is a simple instance of a PNLSS model, where the nonlinear term x_k^2 represents a second-order polynomial nonlinearity in the state.

Suppose the initial condition is $x_0 = 0$, and the input is constant, $u_k = 1$ for all $k \geq 0$. In this case, the state grows rapidly, and the output y_k eventually becomes unbounded (see Figure 4.2), indicating instability under this particular input and parameter setting.

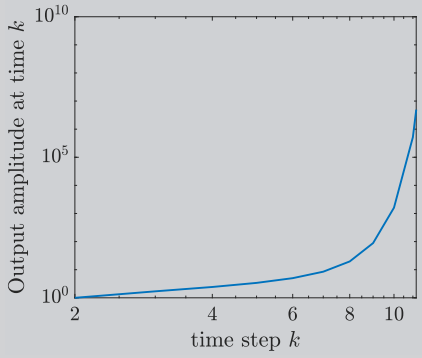
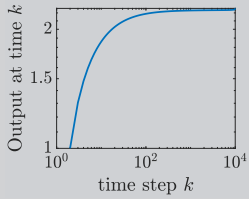
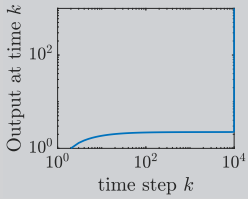


Figure 4.2: Unbounded state and output for a constant input to the system in (4.27).

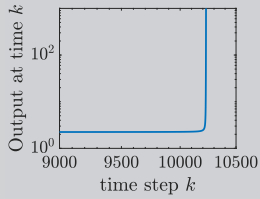
This does not imply that the model structure is inherently unstable. Rather, the system's stability is highly sensitive to the values of the linear parameters, particularly the state-transition coefficient $\boldsymbol{A}$. For example, reducing $\boldsymbol{A} = 0.5$ in (4.27) to a carefully chosen value such as $\boldsymbol{A} = 0.10557302$ can yield bounded state trajectories over a finite horizon (e.g., $N = 10{,}000$ samples), shown in Figure 4.3 (a). However, even a slight perturbation (e.g., to 0.10557303 in Figure 4.3 (b)), or increasing the simulation length (e.g., $N = 10{,}300$ in Figure 4.3 (c)), can result in divergence. This illustrates that the nonlinear system can exhibit metastable behaviour, i.e., appearing stable for a while before diverging due to the compounding effect of the nonlinear term. Therefore, this system is not Input-to-State stable, i.e., there exists a bounded input which leads to an unbounded state value.



(a) Stable output for $\boldsymbol{A} = 0.10557302$, $N = 10{,}000$



(b) Unstable output for $\boldsymbol{A} = 0.10557303$, $N = 10{,}000$



(c) Unstable output for $\boldsymbol{A} = 0.10557302$, $N = 10{,}300$

Figure 4.3: Illustration of how small variations in the state parameter $\boldsymbol{A}$ or the simulation length N affect output boundedness. The figure demonstrates that a system may appear stable for a given combination of $\boldsymbol{A}$ and N, but can become unstable under minute perturbations of $\boldsymbol{A}$ or slight increases in N, highlighting the sensitivity of stability to both parameters.

The nonlinear term $0.2x_k^2$ has negligible influence on the system's behaviour near the origin. This is because the Jacobian of the state-update function with respect to x_k is:

$$\left.\frac{\partial x_{k+1}}{\partial x_k}\right|_{x_k=0} = 0.5 + 0.4x_k\Big|_{x_k=0} = 0.5 < 1.$$

Since the derivative of the nonlinear term is zero at the origin, it does not contribute to the local linearised dynamics. Therefore, when analysing local stability (typically around an equilibrium point such as the origin under zero input), the system behaves like a linear system governed solely by $\boldsymbol{A}$.

However, as x_k grows large (i.e., the state moves away from the equilibrium region), the quadratic term becomes dominant. This introduces strong nonlinear effects that can eventually destabilise the system, even if the linear part appears stable. Thus, the nonlinear term affects the global behaviour significantly, though it may appear safe in short simulations or near equilibrium.

In the context of system identification, this underscores a key insight: the estimated model's stability is primarily governed by the linear part of the system. Nonlinear terms are essential for accuracy but typically have a limited effect on local stability unless the system operates far from equilibrium. Consequently, stability enforcement during PNLSS model estimation is often achieved by constraining $\boldsymbol{A}$ (e.g., ensuring its eigenvalues lie within the unit circle in discrete-time systems), while adapting $\boldsymbol{b}$, $\boldsymbol{c}$, and d to ensure a good fit to the observed data.

Stabilising a PNLSS model by modifying $\boldsymbol{A}$ alone may degrade the model's predictive performance. Therefore, practical identification methods must balance stability enforcement with parameter re-optimisation to maintain consistency with the system's input-output behaviour.

Example 11. Consider the following discrete-time state-space model:

$$\begin{aligned} x_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}u_k + \text{Nonlinear terms}, \\ y_k &= \boldsymbol{c}^\top \boldsymbol{x}_k + \text{Nonlinear terms}, \end{aligned} \tag{4.28}$$

where $\boldsymbol{x}_k$ is the internal state and u_k is the external input. The parameters are randomly generated as follows:

$$\boldsymbol{A} = \begin{bmatrix} 1.2 & 0.3 \\ 0.1 & 1.1 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \boldsymbol{c} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{4.29}$$

The eigenvalues of $\boldsymbol{A}$ are outside the unit circle, implying that the system is Input-to-state unstable. When simulating this model, the output diverges even for bounded input signals.

To enforce stability, several strategies have been proposed in the literature (Lacy and Bernstein, 2003; Qin, 2006; Di Natale et al., 2024; Umenberger and Manchester, 2016; Decuyper et al., 2021). Constrained optimisation methods impose stability directly on the system matrix (e.g., Schur or linear matrix inequality (LMI)-based constraints) during re-estimation (Lacy and Bernstein, 2003). Projection and subspace-based techniques enforce stability by re-parametrising the state matrix within the stable (Schur) manifold (Qin, 2006). Machine learning–based formulations, such as System Identification Methods leveraging Backpropagation (SIMBa) (Di Natale et al., 2024), embed an LMI-based free parametrisation of Schur matrices within a backpropagation framework to guarantee stability during training. Gradient-based or machine learning approaches achieve stability by embedding stable parametrisations or Lyapunov constraints in the identification loss (Umenberger and Manchester, 2016). In this work, however, the PNLSS model obtained from BLA-based initial estimates adopts the pole reflection strategy proposed by Decuyper et al. (2021), in which unstable poles are mirrored inside the unit circle and the frequency response is recomputed to yield a stable realisation.

Decuyper et al. (2021) reflects the unstable poles to the unit circle in the discrete-time case by following the steps below:

- Diagonalise the state matrix $\boldsymbol{A}$ and separate the stable and unstable parts.

- Reflect the unstable poles, i.e., map each eigenvalue λ with $|\lambda| > 1$ to $\frac{1}{\lambda}$.
- Construct FRFs of the original stable part and the reflected unstable part.
- Merge them and fit a new state-space model to the combined FRF.

The stabilised model is not mathematically identical to the original unstable model, but it is intended to be close in terms of input-output behaviour, particularly in the amplitude characteristics of the frequency response. This is because the pole-reflection procedure preserves the magnitude response across frequencies. However, it alters the phase, since no all-pass compensation is applied. Reflecting unstable poles across the unit circle maintains amplitude but introduces phase distortion due to the change in pole locations. While an all-pass filter could, in principle, restore the original phase, such compensation is typically omitted to keep the stabilisation procedure simple.

After stabilisation, the system is re-identified by fitting a new model to the modified frequency response. This re-identification step is approximate and subject to numerical error, noise, and non-uniqueness in model realisation. As a result, the poles of the stabilised state matrix $\boldsymbol{A}_{\text{stable}}$ may still lie outside the unit circle, particularly if the reflection procedure or frequency resolution is not sufficiently accurate. In practice, this issue is often mitigated by applying pole projection or regularisation techniques (enforcing a spectral radius constraint during optimisation), which typically yield a stable model even if stability is not analytically guaranteed.

The Example 10 and Example 11 confirm that the stability of the system primarily depends on the linear part, i.e., the matrix $\boldsymbol{A}$. This is because the nonlinear state contribution $\boldsymbol{E}\boldsymbol{\zeta}(\boldsymbol{x}_k, u_k)$ is commonly treated as a bounded nonlinear perturbation, for small input and state. The nonlinear term is initially zero or negligible and grows slowly with $\boldsymbol{x}_k$ and u_k. Hence, it does not affect the linear stability analysis directly.

However, when the input or state amplitude becomes large, the nonlinear term may no longer be negligible, and its influence on the system's stability becomes significant. In such cases, local stability is no longer sufficient, and one must consider methods that address the nonlinear part explicitly. One approach is to regularise or constrain the nonlinear coefficient matrix $\boldsymbol{E}$, for example, by scaling or penalising its norm during model estimation. Although this idea is not extensively studied in the PNLSS model, recent advances in machine learning and nonlinear system identification offer valuable insights. In particular, contraction-based methods enforce global stability by bounding the Lipschitz gain of the nonlinear dynamics (Manchester et al., 2021) for a comprehensive tutorial on using contraction constraints to stabilise black-box models. Additionally, kernel-based regularisation framework can implicitly stabilise nonlinear terms by penalising large dynamic coefficients (Pillonetto et al., 2022).

4.3.5 Nonlinear Optimisation of the Linear Model

The initial LSS estimates $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d)$ obtained through linear identification techniques such as BLA (Section 2.3.6), MOESP (Section 2.4.2) or N4SID (Section 2.4.3) serve as a starting point for PNLSS model construction. In this Section, these initial estimates are refined through nonlinear optimisation. The nonlinear mappings $(\boldsymbol{E}, \boldsymbol{f})$ of the PNLSS model are typically initialised to zero. Subsequently, all parameters, both linear and nonlinear, are jointly optimised to better capture the system's nonlinear behaviour. This procedure, developed in Paduart (2008); Paduart et al. (2010); Pintelon and Schoukens (2012); Decuyper et al. (2021), ensures a more accurate and complete representation of the system dynamics within the PNLSS framework. The implementation in the PNLSS toolbox (Decuyper et al., 2021) applies this procedure to initial estimates obtained via BLA (see Algorithm 1). The same process applies to estimates obtained via MOESP or N4SID.

The weighted least-squares cost function V_{ss}, defined in (2.54), provides a measure of model quality. In many cases, the initial linear estimates already yield low values of V_{ss}, indicating a reasonable model. However, in practice, the value of V_{ss} is highly sensitive to the dimension parameter r, which is chosen to be greater than the model order M in the first step of Algorithm 1. A straightforward strategy to select this hyperparameter is to perform linear identification for several values of r, and choose the model that minimises V_{ss}.

A more robust approach, however, is to directly minimise the cost function V_{ss} with respect to the parameters $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d)$. This results in a nonlinear least-squares problem that is solved using the Levenberg–Marquardt (LM) algorithm. The LM method requires the Jacobian of the model error $\epsilon(l)$ (2.55) with respect to the parameters. Using the expression for $\epsilon(l)$ (2.55) and the frequency-domain state-space transfer function in (2.56), the partial derivatives are given by:

$$\begin{aligned}
\frac{\partial \epsilon(l)}{\partial A_{ij}} &= \boldsymbol{c}^\top (z_l \boldsymbol{I}_M - \boldsymbol{A})^{-1} E_{ij}^{M\times M} (z_l \boldsymbol{I}_M - \boldsymbol{A})^{-1} \boldsymbol{b}, \\
\frac{\partial \epsilon(l)}{\partial b_i} &= \boldsymbol{c}^\top (z_l \boldsymbol{I}_M - \boldsymbol{A})^{-1} \boldsymbol{e}_i, \\
\frac{\partial \epsilon(l)}{\partial c_j} &= \boldsymbol{e}_j^T (z_l \boldsymbol{I}_M - \boldsymbol{A})^{-1} \boldsymbol{b}, \\
\frac{\partial \epsilon(l)}{\partial d} &= 1,
\end{aligned} \tag{4.30}$$

where $E_{ij}^{M\times M}$ denotes the standard basis matrix with a 1 at position (i, j), and $\boldsymbol{e}_i$ is the i-th canonical unit vector.

For the full parametrisation of the PNLSS model (4.10), the input is assumed to be noiseless. Identification is performed in the frequency domain, as this allows noise characteristics to be incorporated non-parametrically into the cost function. Consider the weighted LS cost function

$$V_{\mathrm{WLS}}(\boldsymbol{\theta}) = \sum_{l=1}^{F} w(l)\epsilon^*(l,\boldsymbol{\theta})\epsilon(l,\boldsymbol{\theta}), \tag{4.31}$$

where $\boldsymbol{\theta}$ denotes the vector of all model parameters, and $w(l)$ is a user-defined weighting function. The model error $\epsilon(l,\boldsymbol{\theta})$ is defined as:

$$\epsilon(l,\boldsymbol{\theta}) = Y(l,\boldsymbol{\theta}) - Y(l), \tag{4.32}$$

where $Y(l,\boldsymbol{\theta})$ and $Y(l)$ are the Discrete Fourier Transforms (DFT) of the modelled and measured outputs, respectively.

The cost function $V_{\mathrm{WLS}}(\boldsymbol{\theta})$ is minimised with respect to $\boldsymbol{\theta}$ using the LM algorithm, which requires the computation of the Jacobian of the model error with respect to $\boldsymbol{\theta}$. These Jacobians are computed in the time domain, followed by a DFT to obtain their frequency-domain representation. For the PNLSS model (4.10), the derivation of the Jacobians is detailed in Paduart (2008).

When computing the state sequence $\boldsymbol{x}_k$, the initial state $\boldsymbol{x}_0$ must be taken into account. A straightforward strategy is to compute the Jacobian over the entire dataset and discard the initial transient samples of both the model error and the Jacobian. However, this means part of the data is not used for parameter estimation. If the excitation is periodic, a more effective approach is to compute the Jacobians over several periods and select one in which the transients have sufficiently decayed. For both periodic and non-periodic inputs, it is recommended to treat the initial state and input as additional model parameters to be estimated.

A well-known issue in state-space identification is the non-uniqueness of the model due to similarity transformations. Any non-singular transformation of the state-space matrices that preserves input-output behaviour introduces at least M^2 redundant degrees of freedom in the parameter space (where M is the state dimension). This leads to rank deficiency in the Jacobian matrix and results in an infinite number of equivalent solutions. To address this, the parameter update step in LM optimisation uses a pseudoinverse based on truncated SVD (Pintelon and Schoukens, 2012), which enables full parametrisation while avoiding issues due to rank deficiency.

The nonlinear optimisation is continued until the cost function V_{WLS} no longer decreases. However, as is common in models with many parameters, overfitting can occur. To prevent this, a stopped search strategy is employed (Sjöberg and Schoukens, 2012): at each iteration, the quality of the estimated model is evaluated on a separate validation set, and the model that achieves the best performance on this set is selected. This

approach acts as a form of implicit regularisation, as it avoids overfitting not by explicitly removing parameters, but by selecting a model that generalises best.

4.4 Volterra Series

The Volterra series is a powerful analytical framework in nonlinear system identification. It is a generalisation of the Taylor series for systems with memory, allowing us to describe a wide class of causal, TI and nonlinear systems through a structured expansion of input-output relationships. This work focuses on discrete-time, SISO systems. The output signal sequence $\boldsymbol{y}$ of a nonlinear system excited by the input sequence $\boldsymbol{u}$ can be expressed as an infinite sum of discrete-time Volterra functions:

$$\boldsymbol{y} = h_0 + \sum_{d=1}^{\infty} H_d(\boldsymbol{u}), \tag{4.33}$$

where each $H_d(\boldsymbol{u})$ is the dth order Volterra operator, capturing the contribution of d-fold nonlinear interactions of past inputs to the current output. The constant term h_0 corresponds to the zeroth-order Volterra kernel, which is independent of the input signal. It captures any static bias present in the output, regardless of the input, which is required when identifying the DC component of a nonlinear system. In this work, it is assumed that there is no static bias. Hence, the constant term in the static nonlinear model q_k in (4.3) is not considered, which in turn also makes $h_0 = 0$.

In discrete-time, the operators $H_d(\boldsymbol{u})$ are defined as multidimensional convolutions:

$$H_d(\boldsymbol{u}) = \sum_{m_1=0}^{\infty} \cdots \sum_{m_d=0}^{\infty} h_d(m_1, \cdots, m_d) \prod_{j=1}^{d} u_{k-m_j}, \tag{4.34}$$

where $h_d(m_1, \cdots, m_d)$ being the dth order Volterra kernel. These kernels are analogous to impulse responses in linear systems ($d = 1$ case), but allow for modelling higher-order memory and nonlinear interactions. For practical applications, the series is truncated to a finite memory length M and maximum order D, resulting in a finite-dimensional model (1.16) that can still capture rich nonlinear dynamics.

Despite its flexibility and linear-in-the-parameter characteristics, the Volterra model comes with certain challenges:

- **Parameter count:** The number of coefficients grows combinatorial with the model order D and memory length M.
- **Convergence and numerical stability:** Like power series, Volterra series may diverge or fail to accurately represent discontinuous nonlinearities, such as dead zones, saturation, etc.

- **Overfitting and interpretability:** Without proper structural knowledge of the system, estimating higher-order kernels can lead to poor generalisation and interpretation of the underlying physical phenomena.

A widely adopted solution to these issues is regularisation, which introduces prior knowledge to control model complexity. In particular, Birpoutsoukis et al. (2017, 2018) propose a non-parametric Volterra kernel estimation framework that incorporates smoothness and exponential decay priors through Kronecker-structured regularisation matrices. These regularisers penalise large or oscillatory kernel coefficients, thereby promoting fading memory, numerical robustness, and improved generalisation. This approach allows higher-order Volterra kernels to be estimated more reliably even in short-data or noisy regimes, and has since been extended and applied in various block-oriented model identification tasks (Birpoutsoukis et al., 2017, 2018).

Despite these challenges, the Volterra series remains highly valuable for system identification, especially when nonlinear behaviour is smooth, analytic, and memory-dependent. When combined with appropriate regularisation techniques, Volterra models can effectively capture complex dynamics without overfitting. In fact, under the assumption of fading memory (see Definition 4.4.1), as introduced in Boyd and Chua (1985), finite-order Volterra models are guaranteed to approximate a broad class of stable nonlinear systems uniformly. Fading memory implies that recent input values dominate the output behaviour, while distant past inputs have diminishing influence.

Definition 4.4.1 (Fading memory)**.** A system has *fading memory* if the recent inputs affect the output more than the distant past inputs. Formally, let f be an operator mapping bounded input sequences to outputs, and let $\boldsymbol{u}, \boldsymbol{v}$ be two input signals defined on $\mathbb{Z}_{\leq 0}$ and belonging to a compact subset $\mathcal{K}$ of the input space. Then f has fading memory if for every $\epsilon > 0$, there exists a $\delta > 0$, such that

$$\sup_{k \leq 0} w_{-k} |u_k - v_k| < \delta \implies |f(\boldsymbol{u}) - f(\boldsymbol{v})| < \epsilon, \tag{4.35}$$

where w_{-k} is a non-negative weight function decreasing with k. A small difference in recent input values (with appropriate weighting) results in a small difference in the output. This is crucial to ensure convergence of the Volterra representation over infinite time horizons.

In this thesis, the Volterra series serves as both a modelling tool and a foundational bridge to other structured nonlinear models such as Hammerstein, Wiener, and WH systems. Furthermore, we exploit its linear-in-the-parameters nature to construct initial estimates for more compact and interpretable nonlinear representations, including

the PNLSS model. This thesis will continue to focus on discrete-time SISO systems, excluding continuous-time and MIMO extensions. The techniques developed and the algorithms presented in the following chapters are tailored specifically for the identification and analysis of these systems.

4.5 Identification of Block-oriented Models

Block-oriented models such as Hammerstein, Wiener and WH structures play a central role in nonlinear system identification due to their modularity and interoperability. This section reviews selected literature that addresses the identification of these models, with a particular focus on the WH system. Since the Wiener and Hammerstein models are special cases of the WH structure, studying the WH system provides insights and identification strategies that are directly applicable to those simpler configurations as well. The discussion centres on five key works (Sections 4.5.1-4.5.5) that are also closely related to the methodology proposed in this thesis. The works on the WH model decomposition, pole allocation via the BLA, and structural equivalence with the PNLSS model are examined. Additionally, recent efforts leveraging Volterra series and tensor-based representations for identifying the WH system are reviewed. Beyond these foundational studies, several recent contributions, given in Section 4.5.6, further enrich the landscape of block-oriented model identification.

4.5.1 Decoupling the WH System into Subsystems

Lauwers et al. (2008) proposes an initialisation method for the WH system based on the BLA. The core idea is to split the system into two subsystems by expressing the static nonlinearity as a concatenation of two simpler static nonlinear functions. This allows the system to be decomposed into two subsystems, each consisting of an LTI block and a static nonlinearity. The LTI blocks are then expressed as a linear combination of basis functions constructed from the poles and zeros of G_{BLA} (2.13).

To obtain these poles and zeros, the BLA is first estimated nonparametrically and then fitted with a parametric model. Assuming no pole-zero cancellations and simple poles and zeros, the resulting transfer function is decomposed into partial fractions, which are used to build basis functions for each subsystem.

Then, a general model structure is proposed that is linear-in-the-parameters, which enables estimation of the static nonlinearities using multivariable polynomials. A Total Least Squares (TLS) approach is used to estimate the unknown parameters, after which the initial estimates of the LTI blocks and the static nonlinearity are reconstructed from the identified parameters.

The proposed initialisation method is applied to a noise-free simulation dataset of the WH system in Lauwers et al. (2008). The simulation results show that the pass band of both filters and the transmission zero of the second filter are accurately retrieved. Thus, the initial estimates of both filters provide a good approximation of the dynamics of the linear blocks in the WH system. The nonlinear function is also well approximated in this procedure. The relative root mean square error (RRMSE) between the true output and the predicted output reported in Lauwers et al. (2008) is 0.125, which is approximately 65% lower than that of the BLA (RRMSE 0.36).

Despite the efficiency of the identification procedure, it has several drawbacks. It relies on assumptions such as the absence of pole-zero cancellation and the presence of simple poles and zeros. These assumptions may not always hold in practical systems, where exact or approximate cancellation might occur. To ensure that the subsystems remain consistent, i.e., the outputs of the two subsystems are equal, the authors introduce two static nonlinearities and a general model structure with multiple regressors. This increases the complexity of the regression problem and may cause overfitting or numerical instability in some cases. Additionally, the TLS step can be sensitive to noise and poorly conditioned regressors, an issue that could be further investigated by conducting experiments under output noise conditions.

4.5.2 Pole Allocation Based on the BLA

A key challenge in block-oriented system identification, especially for WH models, is generating reliable initial estimates for the dynamics of the individual linear blocks, as discussed in Section 4.2.4. Traditional methods rely on the BLA of the nonlinear system, followed by brute-force scanning over all possible pole-zero allocations between the two blocks (Sjöberg and Schoukens, 2012). While it is relatively straightforward to estimate the product of the two linear dynamic blocks by measuring the BLA, initialising a parametric identification algorithm for the full WH model requires splitting this product into its components. This separation is nontrivial and lies at the heart of the WH identification challenge.

The approach of Sjöberg and Schoukens (2012) proceeds in several steps:

- Estimating the overall linear dynamics from data,
- Extracting the poles and zeros from this estimate,
- Enumerating all possible partitions of these poles and zeros between the two linear subsystems,
- Estimating the static nonlinearity and evaluating the resulting full model on the data for each candidate split.

While this method provides a systematic way to generate initial guesses, it suffers from a combinatorial explosion, i.e., the number of pole-zero allocations grows rapidly with model order, making the procedure computationally expensive and impractical for higher-order systems.

Alternative structured approaches based on higher-order BLAs (e.g., using squared or cubed input signals) can reveal pole shifts (e.g., from p_i to $2p_i$ or $3p_i$) that help identify which poles belong to which block (Westwick and Schoukens, 2012). However, these methods required very long experiments to achieve adequate precision.

To address the above limitations, Schoukens et al. (2014a) developed a refined pole-shifting strategy that significantly reduces the number of candidate models by classifying poles and zeros immediately after measuring the BLA, without needing to scan through all possible configurations. This is achieved using a specially designed class of excitation signals called phase-coupled multisines, which embed useful structural information into the response, leveraging only the first-order BLA.

The key idea is to design a multisine input signal in which the phases of certain frequency components are coupled in a specific pattern. Instead of assigning random phases independently to each excited frequency, the signal is constructed such that frequencies appear in pairs (or groups) with predetermined phase relationships. These phase couplings cause specific nonlinear cross-terms to manifest as frequency-shifted contributions in the system's BLA. These shifted contributions introduce asymmetric patterns in the estimated frequency response, breaking the usual symmetry expected from real-valued systems.

This asymmetry is particularly valuable: while the dynamics of one linear block (e.g., the output filter) appear unchanged in the shifted BLA, the poles or zeros of the other block (e.g., the input filter) are effectively shifted. By analysing these shifts, it becomes possible to attribute each pole or zero to the correct linear subsystem, a task that cannot be achieved with standard BLA measurement alone.

The method was validated on a simulated WH system composed of two second-order Chebyshev filters with different bandwidths and ripple levels, separated by a static nonlinearity. A phase-coupled multisine with controlled frequency resolution and shift was used to excite the system, and the resulting BLA was estimated from multiple realisations. The shifted frequency responses revealed clear signatures of the underlying system structure. Using parametric modelling and comparison of shifted versus non-shifted poles in the complex plane, the authors successfully identified which poles belonged to which linear subsystem.

While promising, this approach comes with limitations. First, it depends critically on the use of a specially designed excitation signal that may not be feasible in practical applications where the input cannot be freely chosen. Second, the method's effectiveness in the presence of measurement noise has not been thoroughly assessed,

as the simulations were conducted in a noise-free setting. Thus, although the technique shows clear advantages over brute-force and higher-order methods in simulation, its robustness and applicability to real-world systems remain open questions.

4.5.3 Representing Block-oriented Models Using the PNLSS model

This section discusses the structural relationship between the PNLSS model and standard block-oriented models: Hammerstein, Wiener, and WH models. The equivalence between these structures has been previously established in Paduart et al. (2007); Paduart (2008); Paduart et al. (2010), and forms an important foundation for the methodology developed in this thesis.

4.5.3.1 Hammerstein Structure

The Hammerstein model, introduced in Section 4.2.1, is composed of a static nonlinear block followed by a linear dynamic system. The structural equivalence between this model (4.5) and a PNLSS representation (4.10) has been well established in prior work (Paduart et al., 2007; Paduart, 2008).

At each time step k, the static nonlinear block transforms the input v_k into an intermediate signal q_k. By the Weierstrass approximation theorem, any continuous function can be uniformly approximated by a polynomial over a compact interval (Kreider et al., 1966). Hence, the static nonlinearity is modelled as follows:

$$q_k = \sum_{i=1}^{D} p_i v_k^i, \tag{4.36}$$

where D is the degree of the polynomial nonlinearity and p_i are the corresponding coefficients. For simplicity, the constant term (zeroth-order) is omitted in this formulation.

This intermediate signal q_k then passes through a linear dynamic system of order M, defined by the following state-space equations:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \tilde{\boldsymbol{A}}\boldsymbol{x}_k + \tilde{\boldsymbol{b}}q_k, \\ y_k &= \tilde{\boldsymbol{c}}^\top \boldsymbol{x}_k + \tilde{d}q_k, \end{aligned} \tag{4.37}$$

where $\{\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{b}}, \tilde{\boldsymbol{c}}, \tilde{d}\}$ are the parameters of linear system. Notably, the Hammerstein model exhibits nonlinearity only in the input, and this input nonlinearity influences

both the states and the output. Now, replacing the intermediate signal q_k in (4.37) with the definition of the nonlinear static block in (4.36) yields:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \tilde{\boldsymbol{A}}\boldsymbol{x}_k + \tilde{\boldsymbol{b}}\sum_{i=1}^{D} p_i v_k^i = \tilde{\boldsymbol{A}}\boldsymbol{x}_k + \tilde{\boldsymbol{b}}p_1 v_k + \tilde{\boldsymbol{b}}\begin{bmatrix} p_2 \\ \vdots \\ p_D \end{bmatrix}^\top \begin{bmatrix} v_k^2 \\ \vdots \\ v_k^D \end{bmatrix} \\ &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{b}v_k + \boldsymbol{E}\boldsymbol{\zeta}(v_k), \\ y_k &= \tilde{\boldsymbol{c}}^\top \boldsymbol{x}_k + \tilde{d}\sum_{i=1}^{D} p_i v_k^i = \tilde{\boldsymbol{c}}^\top \boldsymbol{x}_k + \tilde{d}p_1 v_k + \tilde{d}\begin{bmatrix} p_2 \\ \vdots \\ p_D \end{bmatrix}^\top \begin{bmatrix} v_k^2 \\ \vdots \\ v_k^D \end{bmatrix} \\ &= \boldsymbol{c}^\top \boldsymbol{x}_k + dv_k + \boldsymbol{f}^\top \boldsymbol{\eta}(v_k), \end{aligned} \tag{4.38}$$

where $\boldsymbol{\zeta}(v_k)$ and $\boldsymbol{\eta}(v_k)$ are the vectors of monomials from degree 2 to D, i.e., $\boldsymbol{\xi}_{\{2:D\}}(v_k)$, as defined in (4.23).

Thus, the equivalent PNLSS model parameters corresponding to the Hammerstein structure are:

$$\begin{aligned} &\boldsymbol{A} = \tilde{\boldsymbol{A}}, \quad \boldsymbol{b} = p_1\tilde{\boldsymbol{b}}, \quad \boldsymbol{c} = \tilde{\boldsymbol{c}}, \quad d = p_1\tilde{d}, \\ &\boldsymbol{E} = \begin{bmatrix} p_2 & \dots & p_D \end{bmatrix}\tilde{\boldsymbol{b}}, \quad \boldsymbol{f} = \tilde{d}\begin{bmatrix} p_2 & \dots & p_D \end{bmatrix}. \end{aligned} \tag{4.39}$$

4.5.3.2 Wiener Structure

The Wiener system consists of a linear dynamic block followed by a static nonlinear function, as shown in Figure 1.5. The relationship between the Wiener model (4.6) and the PNLSS model (4.10) is well established (Paduart et al., 2007; Paduart, 2008).

The linear dynamic part is defined as:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \tilde{\boldsymbol{A}}\boldsymbol{x}_k + \tilde{\boldsymbol{b}}u_k, \\ v_k &= \tilde{\boldsymbol{c}}^\top \boldsymbol{x}_k + \tilde{d}u_k, \end{aligned} \tag{4.40}$$

where $\{\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{b}}, \tilde{\boldsymbol{c}}, \tilde{d}\}$ are the parameters of the linear state-space model, and v_k is the internal (unmeasured) variable.

The output is obtained by passing v_k through a static nonlinearity:

$$y_k = \sum_{i=1}^{D} p_i v_k^i, \tag{4.41}$$

where D is the nonlinearity degree and p_i's are polynomial coefficients (excluding the constant term for simplicity). Substituting v_k from (4.40) into (4.41) yields:

$$y_k = \sum_{i=1}^{D} p_i (\tilde{\boldsymbol{c}}^\top \boldsymbol{x}_k + \tilde{d} u_k)^i. \tag{4.42}$$

This expression maps directly to the output nonlinearity of the PNLSS model. Therefore, the Wiener model can be embedded in a PNLSS framework with the following parameter mapping:

$$\begin{aligned} &\boldsymbol{A} = \tilde{\boldsymbol{A}}, \quad \boldsymbol{b} = \tilde{\boldsymbol{b}}, \quad \boldsymbol{c} = p_1 \tilde{\boldsymbol{c}}, \quad d = p_1 \tilde{d}, \quad \boldsymbol{E} = \emptyset, \\ &\boldsymbol{f} = \begin{bmatrix} p_2 \tilde{c}_1^2 & 2 p_2 \tilde{c}_1 \tilde{c}_2 & \dots & D p_D \tilde{c}_M \tilde{d}^{D-1} & p_D \tilde{d}^D \end{bmatrix}, \end{aligned} \tag{4.43}$$

with the nonlinear monomial vectors:

$$\boldsymbol{\zeta}(\boldsymbol{x}_k, u_k) = \emptyset, \quad \boldsymbol{\eta}(\boldsymbol{x}_k, u_k) = \boldsymbol{\xi}(\boldsymbol{x}_k, u_k)_{\{2:D\}}, \tag{4.44}$$

where $\boldsymbol{\xi}$ contains all multivariate monomials of degree 2 to D in lexicographic order (see Section 4.3.2).

4.5.3.3 Wiener-Hammerstein Structure

The relationship between the WH model (4.9) and the PNLSS model (4.10) is well established (Paduart et al., 2007; Paduart, 2008). In the WH model, a static nonlinear block is sandwiched between two linear dynamic systems. The first linear block (of order M_w) transforms the input signal u_k into an intermediate signal v_k, and the static nonlinear block produces a nonlinear output q_k, which is then passed through the second linear block (of order M_h) to generate the system output y_k.

First Linear Block (Front Dynamics)

$$\begin{aligned} \boldsymbol{x}_{k+1}^w &= \boldsymbol{A}_1 \boldsymbol{x}_k^w + \boldsymbol{b}_1 u_k, \\ v_k &= \boldsymbol{c}_1^\top \boldsymbol{x}_k^w + d_1 u_k, \end{aligned} \tag{4.45}$$

where $\boldsymbol{x}_k^w \in \mathbb{R}^{M_\mathrm{w}}$ is the state vector, and $(\boldsymbol{A}_1, \boldsymbol{b}_1, \boldsymbol{c}_1, d_1)$ are the state-space parameters of the first linear block.

Static Nonlinearity

$$q_k = \sum_{i=1}^{D} p_i v_k^i, \tag{4.46}$$

where D is the maximum degree of nonlinearity and p_i's are the polynomial coefficients.

Second Linear Block (Rear Dynamics)

$$\begin{aligned} \boldsymbol{x}_{k+1}^h &= \boldsymbol{A}_2 \boldsymbol{x}_k^h + \boldsymbol{b}_2 q_k, \\ y_k &= \boldsymbol{c}_2^\top \boldsymbol{x}_k^h + d_2 q_k, \end{aligned} \tag{4.47}$$

where $\boldsymbol{x}_k^h \in \mathbb{R}^{M_\mathrm{h}}$ is the state vector, and $(\boldsymbol{A}_2, \boldsymbol{b}_2, \boldsymbol{c}_2, d_2)$ are the parameters of the second linear block.

Expanding the nonlinear dependency of the second block's state update:

$$\begin{aligned} \boldsymbol{x}_{k+1}^h &= \boldsymbol{A}_2 \boldsymbol{x}_k^h + \boldsymbol{b}_2 \sum_{i=1}^{D} p_i v_k^i \\ &= \boldsymbol{A}_2 \boldsymbol{x}_k^h + \boldsymbol{b}_2 \sum_{i=1}^{D} p_i \{\boldsymbol{c}_1^\top \boldsymbol{x}_k^w + d_1 u_k\}^i \\ &= \boldsymbol{A}_2 \boldsymbol{x}_k^h + p_1 \boldsymbol{b}_2 \boldsymbol{c}_1^\top \boldsymbol{x}_k^w + p_1 d_1 \boldsymbol{b}_2 u_n \ldots \\ &+ \boldsymbol{b}_2 \Big[p_2 (c_{11} x_{k1}^w)^2 + 2 p_2 c_{11} c_{12} x_{k1}^w x_{k2}^w + \cdots + D p_D c_{1M_\mathrm{w}} d_1^{D-1} x_{kM_\mathrm{w}}^w {u_k}^{D-1} \ldots \\ &+ p_D (d_1 u_k)^D \Big]. \end{aligned} \tag{4.48}$$

Combining the two state equations into a single augmented state vector:

$$\begin{aligned} \begin{bmatrix} \boldsymbol{x}_{k+1}^w \\ \boldsymbol{x}_{k+1}^h \end{bmatrix} &= \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0} \\ p_1 \boldsymbol{b}_2 \boldsymbol{c}_1^\top & \boldsymbol{A}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^w \\ \boldsymbol{x}_k^h \end{bmatrix} + \begin{bmatrix} \boldsymbol{b}_1 \\ p_1 d_1 \boldsymbol{b}_2 \end{bmatrix} u_k \ldots \\ &+ \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{0} & \boldsymbol{0} \\ p_2 c_{11}^2 \boldsymbol{b}_2 & 2 p_2 c_{11} c_{12} \boldsymbol{b}_2 & \ldots & D p_D c_{1M_\mathrm{w}} d_1^{D-1} \boldsymbol{b}_2 & p_D d_1^D \boldsymbol{b}_2 \end{bmatrix} \begin{bmatrix} (x_{k1}^w)^2 \\ x_{k1}^w x_{k2}^w \\ \vdots \\ x_{kM_\mathrm{w}}^w u_k^{D-1} \\ u_k^D \end{bmatrix}. \end{aligned} \tag{4.49}$$

Therefore, the state parameters of the WH system are:

$$\begin{aligned}\boldsymbol{A} &= \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{0} \\ p_1\boldsymbol{b}_2\boldsymbol{c}_1^\top & \boldsymbol{A}_2 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_1 \\ p_1 d_1 \boldsymbol{b}_2 \end{bmatrix}, \\ \boldsymbol{E} &= \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} & \boldsymbol{0} \\ p_2 c_{11}^2 \boldsymbol{b}_2 & 2p_2 c_{11} c_{12} \boldsymbol{b}_2 & \dots & D p_D c_{1M_\mathrm{w}} d_1^{D-1} \boldsymbol{b}_2 & p_D d_1^D \boldsymbol{b}_2 \end{bmatrix}.\end{aligned} \tag{4.50}$$

Similarly, expanding the output equation:

$$\begin{aligned} y_k &= \boldsymbol{c}_2^\top \boldsymbol{x}_k^h + d_2 \sum_{i=1}^{D} p_i v_k^i \\ &= \boldsymbol{c}_2^\top \boldsymbol{x}_k^h + d_2 \sum_{i=1}^{D} p_i \{\boldsymbol{c}_1^\top \boldsymbol{x}_k^w + d_1 u_k\}^i \\ &= \begin{bmatrix} p_1 d_2 \boldsymbol{c}_1^\top & \boldsymbol{c}_2^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^w \\ \boldsymbol{x}_k^h \end{bmatrix} + p_1 d_1 d_2 u_k \dots \\ &\quad + d_2 \begin{bmatrix} p_2 c_{11}^2 & 2p_2 c_{11} c_{12} & \dots & D p_D c_{1M_\mathrm{w}} d_1^{D-1} & p_D d_1^D \end{bmatrix} \begin{bmatrix} (x_{k1}^w)^2 \\ x_{k1}^w x_{k2}^w \\ \vdots \\ x_{kM_\mathrm{w}}^w u_k^{D-1} \\ u_k^D \end{bmatrix}. \end{aligned} \tag{4.51}$$

The output parameters of the WH system are:

$$\begin{aligned} \boldsymbol{c}^\top &= \begin{bmatrix} p_1 d_2 \boldsymbol{c}_1^\top & \boldsymbol{c}_2^\top \end{bmatrix}, \quad d = p_1 d_1 d_2, \\ \boldsymbol{f}^\top &= d_2 \begin{bmatrix} p_2 c_{11}^2 & 2p_2 c_{11} c_{12} & \dots & D p_D c_{1M_\mathrm{w}} d_1^{D-1} & p_D d_1^D \end{bmatrix}. \end{aligned} \tag{4.52}$$

The monomial vectors $\boldsymbol{\zeta}$ and $\boldsymbol{\eta}$ contains polynomial terms in the original states $\boldsymbol{x}_k^w$ and input u_k.

4.5.4 Diagonal Volterra Kernel-Based Modelling of WH system

Kibangou and Favier (2006) shows that the diagonal coefficients of its kernels completely characterise the Volterra model associated with the WH system. Consequently, the

effort dedicated to the estimation process of a Volterra model associated with the WH structure can be limited to estimating the diagonal coefficients. To compute the non-diagonal coefficients of the equivalent Volterra model, the authors propose an approach based on the bilinear decomposition of a structured matrix $\boldsymbol{H} = \boldsymbol{G}\boldsymbol{L}$, where $\boldsymbol{G}$ encodes the Toeplitz structure of the output linear filter and $\boldsymbol{L}$ gathers the Volterra kernels of the Wiener part of the WH system. Exploiting the structure of the WH system, they formulate the Volterra coefficient estimation as a matrix factorisation problem using an Alternating Least Squares (ALS) algorithm. Given the Toeplitz structure of $\boldsymbol{G}$, the decomposition is unique up to scaling and permutation, allowing identification of all Volterra coefficients, including the non-diagonal terms, with no ambiguity.

To solve the problem, the algorithm alternates between estimating $\boldsymbol{G}$ and $\boldsymbol{L}$ by minimising the cost function $||\boldsymbol{H}-\boldsymbol{G}\boldsymbol{L}||_F^2$. The matrix $\boldsymbol{H}$ is partitioned using block Toeplitz matrices, and updates are performed using standard LS solutions. To ensure stability and proper convergence, the estimation is repeated over several random initialisations.

The method is validated using a WH system example, where the input and output filters have known coefficients, and the output nonlinearity is a cubic polynomial. The authors introduce two performance metrics: (1) the square error between the estimated and true $\boldsymbol{H}$, and (2) the normalised projection misalignment between the estimated and true $\boldsymbol{g}$, the vector of output linear coefficients. The results indicate good convergence of both error criteria across 500 Monte Carlo runs, with the error dropping below 10^{-6} in most cases.

However, the approach has some limitations. First, the ALS algorithm is sensitive to initialisation, and despite overall convergence, some runs can lead to local minima or slow convergence, requiring multiple restarts. Second, the method assumes knowledge of the full structured matrix $\boldsymbol{H}$, which itself must be estimated or known beforehand, potentially introducing errors or constraints in practical applications. Lastly, while the Toeplitz structure simplifies identifiability, the method still relies on noise-free or low-noise settings, and its performance under high noise or model mismatch is not discussed in detail. More advanced tensor-based approaches for modelling nonlinear systems, particularly the Volterra model, are reviewed in Favier and Kibangou (2023).

4.5.5 Tensor-Based Methods for WH Identification

In Ben Ahmed et al. (2013), the authors propose tensor-based methods for identifying nonlinear WH systems, focusing on the fifth-order Volterra kernel, which they treat as a symmetric tensor. They begin by estimating this kernel using i.i.d. input signals, based on closed-form expressions derived under standard assumptions (e.g., ergodicity, zero-mean i.i.d. input, noise independence). Unlike earlier works that used third-order

kernels or required estimating multiple kernels (orders 3–5), their method simplifies the process by focusing solely on the fifth-order kernel. To extract the parameters of the linear subsystems, they apply PARAllel FACtor (PARAFAC) decomposition using two methods: the iterative ALS algorithm and the non-iterative TOeplitz Matrix FActor Computation (TOMFAC) algorithm, which is tailored to handle the Toeplitz structure in one of the factors. This decomposition isolates the linear subsystem parameters, after which the nonlinear block, modelled as a 5th-degree polynomial, is estimated using Recursive Least Squares.

In the iterative ALS procedure, the mode-1 and mode-5 unfolded matrices of the fifth-order symmetric Volterra kernel are given. The idea is to model these matrices using repeated structures that represent the linear subsystems and a scaling vector. Starting with a random guess for this scaling vector, the algorithm alternates between estimating the first subsystem (assumed to have a Toeplitz structure, like a convolution filter) and updating the scaling vector. This process continues iteratively until the solution stabilises. Once the iterations stop, the second subsystem is obtained by normalising the scaling vector. In the end, both linear subsystems are extracted from the original nonlinear kernel.

TOMFAC is a non-iterative approach that uses a higher-order Volterra kernel to improve the estimation. It starts by applying a standard tensor decomposition (PARAFAC) to the kernel, which breaks it into several factor matrices, each corresponding to a subsystem. Then, it uses the special Toeplitz structure of one of the subsystems to better separate and identify it from the others. This is done by selecting and combining specific rows from the factor matrices in a structured way. By solving a set of linear equations built from this structure, the impulse response of the Toeplitz subsystem is accurately recovered, followed by the estimation of the remaining subsystem.

The authors evaluate the performance of their proposed tensor-based identification methods for WH systems using Monte Carlo simulations. They generate 100 different WH models with randomly chosen linear and nonlinear subsystem parameters, and test each under 10 noise realisations, using a specially designed random multilevel input sequence. The performance is measured using Normalised Mean Square Error on the output and the estimated subsystems. Results show that both ALS and TOMFAC algorithms perform better as the SNR increases, with accurate recovery of the system parameters. TOMFAC has the advantage of being non-iterative, whereas ALS is iterative and sensitive to initialisation. However, the accuracy of identifying the nonlinear subsystem is affected by errors in the earlier estimated linear subsystems. Overall, while the Volterra-tensor-based approach allows structured and interpretable estimation of W-H models, it relies heavily on accurate estimation of high-order Volterra kernels, which are difficult to obtain in practice, especially in noisy conditions. Additionally, the method's computational cost and sensitivity to convergence can be limiting factors in real-world applications.

4.5.6 Recent Contributions in Nonlinear System Identification

Recent research in nonlinear system identification has increasingly focused on the use of Volterra series, block-oriented, and PNLSS models. The following selection of works illustrates key developments and applications in these areas. This overview is not exhaustive but focuses on studies most relevant to the scope of this thesis.

• The work in Xu et al. (2025) introduces a kernel-regularised Volterra identification framework specifically designed for WH systems. This approach constructs structured regularisation matrices based on prior knowledge of the system's memory structure, allowing for enhanced identifiability and robustness in the presence of noise. The methodology aligns well with the *prior-informed* kernel estimation strategy proposed in this thesis, particularly in its use of kernel shaping and complexity control to achieve compact Volterra representations.

• In a related direction, the study in Soheili et al. (2024) employs two-dimensional Volterra series expansion in the frequency domain to model nonlinearities in mechanical systems, with an emphasis on fault detection. By exploiting the orthogonality and structural properties of the Volterra series, the authors develop a computationally efficient identification scheme that can isolate nonlinear dynamic behaviours across input–output channels, which is particularly useful for multivariable or distributed parameter systems.

• The article in Sidorov et al. (2024) provides a comparative analysis of two popular methods: direct collocation and least squares, for the identification of black-box Volterra models. The study demonstrates that direct collocation offers superior accuracy and computational efficiency, especially for higher-order systems and under conditions of limited data. This finding supports the growing trend towards using variational or optimisation-based identification schemes over traditional regression approaches in nonlinear modelling.

• In addition to advancements in Volterra-based identification, recent efforts have also revisited PNLSS models, especially in the context of system identification benchmarks and extrapolation challenges. The study in Balaji et al. (2020) investigates the performance of PNLSS and nonlinear modal models under near-resonant excitation scenarios. It provides a detailed analysis of prediction accuracy and generalisation ability, highlighting that PNLSS models can effectively capture both amplitude-dependent and frequency-dependent behaviours in lightly damped systems.

• The benchmark study in Champneys et al. (2024) offers a comprehensive evaluation of several nonlinear system identification techniques, including PNLSS, across a wide range of canonical test systems. The work emphasises reproducibility and robustness by providing open-source implementations and consistent evaluation metrics. Notably,

one of the benchmark systems used in Champneys et al. (2024) is also revisited in this thesis in Chapter 7 for validation and comparison, providing a valuable point of reference for the proposed identification approach.

• Lastly, the work in Schüssler and Nelles (2021) explores the extrapolation behaviour of various nonlinear state-space models, including PNLSS, under both training and validation scenarios. The study assesses each model's performance in terms of robustness, sensitivity to initial conditions, and capacity to generalise beyond the excitation regime. These aspects are particularly relevant for applications involving long-term prediction and model-based control, where stable and physically plausible extrapolation is essential.

4.6 Conclusion

This chapter introduced key nonlinear system identification models that form the foundation for the methodologies developed in this thesis. Starting from classical block-oriented structures: Hammerstein, Wiener, and Wiener–Hammerstein, it highlighted their interpretability and relevance in practical applications, along with the inherent challenges in their identification. The PNLSS model was then presented as a flexible alternative capable of capturing complex nonlinear dynamics in state-space form. The Volterra series was also discussed as a general framework for representing nonlinear dynamic systems.

Importantly, the chapter examined the interconnection of these models in the literature, demonstrating how block-oriented structures can be embedded within a Volterra or PNLSS representation. It also reviewed established identification techniques, such as BLA-based pole allocation, diagonal kernel methods, and tensor-based formulations, that help bridge these model classes. These insights serve as a foundation for the kernel-based identification and realisation strategies proposed in the following chapters.

Chapter 5

Prior-Informed Volterra Kernel Estimation for Hammerstein Model

Abstract: This chapter presents a comprehensive modelling framework for identifying nonlinear dynamic systems modelled as Hammerstein structures using *prior-informed* Volterra series representations. The methodology begins with the estimation of the Volterra kernel, where prior knowledge about the block-oriented structure is incorporated to guide the identification process. To ensure stability across all stages of the identification process, shape-promoting regularisation is applied during Volterra kernel estimation. The structure of the Hammerstein kernel is analytically constructed and used to reconstruct the underlying system's static nonlinearity and linear dynamics through low-rank matrix-based approximations. A nonlinear optimisation with additional shape constraints is then performed in the block-oriented model parameter space to refine the estimate. The shape penalty, introduced during the block-oriented structure optimisation, enforces stable filter characteristics, in turn resulting in a stable PNLSS model realisation using subspace techniques, enabling a compact and simulation-ready representation. The performance of the proposed framework is validated through both numerical simulations and real-world measurements. Results show that *prior-informed* initialisation significantly enhances convergence speed and accuracy while ensuring physical interpretability and model stability. The chapter demonstrates that incorporating structural priors and leveraging the low-rank property of the kernel offers an efficient and accurate approach to identifying complex nonlinear systems.

Note on Chapter Content: This chapter comprises both previously published and unpublished work by the PhD candidate, supplemented with selected material from existing literature to provide context and facilitate comparison. Specifically:

- Sections 5.1, 5.2, 5.2.2, 5.2.3, and some parts of Section 5.3 are based on P. Vijayan, P. Dreesen, J. Lataire and M. Ishteva, "Enhancing Block-Oriented System Identification Using *Prior-Informed* Volterra Kernels," in *IEEE Transactions on Instrumentation and Measurement*, vol. 74, pp. 1–11, 2025, Article No. 6507311.
DOI: 10.1109/TIM.2025.3575968.

- Sections 5.2.1 and 5.2.4 present original, unpublished work by the PhD candidate.

This research was conducted by the PhD candidate under the supervision of the co-authors, who served as (co-)promoters.

5.1 Representation of Volterra Kernel

The discrete-time SISO Volterra model output at time k, described in (1.16), represents the system's output as a weighted sum of past inputs, where the higher-order kernels account for increasingly complex nonlinear interactions between past and current inputs. The identification procedure starts from input-output data obtained through simulation, measurement, or benchmark experiments. It is assumed that the system can be described by a known block-oriented structure (e.g., Hammerstein, Wiener, or Wiener–Hammerstein) and that time-domain input and output signals are available, sampled at discrete intervals. This assumption enables the system to be modelled using a structured Volterra representation suitable for parameter identification.

The focus of this section is not on identifying the Volterra kernels from data, but rather on expressing the model output into a structured regression form that separates the input monomials (features) into a design matrix and the unknown kernel coefficients into a parameter vector. This representation forms the basis for later stages of model fitting and transformation. Importantly, this formulation incorporates prior information about the system's structure and avoids the need for iterative kernel estimation procedures, enabling efficient downstream parameter recovery.

Noiseless Case: The following formulation assumes an ideal (noiseless) setting, where the system output is exactly described by the Volterra series. Equation (1.16) expresses the Volterra model as a summation over increasing nonlinear orders and memory. Rewriting (1.16) in a compact vector form by stacking all monomials of the

input (up to degree D and memory length M) into a feature vector and arranging the corresponding Volterra kernel into a parameter vector, the system output y_k can be given as their inner product:

$$y_k = \begin{bmatrix} 1 \\ \hdashline u_k \\ \vdots \\ u_{k-M} \\ \hdashline u_k^2 \\ u_k u_{k-1} \\ \vdots \\ u_{k-M}^2 \\ \hdashline \vdots \\ \hdashline u_k^D \\ \vdots \\ u_{k-M}^D \end{bmatrix}^\top \begin{bmatrix} h_0 \\ \hdashline h_1(0) \\ \vdots \\ h_1(M) \\ \hdashline h_2(0,0) \\ h_2(0,1) \\ \vdots \\ h_2(M,M) \\ \hdashline \vdots \\ \hdashline h_D(0,\dots,0) \\ \vdots \\ h_D(M,\dots,M) \end{bmatrix}. \tag{5.1}$$

For convenience, let $I = M+2$ and define the vector $\boldsymbol{u}_k = \begin{bmatrix} 1 & u_k & \cdots & u_{k-M} \end{bmatrix}^\top \in \mathbb{R}^I$. The vector $\boldsymbol{u}_k^D$ is defined as

$$\boldsymbol{u}_k^D = \underbrace{\boldsymbol{u}_k \otimes \cdots \otimes \boldsymbol{u}_k}_{D \text{ times}} \in \mathbb{R}^{I^D}, \tag{5.2}$$

where $\otimes$ denotes the Kronecker product[1]. Therefore, the vector $\boldsymbol{u}_k^D$ contains monomials of lagged inputs of degree 0 to D. The model (5.1) can be compactly rewritten using definition (5.2) as

$$y_k = (\boldsymbol{u}_k^D)^\top \boldsymbol{h}, \tag{5.3}$$

[1]The Kronecker product of two matrices $\boldsymbol{A} \in \mathbb{R}^{I\times J}$ and $\boldsymbol{B} \in \mathbb{R}^{K\times L}$ is defined as:

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \dots & a_{1J}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \dots & a_{2J}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\boldsymbol{B} & a_{I2}\boldsymbol{B} & \dots & a_{IJ}\boldsymbol{B} \end{bmatrix} \in \mathbb{R}^{IK\times JL},$$

where each entry a_{ij} of $\boldsymbol{A}$ is multiplied by the entire matrix $\boldsymbol{B}$. The element-wise definition is given by:

$$(\boldsymbol{A} \otimes \boldsymbol{B})_{(i-1)K+k,(j-1)L+l} = a_{ij} b_{kl}, \quad \forall 1 \le i \le I,\ 1 \le j \le J,\ 1 \le k \le K,\ 1 \le l \le L.$$

where $\boldsymbol{h} \in \mathbb{R}^{I^D}$ is the Volterra kernel coefficients for orders 0 to D. Given N training samples, the following system of linear equations is obtained:

$$\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}, \tag{5.4}$$

where the Volterra kernel vector $\boldsymbol{h}$ when reshaped into a D-way tensor, is symmetric. In (5.4), $\boldsymbol{U} \in \mathbb{R}^{N \times I^D}$ is the input regression matrix and $\boldsymbol{y} \in \mathbb{R}^N$ is the output vector.

Noisy Case: In practice, the system output is often corrupted by measurement noise. In this work, we assume that the input signal is noise-free and only the output is affected by noise. This assumption is standard in system identification, where the input is typically a designed excitation (e.g., multisine or pseudorandom binary sequence) or a digitally generated signal with negligible noise. Disturbances are predominantly present at the output due to sensor imperfections or environmental factors. Under this setting, if additive noise b_k is present, the measured output is given by:

$$y_k^{\text{meas}} = (\boldsymbol{u}_k^D)^\top \boldsymbol{h} + b_k, \tag{5.5}$$

and the system of equations becomes:

$$\boldsymbol{y}^{\text{meas}} = \boldsymbol{U}\boldsymbol{h} + \boldsymbol{b}, \tag{5.6}$$

where $\boldsymbol{b} \in \mathbb{R}^N$ denotes the noise vector. This case is handled via least-squares estimation, as described in later sections. However, the structured representation of the model remains unchanged. The system identification problem now reduces to identifying $\boldsymbol{h}$, for the given finite sequence of measured signals $\{u_k, y_k^{\text{meas}}\}_{k=1}^N$, order D, and memory length M.

To better illustrate the underlying Volterra kernel structure, the kernel vector $\boldsymbol{h}$ can be reshaped into a D-way tensor of size $I \times \cdots \times I$ (D times). While this transformation does not affect the technical aspects of the model, it serves as a valuable tool for understanding, highlighting a kernel structure that is difficult to discern in vector form alone, especially for models in Section 4.2. For a second-order Volterra model with memory M, the kernel vector can be reshaped into a symmetric matrix $\boldsymbol{H}$ of size $I \times I$ as in (5.7). For a third-order system, the kernel tensor can be visualised as in Figure 5.1.

$$\boldsymbol{H} = \left[\begin{array}{c:ccc} h_0 & \frac{h_1(0)}{2} & \cdots & \frac{h_1(M)}{2} \\ \hdashline \frac{h_1(0)}{2} & h_2(0,0) & \cdots & h_2(M,0) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{h_1(M)}{2} & h_2(0,M) & \cdots & h_2(M,M) \end{array}\right] \tag{5.7}$$

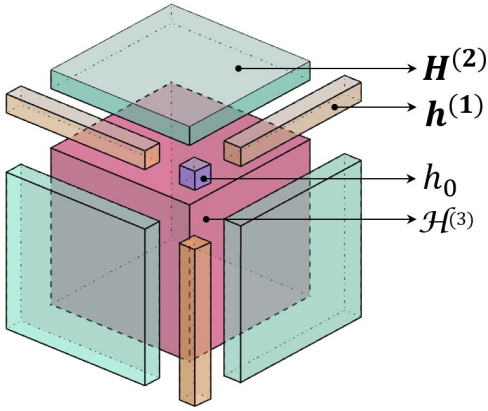


Figure 5.1: Illustration of 3-way Volterra kernel tensor of size $I \times I \times I$. The element at position $(1, 1, 1)$ corresponds to the constant $h_0 \in \mathbb{R}$. The vectors $\boldsymbol{h}^{(1)} \in \mathbb{R}^{I-1}$ in all directions represent the coefficients of linear terms. The submatrix $\boldsymbol{H}^{(2)} \in \mathbb{R}^{I-1 \times I-1}$ in all directions represents the coefficients of second-order monomials. The sub-tensor $\mathcal{H}^{(3)} \in \mathbb{R}^{I-1 \times I-1 \times I-1}$ represents the coefficients of third-order monomials. This figure is inspired by the work in Debals et al. (2017).

The specific configuration of the block-oriented model determines the structure of the Volterra kernel, as will be discussed in this and the subsequent chapters. To streamline terminology, we refer to the resulting structured kernels as Hammerstein kernel, Wiener kernel, and WH kernel, corresponding to the Volterra representations of Hammerstein, Wiener, and Wiener–Hammerstein systems, respectively. In this chapter, we focus on the Hammerstein kernel, while the Wiener and WH kernels are addressed in detail in the following two chapters.

5.2 Construction of Hammerstein Kernel

Assuming a polynomial nonlinearity of degree D and an impulse response $\boldsymbol{t}$ of order M_{h}, the input-output relationship of the Hammerstein structure in Figure 5.2, is given by:

$$y_k = \sum_{i=0}^{M_{\mathrm{h}}} t_i \sum_{j=1}^{D} p_j v_{k-i}^j = \sum_{j=1}^{D} p_j \left(\sum_{i=0}^{M_{\mathrm{h}}} t_i v_{k-i}^j \right). \tag{5.8}$$

The analysis and construction of the Hammerstein kernel presented in this section assumes an idealised, noise-free setting. That is, the output y_k is generated purely from

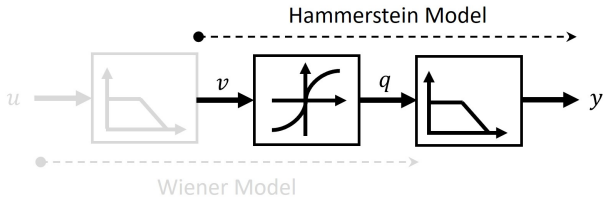


Figure 5.2: Hammerstein model with a linear dynamic block following a static nonlinear block. v is the input, y is the output, q is the intermediate signal.

the deterministic model without any additive disturbance. This assumption allows a direct algebraic correspondence between the Hammerstein structure and its Volterra representation.

To illustrate the equivalence between this Hammerstein model and a Dth-order finite memory Volterra model, consider the minimal working example:

Example 12. For degree $D = 2$ and memory $M_{\text{h}} = 2$, the model (5.8) expands to:

$$\begin{aligned} y_k &= p_1(t_0 v_k + t_1 v_{k-1} + t_2 v_{k-2}) + p_2(t_0 v_k^2 + t_1 v_{k-1}^2 + t_2 v_{k-2}^2) \\ &= \begin{bmatrix} v_k \\ v_{k-1} \\ v_{k-2} \\ v_k^2 \\ v_{k-1}^2 \\ v_{k-2}^2 \end{bmatrix}^\top \begin{bmatrix} p_1 t_0 \\ p_1 t_1 \\ p_1 t_2 \\ p_2 t_0 \\ p_2 t_1 \\ p_2 t_2 \end{bmatrix}. \end{aligned} \tag{5.9}$$

This formulation reveals that the Hammerstein model imposes a specific structure on the parameters. Since the linear dynamic block follows the nonlinear transformation, there are no cross-terms between input samples at different time steps (e.g. v_k, v_{k-1}). However, expanding the Volterra model (1.16) for the same hyperparameters D and M_{h}, gives:

$$y_k = \left(\begin{bmatrix} 1 \\ v_k \\ v_{k-1} \\ v_{k-2} \end{bmatrix} \otimes \begin{bmatrix} 1 \\ v_k \\ v_{k-1} \\ v_{k-2} \end{bmatrix} \right) \boldsymbol{h}. \tag{5.10}$$

It can be seen that the general Volterra model includes all the input monomials. Therefore, the parameters corresponding to the cross-terms that are not included in the Hammerstein model (5.9) are zero.

To derive the block-oriented model parameters from the Volterra kernel, we expand and rewrite (5.8) for order $D = 2$ in the form of the Volterra model (5.4). Thus, we get the reshaped kernel tensor (5.11).

$$\boldsymbol{H} = \left[\begin{array}{c|cccc} 0 & \frac{p_1 t_0}{2} & \frac{p_1 t_1}{2} & \dots & \frac{p_1 t_{M_\mathrm{h}}}{2} \\ \hline \frac{p_1 t_0}{2} & p_2 t_0 & 0 & \dots & 0 \\ \frac{p_1 t_1}{2} & 0 & p_2 t_1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \frac{p_1 t_{M_\mathrm{h}}}{2} & 0 & \dots & 0 & p_2 t_{M_\mathrm{h}} \end{array}\right] \tag{5.11}$$

The first and second-order kernels are the impulse response $\boldsymbol{t}$ scaled by coefficients p_1 and p_2, respectively. This kernel structure can be generalised for any Hammerstein model structure with higher-order nonlinearity. For a D-way tensor, the kernel components of orders 1 through $D - 1$ are symmetrically scaled by a factor $1/D$ along each of the D dimensions to preserve the symmetric structure of the kernel. A third-order Volterra kernel admitting Hammerstein structure is shown in Figure 5.3.

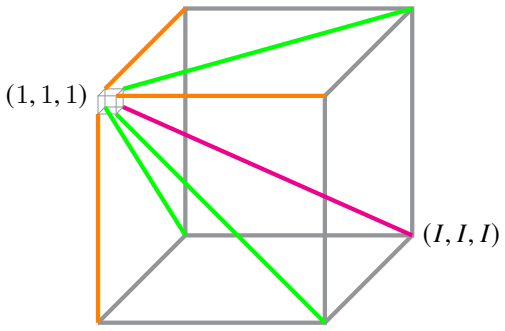


Figure 5.3: Illustration of a 3-way symmetric Hammerstein kernel tensor showing diagonal kernel in all orders.

The structural prior is the absence of cross-term nonlinearity, i.e., the system is diagonal in the Volterra representation. This means that the off-diagonal elements in the Volterra

kernel are zero:

$$h_d(m_1, m_2, \ldots, m_d) = \begin{cases} h_d(m, m, \ldots, m), & \text{if } m_1 = m_2 = \cdots = m_d \\ 0, & \text{otherwise.} \end{cases} \tag{5.12}$$

Thus, the simplified Volterra expansion for the Hammerstein system is:

$$y_k = \sum_{d=1}^{D} \Bigg(\underbrace{\sum_{m=0}^{M_\mathrm{h}} \cdots \sum_{m=0}^{M_\mathrm{h}}}_{d\,\text{times}} h_d(\underbrace{m, \cdots, m}_{d\,\text{times}}) v_{k-m}^d \Bigg), \tag{5.13}$$

where now the kernel coefficients h_d depend only on a single lag m. Since there is no cross-term nonlinearity, some coefficients in the Volterra kernel vector $\boldsymbol{h}$ must be zero. Therefore, in the noiseless case, the columns in $\boldsymbol{U}$ corresponding to zero coefficients can be eliminated, yielding a reduced representation:

$$\boldsymbol{y} = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{h}}, \tag{5.14}$$

where $\tilde{\boldsymbol{U}} \in \mathbb{R}^{N \times D(M_\mathrm{h}+1)}$ is the reduced input matrix obtained by removing columns associated with zero kernel terms, and $\tilde{\boldsymbol{h}}$ is the reduced kernel vector containing only the $D(M_\mathrm{h} + 1)$ non-zero entries. This prior-based construction reduces the number of unknowns, making the estimation more efficient and less prone to overfitting.

In the noisy case, the output vector $\boldsymbol{y}$ is replaced with a noisy version:

$$\boldsymbol{y}^{\mathrm{meas}} = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{h}} + \boldsymbol{b}, \tag{5.15}$$

where $\boldsymbol{b} \in \mathbb{R}^N$ is additive output noise. In this case, the identification of the Hammerstein kernel becomes a linear regression problem, where the structure in $\tilde{\boldsymbol{U}}$ still allows efficient and robust estimation. To mitigate ill-conditioning or overfitting in the presence of noise, regularisation or SVD-based techniques are typically applied.

5.2.1 *Prior-Informed* (Regularised) Kernel Estimation

Under ideal (noiseless) conditions, the reduced kernel vector $\tilde{\boldsymbol{h}}$ is estimated by solving the simplified linear system (5.14) using standard least-squares, when $\tilde{\boldsymbol{U}}$ has full column rank. This estimation step is straightforward and non-iterative. Importantly, the reduced problem size, enabled by incorporating prior structural knowledge, makes the solution both computationally efficient and less sensitive to overfitting.

In practical scenarios, however, only noisy measurements of the output are available. As discussed previously, this study assumes that the inputs are noise-free and that additive noise affects only the output. In such cases, the solution to the resulting overdetermined

linear system (5.15) remains the same in structure, but its accuracy may degrade in the presence of noise.

Although the reduced input matrix $\tilde{\boldsymbol{U}}$ is theoretically full-rank under ideal conditions, it may become rank-deficient or ill-conditioned in practice. This issue typically arises due to insufficient excitation of the system, such as when the input signal is highly correlated, smooth, periodic or lacks sufficient variability. Additionally, redundant or collinear terms can occur when using discretised or low-amplitude input signals, especially in higher-order monomials. These cases result in linearly dependent columns in $\tilde{\boldsymbol{U}}$, impairing the numerical stability and reliability of parameter estimation.

To mitigate rank deficiency or numerical instability, it is standard to apply regularisation techniques or singular value truncation when solving the system. By decomposing the matrix into its orthogonal modes and singular values, SVD enables stable computation of the least-squares solution even when some directions in the input space are weakly excited or numerically degenerate. Truncating small singular values or combining SVD with regularisation (e.g., Tikhonov) ensures that the solution avoids noise amplification and improves robustness.

In this work, the SVD is first applied to the matrix $\tilde{\boldsymbol{U}}$:

$$\tilde{\boldsymbol{U}} = \boldsymbol{Q}_1 \boldsymbol{S} \boldsymbol{V}_1^\top, \tag{5.16}$$

where $\boldsymbol{Q}_1 \in \mathbb{R}^{N \times r}$ and $\boldsymbol{V}_1 \in \mathbb{R}^{D(M_\mathrm{h}+1) \times r}$ contain the left and right singular vectors associated with the non-zero singular values, $\boldsymbol{S} \in \mathbb{R}^{r \times r}$ is a diagonal matrix of the non-zero singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$, and $r = \operatorname{rank}(\tilde{\boldsymbol{U}}) = D(M_\mathrm{h} + 1)$. Then, the unique solution to the noiseless system (5.14) is:

$$\tilde{\boldsymbol{h}} = \tilde{\boldsymbol{U}}^\dagger \boldsymbol{y} = \boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^\top \boldsymbol{y}, \tag{5.17}$$

whereas in the noisy case, where measurements follow (5.15), the corresponding pseudoinverse-based solution is given by

$$\tilde{\boldsymbol{h}} = \tilde{\boldsymbol{U}}^\dagger \boldsymbol{y}^{\mathrm{meas}} = \boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^\top \boldsymbol{y}^{\mathrm{meas}}. \tag{5.18}$$

To ensure physical interpretability and robustness in the proposed Hammerstein system identification framework, we adopt a two-stage regularisation and refinement strategy applied to the Volterra kernel. In the first stage, *prior-informed* regularisation is used during the kernel estimation to promote smoothness, decay, and structural alignment with the Hammerstein model. In the second stage (Section 5.2.3), these properties are further refined through shape-enforced optimisation of the reconstructed block-oriented model. This two-tier approach ensures that all intermediate models remain stable, interpretable, and consistent with the underlying system dynamics.

Stage 1: Shape-Enforcing Refinement
While this estimate benefits from the structural prior and SVD-based numerical stability,

it may still be affected by noise and fail to accurately represent the true system dynamics. To improve interpretability and generalisation, we introduce an optional shape-enforcing refinement step. This step is particularly beneficial when the system is complex, highly nonlinear, or corrupted by significant measurement noise, as is often the case in real-world data. In contrast, when the underlying system is known and well-defined (e.g., in simulation studies), the base algorithm already captures the dynamics accurately, eliminating the need for additional constraints and allowing for faster training. The trade-off between training time and robustness is discussed further in Sections 5.3.1 and 5.3.2, where both scenarios are evaluated.

To refine the kernel $\tilde{\boldsymbol{h}}$, a linear least-squares optimisation is formulated to minimise the output error while simultaneously enforcing physically interpretable shape characteristics across the individual kernel components. The objective function includes regularisation terms that encourage both smoothness and decay across kernel orders up to D:

$$\min_{\tilde{\boldsymbol{h}}} J(\tilde{\boldsymbol{h}}) = \underbrace{\left\|\tilde{\boldsymbol{U}}\tilde{\boldsymbol{h}} - \boldsymbol{y}\right\|_2^2}_{\text{data fit}} + \underbrace{\sum_{d=1}^{D} \mu_d \left\|\boldsymbol{w} \odot \tilde{\boldsymbol{h}}^{(d)}\right\|_2^2}_{\text{decay}} + \underbrace{\sum_{d=1}^{D} \lambda_d \left\|\boldsymbol{\Delta}^2 \tilde{\boldsymbol{h}}^{(d)}\right\|_2^2}_{\text{curvature}}, \tag{5.19}$$

where $\tilde{\boldsymbol{h}}^{(d)}$ denote the dth-order segment of the stacked kernel $\tilde{\boldsymbol{h}}$, $\odot$ denotes element-wise multiplication, and $\boldsymbol{\Delta}^2$ is the second-order difference operator of the Toeplitz form:

$$\boldsymbol{\Delta}^2 = \begin{bmatrix} 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \end{bmatrix} \in \mathbb{R}^{(M_\text{h}-2)\times M_\text{h}}, \tag{5.20}$$

that penalises curvature. The regularisation weights λ_d and μ_d control the trade-off between smoothness and decay, respectively. Moreover, these weights can be scaled across kernel orders to reflect the prior assumption that higher-order kernels should be both smoother and more rapidly decaying than lower-order ones. The decay weights are imposed through the vector $\boldsymbol{w} = \begin{bmatrix} 1 & \dots & M_\text{h} \end{bmatrix}^\top$, which penalises energy at later lags and encourages concentration of kernel energy near the origin, typical of physically meaningful impulse responses. This choice of a linearly increasing weight (ramp function) reflects a moderate and interpretable decay prior that penalises later lags more heavily, but not as aggressively as exponential decay or power-law weight functions. While exponential decay is commonly used when strong prior knowledge about rapid system response attenuation exists, the ramp vector provides a softer regularisation profile, allowing greater flexibility in the tail of the kernel. This can be beneficial in our case, where decay behaviour is expected but may vary significantly depending on the system dynamics. Specifically, low-pass, high-pass, band-pass and band-stop filters

can all lead to qualitatively different temporal energy distributions in their impulse responses.

Equation (5.19) results in a *prior-informed*, shape-regularised kernel $\tilde{\boldsymbol{h}}_{\mathrm{reg}}$, which forms the basis for subsequent extraction of block-oriented model parameters and system realisation. This form of regularisation is conceptually aligned with approaches discussed in Birpoutsoukis et al. (2017, 2018).

5.2.2 Reconstructing Hammerstein Structure using Hammerstein Kernel

This subsection marks a key intermediate step in the overall modelling framework of this thesis (see Figure 1.6). Starting from the *prior-informed* (and shape-regularised) Volterra kernel, we aim to reconstruct the underlying Hammerstein block structure. This structural insight lays the foundation for the final stage: identifying a PNLSS model that captures the full system dynamics in a compact state-space form.

The Hammerstein kernel $\tilde{\boldsymbol{h}}$ (if regularised $\tilde{\boldsymbol{h}}_{\mathrm{reg}}$) $\in \mathbb{R}^{D(M_{\mathrm{h}}+1)}$, obtained from the previous sections, is rearranged in a matrix $\boldsymbol{R}$ of size $(M_{\mathrm{h}}+1) \times D$ as follows:

$$\boldsymbol{R} = \begin{bmatrix} p_1 t_0 & \dots & p_D t_0 \\ \vdots & \vdots & \vdots \\ p_1 t_{M_{\mathrm{h}}} & \dots & p_D t_{M_{\mathrm{h}}} \end{bmatrix} = \begin{bmatrix} h_1(0) & \dots & h_D(0, \cdots, 0) \\ \vdots & \vdots & \vdots \\ h_1(M_{\mathrm{h}}) & \dots & h_D(M_{\mathrm{h}}, \cdots, M_{\mathrm{h}}) \end{bmatrix}. \tag{5.21}$$

Each column of $\boldsymbol{R}$ corresponds to the impulse response coefficients scaled by one of the polynomial coefficients p_j. Hence, this matrix captures both the linear dynamics (in the rows) and the nonlinear static nonlinearity (in the columns).

In the ideal noiseless case, $\boldsymbol{R}$ is exactly rank-one due to the separable structure of the Hammerstein model. The Volterra linear kernel, i.e., the first column of the matrix $\boldsymbol{R}$, equals the impulse response scaled by p_1. In the noisy case, matrix $\boldsymbol{R}$ is typically full-rank but remains *approximately* low-rank. This low-rank structure enables us to extract the model parameters using SVD. The rank-one approximation $\tilde{\boldsymbol{R}}$ of $\boldsymbol{R}$ is obtained as:

$$\boldsymbol{R} \approx \sum_{i=1}^{r} \sigma_i \boldsymbol{\alpha}_i \boldsymbol{\beta}_i^* = \tilde{\boldsymbol{R}}, \tag{5.22}$$

where $\boldsymbol{\sigma}_i$ are the singular values, and $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are the left and right singular vectors, respectively. In this case, only the dominant component (i.e., $r = 1$) is retained due to the model's rank-one structure.

To uniquely identify the block parameters, the impulse response estimates $\hat{\boldsymbol{t}}$ are obtained from the first column of $\tilde{\boldsymbol{R}}$, assuming $p_1 = 1$. Columns 2 to D of $\tilde{\boldsymbol{R}}$ then provide estimates $\hat{p}_2, \dots, \hat{p}_D$, respectively.

While this approach provides a direct and interpretable estimate of the Hammerstein block-oriented model, it may still be suboptimal in noisy scenarios. Therefore, the parameters obtained here serve as a structured initialisation for a subsequent nonlinear optimisation (see Section 5.2.3), where the model is further refined to improve accuracy and robustness. The low-rank nature of $\boldsymbol{R}$ is thus not only central to parameter extraction but also acts as a valuable constraint that enhances noise resilience and interpretability throughout the modelling framework.

5.2.3 Nonlinear Optimisation in the Block-oriented Model Parameter Space

Following the extraction of the Hammerstein block parameters $(\hat{\boldsymbol{t}}, \hat{p}_1, \ldots, \hat{p}_D)$ via low-rank approximation of the (regularised) Volterra kernel, a second optimisation step is performed to refine the model. The goal of this stage is to improve the accuracy of output prediction while enforcing physically realistic and interpretable shapes on the model parameters, particularly the linear dynamic component (impulse response).

This stage of optimisation operates directly in the parameter space of the reconstructed block-oriented model (e.g., Hammerstein structure). Beginning with the reconstructed estimates, the model parameters are iteratively adjusted so that the output more accurately approximates the measured system response. Once optimised, these refined parameters form the basis for identifying the final PNLSS model via a subspace-based method. The optimisation procedure described here is also applicable to the Wiener and WH systems discussed in Chapters 6 and 7, with appropriate modifications tailored to their specific structures.

The estimated impulse response $\hat{\boldsymbol{t}}$, and the static nonlinearity coefficients $\hat{p}_j, j = 1, \cdots, D$, derived from the low-rank approximation of the Volterra kernel of Hammerstein structure, can be further optimised using a nonlinear least-squares solver. The initial block parameter estimates are composed in a vector $\boldsymbol{\theta}$ (say) and the following cost function $V(\boldsymbol{\theta})$ is minimized:

$$\min_{\boldsymbol{\theta}} \; V(\boldsymbol{\theta}) = \sum_{k} (\Phi(\boldsymbol{u}_k, \boldsymbol{\theta}) - y_k)^2, \tag{5.23}$$

where $\Phi(\boldsymbol{u}_k, \boldsymbol{\theta})$ is a function describing the Hammerstein model output at time k, computed from the input vector $\boldsymbol{u}_k \in \mathbb{R}^{M_\mathrm{h}+1}$ and parameter vector $\boldsymbol{\theta}$. The true output is denoted by y_k. For instance, for the Hammerstein model, the parameter vector $\boldsymbol{\theta}$ is structured using the initial estimates (from Section 5.2.2) as follows:

- $\boldsymbol{\theta}^{\mathrm{NL}}$: The first D elements correspond to the coefficients of the polynomial nonlinear static function $\hat{p}_1, \ldots, \hat{p}_D$. The estimates are taken up to an order of the Volterra model D, with additional degrees of freedom introduced to allow the

optimiser to fit a higher-order polynomial ($D_+ > D$) if needed. These additional coefficients are zero-initialised.

- $\boldsymbol{\theta}^{\mathrm{L}}$: The next set of elements represents the estimated impulse response of the linear filter $\hat{\boldsymbol{t}}$. Similarly, additional zero-initialised parameters are included to allow the filter to be adapted with a longer memory ($M_{\mathrm{h}+} > M_{\mathrm{h}}$) if needed.

Thus, the vector $\boldsymbol{\theta}$ is constructed to provide sufficient flexibility while capturing both static and dynamic model components and keeping the model interpretable. The function $\Phi(\boldsymbol{u}_k, \boldsymbol{\theta})$ represents the system structure, where the input $\boldsymbol{u}_k$ first passes through a static nonlinear function (modelled as a polynomial) and then through a linear dynamic system (modelled as a filter). The corresponding mathematical discrete-time representation is

$$\Phi\left(\boldsymbol{u}_k, \begin{bmatrix} \boldsymbol{\theta}^{\mathrm{NL}} \\ \boldsymbol{\theta}^{\mathrm{L}} \end{bmatrix}\right) = \sum_{i=0}^{M_{\mathrm{h}+}} \hat{t}_i \sum_{j=1}^{D_+} \hat{p}_j u_{k-i}^j = \sum_{i=0}^{M_{\mathrm{h}+}} \theta_{D_+ + 1 + i} \sum_{j=1}^{D_+} \theta_j u_{k-i}^j. \tag{5.24}$$

The above equation follows the same structure as (5.8), but here, the parameters are optimised estimates, and the hyperparameters $M_{\mathrm{h}+}$, D_+ may be larger than those used in the Volterra model to provide additional flexibility.

Stage 2: Optional Shape-Enforced Optimisation
In addition to minimising the output error, the optimisation can optionally encourage the estimated impulse response to exhibit desired shape properties, such as monotonic decay, energy concentration near the origin, and smoothness. This is achieved by incorporating shape-promoting regularisation terms into the cost function, in a manner consistent with the kernel-based formulation in (5.19). The resulting objective function V is:

$$\min_{\boldsymbol{\theta}} \; V(\boldsymbol{\theta}) = \underbrace{\sum_k \left(\Phi(\boldsymbol{u}_k, \boldsymbol{\theta}) - y_k\right)^2}_{\text{data fit}} + \underbrace{\mu \left\| \boldsymbol{w} \odot \boldsymbol{\theta}^{\mathrm{L}} \right\|_2^2}_{\text{decay}} + \underbrace{\lambda \left\| \boldsymbol{\Delta}^2 \boldsymbol{\theta}^{\mathrm{L}} \right\|_2^2}_{\text{curvature}}, \tag{5.25}$$

where $\boldsymbol{w} = \begin{bmatrix} 1 & \dots & M_{\mathrm{h}+} \end{bmatrix}^\top$ is a linearly increasing vector (as in (5.19)) that penalises large values at later taps to encourage soft decay, and $\boldsymbol{\Delta}^2$ is the second-order difference operator (as in (5.20), with appropriate dimension) promoting smoothness. The weights μ and λ determine the relative importance of the decay and smoothness terms, respectively.

The need for shape enforcement is particularly important in noisy scenarios, where the initial parameter estimates may be inaccurate or uninformative. Here, the optimiser compensates for imperfections in the Volterra-based initialisation. In noise-free settings, however, the optimisation tends to converge rapidly, as the initial parameters are already close to the true values.

The optimisation procedure for the Wiener and WH models follows the same principle, with structural modifications specific to each model as detailed in their respective chapters. The optimisation is performed using the Levenberg-Marquardt (LM) algorithm implemented via the `lsqcurvefit` routine in MATLAB, which is well-suited for nonlinear least-squares problems.

Although this refinement step significantly improves parameter accuracy, especially in noisy settings, it introduces additional computational overhead. In Sections 5.3.1 and 5.3.2, we benchmark the proposed approach for some Hammerstein systems, including this nonlinear optimisation stage, against conventional methods such as BLA and N4SID, followed by PNLSS optimisation, to quantify both performance and efficiency.

5.2.4 Realising PNLSS Model from the Optimised Hammerstein Kernel

Following the nonlinear optimisation of the Hammerstein block-oriented model (Section 5.2.3), this subsection focuses on realising an equivalent PNLSS model (ref. (4.36)-(4.39)). The transformation bridges the parameter space of block-oriented models and state-space representations, enabling a unified nonlinear framework that can be further analysed or compared against existing techniques such as BLA- or N4SID-initialised PNLSS models. This model realisation step completes the final stage of the proposed identification framework (see Figure 1.6).

Since the Hammerstein parameters have already been optimised to account for possible model mismatches, nonlinearity degrees, and extended memory lengths, the to-be-realised PNLSS model's output is expected to match the output of the optimised Hammerstein block-oriented model closely. The goal here is not further optimisation but a structured transformation, preserving system dynamics while translating them into a PNLSS-compatible form. Specifically, the optimised impulse response coefficients $\boldsymbol{\theta}^{\mathrm{L}}$ and polynomial nonlinearity coefficients $\boldsymbol{\theta}^{\mathrm{NL}}$ are used to form the state-space parameters $\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d$, as well as the nonlinear parameters $\boldsymbol{E}, \boldsymbol{f}$.

The first step involves reconstructing a discrete-time LSS form from the optimised impulse response vector $\boldsymbol{\theta}^{\mathrm{L}}$. In the ideal, noiseless settings, where the LTI block is an FIR model of order $M_{\mathrm{h}+}$ and the nonlinear block is a static polynomial of degree D_+, the state dimension (say $M_{\mathrm{h}-}$) of the PNLSS model is taken as $M_{\mathrm{h}-} = M_{\mathrm{h}+}$. In this

case, the PNLSS parameters (4.39) take the following form theoretically:

$$\boldsymbol{A} = \begin{bmatrix} 0 & \dots & \dots & \dots & 0 \\ 1 & \ddots & & & \vdots \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \theta_1^{\mathrm{NL}} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \boldsymbol{c} = \begin{bmatrix} \theta_2^{\mathrm{L}} \\ \vdots \\ \theta_{M_{\mathrm{h}+}}^{\mathrm{L}} \end{bmatrix}^\top, \quad d = \theta_1^{\mathrm{NL}}\theta_1^{\mathrm{L}},$$
$$\boldsymbol{E} = \begin{bmatrix} \theta_2^{\mathrm{NL}} & \dots & \theta_{D+}^{\mathrm{NL}} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \boldsymbol{f} = \begin{bmatrix} \theta_2^{\mathrm{NL}} & \dots & \theta_{D+}^{\mathrm{NL}} \end{bmatrix} \theta_1^{\mathrm{L}}. \tag{5.26}$$

In this representation, the matrix $\boldsymbol{A}$ shifts the state as in a tapped delay line (a delay line provides access to its contents at multiple arbitrary intermediate delay lengths). The vector $\boldsymbol{b}$ injects the current input scaled by θ_1^{NL} (i.e., the linear gain of the static nonlinearity). The output map $\boldsymbol{c}$ reconstructs the linear dynamics, and d captures the instantaneous output contribution. The nonlinear terms $\boldsymbol{E}$ and $\boldsymbol{f}$ reflect the contributions from higher-order polynomial terms. This equivalence is purely mathematical. In practice, the numerical realisation of such a state-space model from finite length data is not feasible at this full order. Furthermore, in real-world scenarios, the LTI block may not be an FIR model, and the static nonlinearity may not be strictly polynomial. Additionally, there may be unmodelled nonlinearities and output noise. Under these conditions, it is often advantageous to exploit the modelling flexibility of state-space representations and realise a lower-dimensional model with state dimension $M_{\mathrm{h}-} \ll M_{\mathrm{h}+}$.

Established subspace methods such as N4SID and MOESP (reviewed in Chapter 2) are suitable for this task. However, in this thesis, a simpler approach based on the Eigensystem Realisation Algorithm (ERA) is adopted, as explained later in this section.

Before understanding ERA, it is essential to comprehend the various model orders employed at different stages of the modelling framework, which are tailored to suit the structure and flexibility of each representation. In the context of Volterra-based and block-oriented system identification, memory length plays a fundamental role in determining model complexity and accuracy. The key concepts can be formalised as follows:

Definition 5.2.1 (Initial Memory Length M_{h})**.** An initial memory length M_{h} is used during the estimation of the Hammerstein kernel (see equations (5.17) and (5.18)).

This same length is employed to extract the linear impulse response of the system (equation (5.22)).

Definition 5.2.2 (Extended Memory Length $M_{\mathrm{h}+}$)**.** During the nonlinear optimisation of the Hammerstein model (see Section 5.2.3), the linear block is allowed to grow to a longer memory length $M_{\mathrm{h}+}$ to better capture dynamic behaviour under noise and nonlinearity. This provides more flexibility to fit real-world data.

In the Volterra framework, the memory length M_{h} determines how many past input samples influence the current output and directly controls the number of lag terms in the input monomials. A larger M_{h} enables the model to capture longer-range dependencies and higher-order nonlinear interactions, but also causes an exponential growth in the number of parameters, increasing computational cost and the risk of overfitting. Conversely, a smaller M_{h} limits model expressiveness, possibly leading to underfitting. Thus, selecting M_{h} involves a trade-off between accuracy and computational efficiency.

Eigensystem Realisation Algorithm (ERA) (Juang and Pappa, 1985) is a subspace-based system identification method that constructs a minimal realisation of a discrete-time LTI system directly from its impulse response, under the assumption that the model order $M_{\mathrm{h}-}$ (i.e., the number of states) is known a *priori*. It is widely used in applications such as modal analysis and structural health modelling. In this work, ERA is employed to realise the LSS component of the PNLSS model from the optimised Hammerstein impulse response.

Compared to more general subspace methods such as N4SID and MOESP (see chapter 2), which operate directly on input-output data and are better suited for handling measurement noise and arbitrary excitation signals, ERA is significantly more efficient and numerically robust when the system's impulse response is already available (this is exactly our case here). Unlike N4SID, ERA does not require input-output data projections or noise modelling, making it especially attractive in simulation-based settings or when a reliable estimate of the impulse response is known *a priori*. Additionally, while Kung's realisation (Kung et al., 1983) shares ERA's reliance on Hankel matrices and SVD, ERA offers improved numerical stability and handles noise better (Juang and Pappa, 1985). These properties make ERA a natural choice in this work, where a high-fidelity estimate of the system impulse response is available from the optimised block-oriented model.

When transitioning from the block-oriented model to a PNLSS representation, the memory concept is translated into a state-space realisation.

Definition 5.2.3 (Reduced Model Order $M_{\mathrm{h}-}$)**.** In the context of state-space modelling, the reduced model order $M_{\mathrm{h}-}$ refers to the state dimension of the equivalent LTI block when represented as a state-space model. It corresponds to the rank of the system matrix $\boldsymbol{A}$ in the PNLSS model, or equivalently, the order of an IIR approximation of the system dynamics.

Due to the compactness and modelling flexibility of state-space models, $M_{\mathrm{h}-}$ is often significantly smaller than the memory length M_{h} used in the Volterra or block-oriented model, or the extended length $M_{\mathrm{h}+}$ used during nonlinear optimisation. Subspace-based realisation methods such as ERA are designed to extract the dominant modes of the system and approximate the full impulse response behaviour using a low-dimensional state vector.

As a result, one typically observes the ordering

$$M_{\mathrm{h}-} \ll M_{\mathrm{h}} \ll M_{\mathrm{h}+},$$

reflecting the increasing levels of flexibility, redundancy, and modelling capacity across different stages of the framework.

For a SISO Hammerstein system, the optimised discrete-time impulse response coefficients are $\theta_1^{\mathrm{L}}, \theta_2^{\mathrm{L}}, \ldots, \theta_{M_{\mathrm{h}+}}^{\mathrm{L}}$. These coefficients correspond directly to the system's Markov parameters (2.66) as follows:

$$\theta_1^{\mathrm{NL}} \begin{bmatrix} \theta_1^{\mathrm{L}} \\ \theta_2^{\mathrm{L}} \\ \theta_3^{\mathrm{L}} \\ \vdots \\ \theta_{M_{\mathrm{h}+}}^{\mathrm{L}} \end{bmatrix} = \begin{bmatrix} d \\ \boldsymbol{c}^\top \boldsymbol{b} \\ \boldsymbol{c}^\top \boldsymbol{A} \boldsymbol{b} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{M_{\mathrm{h}+}-1} \boldsymbol{b} \end{bmatrix}, \tag{5.27}$$

where d is the feed-through term, and $\boldsymbol{A}, \boldsymbol{b}$ and $\boldsymbol{c}$ are the state-space parameters to be identified. Since the first coefficient θ_1^{L} directly gives d, the task is to recover $\boldsymbol{A}, \boldsymbol{b}$ and $\boldsymbol{c}$ from the remaining impulse response coefficients.

To achieve this, the following data matrix can be constructed in Hankel form $\boldsymbol{H}_0 \in \mathbb{R}^{r\times s}$:

$$\boldsymbol{H}_0 = \theta_1^{\mathrm{NL}} \begin{bmatrix} \theta_2^{\mathrm{L}} & \theta_3^{\mathrm{L}} & \dots & \theta_s^{\mathrm{L}} \\ \theta_3^{\mathrm{L}} & \theta_4^{\mathrm{L}} & \dots & \theta_{s+1}^{\mathrm{L}} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_r^{\mathrm{L}} & \theta_{r+1}^{\mathrm{L}} & \dots & \theta_{M_{\mathrm{h}+}-1}^{\mathrm{L}} \end{bmatrix}, \tag{5.28}$$

where r and s are user-defined integers satisfying $r + s = M_{\mathrm{h}+}$ and $r, s \geq M_{\mathrm{h}-}$. Rewriting the data matrix in terms of Markov parameters yields:

$$\boldsymbol{H}_0 = \begin{bmatrix} \boldsymbol{c}^\top \boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} & \dots & \boldsymbol{c}^\top \boldsymbol{A}^{s-1}\boldsymbol{b} \\ \boldsymbol{c}^\top \boldsymbol{A}\boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}^2\boldsymbol{b} & \dots & \boldsymbol{c}^\top \boldsymbol{A}^{s}\boldsymbol{b} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1}\boldsymbol{b} & \boldsymbol{c}^\top \boldsymbol{A}^{r}\boldsymbol{b} & \dots & \boldsymbol{c}^\top \boldsymbol{A}^{M_{\mathrm{h}+}-2}\boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \dots & \boldsymbol{A}^{s-1}\boldsymbol{b} \end{bmatrix}. \tag{5.29}$$

Thus, the data matrix equals the product of the observability and controllability matrices:

$$\boldsymbol{H}_0 = \boldsymbol{\Gamma}_r \boldsymbol{\Delta}_s. \tag{5.30}$$

To estimate the state matrix $\boldsymbol{A}$, a time-shifted version of the matrix $\boldsymbol{H}_0$, denoted $\boldsymbol{H}_1 \in \mathbb{R}^{r\times s}$, is similarly constructed:

$$\boldsymbol{H}_1 = \theta_1^{\mathrm{NL}} \begin{bmatrix} \theta_3^{\mathrm{L}} & \theta_4^{\mathrm{L}} & \dots & \theta_{s+1}^{\mathrm{L}} \\ \theta_4^{\mathrm{L}} & \theta_5^{\mathrm{L}} & \dots & \theta_{s+2}^{\mathrm{L}} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{r+1}^{\mathrm{L}} & \theta_{r+2}^{\mathrm{L}} & \dots & \theta_{M_{\mathrm{h}+}}^{\mathrm{L}} \end{bmatrix}. \tag{5.31}$$

Rewriting matrix $\boldsymbol{H}_1$ similarly as in (5.29), the following simplification is obtained:

$$\boldsymbol{H}_1 = \begin{bmatrix} \boldsymbol{c}^\top \\ \boldsymbol{c}^\top \boldsymbol{A} \\ \vdots \\ \boldsymbol{c}^\top \boldsymbol{A}^{r-1} \end{bmatrix} \boldsymbol{A} \begin{bmatrix} \boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \dots & \boldsymbol{A}^{s-1}\boldsymbol{b} \end{bmatrix} = \boldsymbol{\Gamma}_r \boldsymbol{A} \boldsymbol{\Delta}_s. \tag{5.32}$$

From the above two data matrices $\boldsymbol{H}_0$ and $\boldsymbol{H}_1$, the ERA tries to approximate the observability and controllability matrices and isolate the required state matrix $\boldsymbol{A}$.

At this stage, it is important to understand the impact of noise and model structure on linear system identification:

1. **Ideal case:** In the ideal, noiseless case, where the linear block is an FIR filter of order $M_{\mathrm{h}+}$ and the nonlinear block is a static polynomial, the equivalent state-space form in (5.26) can be derived analytically. However, this form cannot be realised numerically. This is because the PNLSS model would then have a state dimension $M_{\mathrm{h}-} = M_{\mathrm{h}+}$, which contradicts the requirement in subspace methods that the Hankel matrix dimensions satisfy $r, s > M_{\mathrm{h}-}$ and $r + s = M_{\mathrm{h}+}$. These conditions are mutually incompatible if $M_{\mathrm{h}-} = M_{\mathrm{h}+}$, making the theoretical form unrealisable in practice. This limitation applies not only to ERA but to other subspace-based methods such as MOESP and N4SID as well.

2. **Low-order IIR system:** Consider a system with a third-order IIR linear block ($M_{\mathrm{h}-} = 3$), a polynomial nonlinear block, and no measurement noise. Suppose the impulse response has been optimised to length $M_{\mathrm{h}+} = 48$. Then, a Hankel matrix $\boldsymbol{H}_0 \in \mathbb{R}^{r \times s}$ can be constructed with $r, s > M_{\mathrm{h}-}$ and $r + s = M_{\mathrm{h}+}$ (e.g., $r = s = 24$). Due to the model mismatch, i.e., representing an IIR system with a finite-length response, the matrix $\boldsymbol{H}_0$ is full-rank. However, its singular values decay rapidly, revealing an approximately low-rank structure. This is shown in Figure 5.4, where only three singular values dominate, consistent with the true model order $M_{\mathrm{h}-} = 3$.

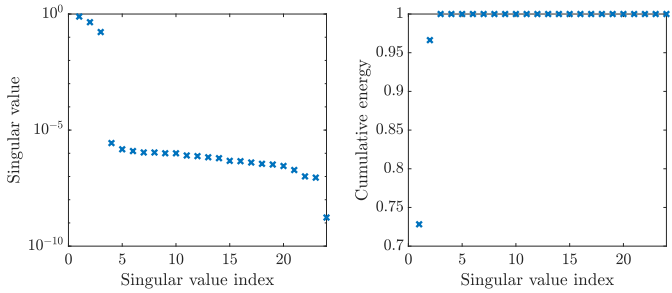


Figure 5.4: SVD reveals an approximate rank-3 structure of $\boldsymbol{H}_0$ in the noiseless case. This reflects the underlying third-order low-pass filter, despite the impulse response being of length 48.

3. **Noisy system:** If the same Hammerstein system produces noisy output, then the optimised impulse response may fail to accurately represent the true LTI block, depending on the noise level. Figure 5.5 shows the singular values of $\boldsymbol{H}_0$ when the output is corrupted by additive WGN at an SNR of 5 dB. While the dominant singular values still indicate an approximate rank-3 structure, the separation between signal and noise components is less clear, complicating model order selection.

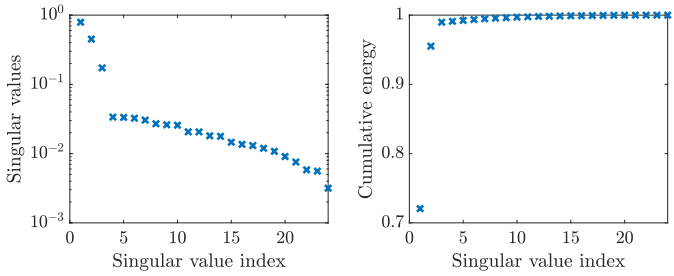


Figure 5.5: SVD of $\boldsymbol{H}_0$ in the presence of noise (SNR = 5 dB). Although a low-rank structure is still present, the noise degrades the separation between significant and negligible modes.

Therefore, regardless of the underlying nonlinearity, noise level, or modelling mismatch, ERA applies SVD to $\boldsymbol{H}_0$ to isolate the dominant system modes:

$$\boldsymbol{H}_0 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \begin{bmatrix}\boldsymbol{U}_1 & \boldsymbol{U}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{\Sigma}_1 & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Sigma}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{V}_1^\top\\ \boldsymbol{V}_2^\top\end{bmatrix} \approx \boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}_1^\top, \tag{5.33}$$

where the truncation retains the first $M_{\mathrm{h}-}$ singular values, consistent with the assumed system order. The truncated matrices $\boldsymbol{U}_1 \in \mathbb{R}^{r\times M_{\mathrm{h}-}}, \boldsymbol{\Sigma}_1 \in \mathbb{R}^{M_{\mathrm{h}-}\times M_{\mathrm{h}-}}, \boldsymbol{V}_1 \in \mathbb{R}^{s\times M_{\mathrm{h}-}}$ are used to define the observability and controllability matrices as:

$$\boldsymbol{\Gamma}_r = \boldsymbol{U}_1\boldsymbol{\Sigma}_1^{1/2} \quad \boldsymbol{\Delta}_s = \boldsymbol{\Sigma}_1^{1/2}\boldsymbol{V}_1^\top. \tag{5.34}$$

The choice of hyperparameters r and s, which define the dimensions of the data matrices $\boldsymbol{H}_0$ and $\boldsymbol{H}_1$, plays a crucial role in the accuracy and robustness of the subspace-based estimation. This selection strategy closely mirrors that used in Chapter 3, Section 3.8, where the structure and aspect ratio of the Hankel matrix were shown to impact the quality of the estimated poles significantly. In particular, constructing a nearly square Hankel matrix, by choosing r such that $\frac{M_{\mathrm{h}+}}{3} \leq r \leq \frac{2M_{\mathrm{h}+}}{3}$, strikes a favourable trade-off between numerical stability and estimation performance. This also facilitates informed truncation of the SVD to determine $M_{\mathrm{h}-}$. The same guideline is adopted in this section to ensure consistent and reliable subspace conditioning across different experiments.

Recovery of State-Space Parameters:
The state-space matrix $\boldsymbol{A}$ is recovered by solving:

$$\boldsymbol{A} = \boldsymbol{\Gamma}_r^{-1}\boldsymbol{H}_1\boldsymbol{\Delta}_s^{-1} = \boldsymbol{\Sigma}_1^{-1/2}\boldsymbol{U}_1^\top\boldsymbol{H}_1\boldsymbol{V}_1\boldsymbol{\Sigma}_1^{-1/2}. \tag{5.35}$$

The input and output vectors $\boldsymbol{b}$ and $\boldsymbol{c}$ are extracted as the first column of $\boldsymbol{\Delta}_s$ and the first row of $\boldsymbol{\Gamma}_r$, respectively. This yields the discrete-time LSS model that approximates the

system characterised by the impulse response $\boldsymbol{\theta}^{\mathrm{L}}$. The ERA offers a systematic and efficient means of estimating the LSS model from data, provided the noise level is low and the impulse response is sufficiently rich to capture the system dynamics. However, in the presence of measurement noise or nonlinear distortions, additional filtering or regularisation may be required to ensure robustness.

To estimate the nonlinear parameters $\boldsymbol{E}$ and $\boldsymbol{f}$, this thesis exploits the equivalence relationship between block-oriented models and the PNLSS model, as established in Paduart et al. (2007). Based on the structure in equation (4.39), each column of the matrix $\boldsymbol{E}$ can be interpreted as scaling the input vector $\boldsymbol{b}$ by the higher-order polynomial coefficients $\theta_2^{\mathrm{NL}}, \ldots, \theta_{D+}^{\mathrm{NL}}$, which are obtained as described in Section 5.2.3. Similarly, the nonlinear output vector $\boldsymbol{f}$ is constructed by scaling the output gain d with the same polynomial coefficients $\theta_2^{\mathrm{NL}}, \ldots, \theta_{D+}^{\mathrm{NL}}$. It is important to note that the optimised parameter θ_1^{NL} is not explicitly used in this realisation step. This is due to an inherent gain ambiguity (or "gain transfer") between the linear and nonlinear blocks, which is unavoidable in block-oriented structures. Specifically, the nonlinear parameter vector $\boldsymbol{\theta}^{\mathrm{NL}}$ may be scaled by a factor $1/\alpha$, while the linear parameter vector $\boldsymbol{\theta}^{\mathrm{L}}$ is simultaneously scaled by α, resulting in an identical input-output behaviour but differing internal representations.

The effect of this scaling ambiguity does not alter the overall system response but may impact the interpretability and comparability of the individual block parameters. For instance, if the static nonlinearity is scaled down, the dynamic block compensates with a proportionally larger gain, making direct comparisons between true and estimated block parameters potentially misleading. This issue becomes especially important in validation studies or when physical interpretability of the blocks is desired. In this thesis, to circumvent this ambiguity during performance evaluation, both the true and estimated parameters are normalised before comparison. This ensures that the gain transfer does not distort the interpretation of identification accuracy. Alternative strategies, such as constraining the nonlinear slope at the origin or enforcing unit DC gain on the linear filter, may further enhance block-level accuracy without prior knowledge. These trade-offs are especially relevant when designing control strategies or interpreting physical subsystems from the identified models.

This transformation from the optimised block-oriented model to the PNLSS form is effectively a model reparametrisation. Under ideal noise-free conditions and sufficient excitation, the input-output behaviour of both models should match. However, unlike models initialised from linear identification techniques such as BLA or N4SID followed by LM optimisation, the PNLSS model derived from the optimised block-oriented structure offers the potential for improved accuracy without necessarily increasing the number of parameters, discussed in Section 5.3. In practical scenarios, this approach may involve higher training time, primarily due to the additional effort needed to ensure stability throughout the identification process. Importantly, the stability of

the final PNLSS model depends on the stability of the intermediate block-oriented model, especially the linear dynamic component. If this is not inherently stable, prior knowledge such as known filter shapes or spectral properties may be required to enforce stability. While this introduces some added complexity and demands domain insight, it is essential for obtaining a reliable and physically consistent model.

Stable Realisation via Regularised Impulse Response:
The shape-regularised optimisation in (5.25) ensures that the estimated linear impulse response $\boldsymbol{\theta}^{\mathrm{L}}$ decays smoothly. This implies that the underlying system is bounded-input bounded-output (BIBO)-stable. When such a response is used as input to the ERA algorithm, the dominant stable modes are preserved, and the realised state-space system inherits this stability. Empirically, the eigenvalues of the resulting matrix $\boldsymbol{A}$ are observed to lie within the unit circle. Theoretically, this is justified by the decay of $\boldsymbol{\theta}^{\mathrm{L}}$ and the structure-preserving nature of ERA.

Proposition 1 (Effect of regularisation). The regularised cost function (5.25)

$$V(\boldsymbol{\theta}) = \underbrace{\sum_k \left(\Phi(\boldsymbol{u}_k, \boldsymbol{\theta}) - y_k\right)^2}_{\text{data fit}} + \underbrace{\mu \left\|\boldsymbol{w} \odot \boldsymbol{\theta}^{\mathrm{L}}\right\|_2^2}_{\text{decay}} + \underbrace{\lambda \left\|\boldsymbol{\Delta}^2 \boldsymbol{\theta}^{\mathrm{L}}\right\|_2^2}_{\text{smoothness}},$$

with $\mu, \lambda > 0, \boldsymbol{w} = [1, \ldots, M_{\mathrm{h}+}]^\top$ and $\boldsymbol{\Delta}^2$ the second-difference operator (refer to (5.20)), promotes impulse responses that are both decaying and smooth.

The decay term $\mu\|\boldsymbol{w} \odot \boldsymbol{\theta}^{\mathrm{L}}\|_2^2 = \mu \sum_i i^2 \theta_i^2$ penalises large coefficients more strongly for increasing index i, thus suppressing non-decaying tails and encouraging a magnitude decrease with time. The curvature term $\lambda\|\boldsymbol{\Delta}^2 \boldsymbol{\theta}^{\mathrm{L}}\|_2^2 = \lambda \sum_i (\theta_{i+2} - 2\theta_{i+1} + \theta_i)^2$ penalises rapid local changes, thereby enforcing smooth transitions between consecutive coefficients. Together, these two penalties promote an estimated impulse response $\boldsymbol{\theta}^{\mathrm{L}}$ that is both attenuating and smooth along its support.

Corollary 1 (Decay implies BIBO stability). As a consequence of Proposition 1, the decaying impulse response corresponds to a discrete-time LTI system that is BIBO-stable.

ERA does not inherently guarantee stability unless the input impulse response corresponds to a stable system, and the rank truncation used in SVD (5.33) does not introduce unstable dynamics.

Proposition 2 (Structure-preserving Stability of ERA). As a consequence of Proposition 1 and Corollary 1, any state-space realisation $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}\}$ obtained from

$\boldsymbol{\theta}^{\mathrm{L}}$ via ERA results in a stable LTI system with the eigenvalues of the state transition matrix $\boldsymbol{A}$ satisfying:

$$|\lambda_i(\boldsymbol{A})| < 1, \quad \forall i.$$

Proof. Recall that the Hankel matrix $\boldsymbol{H}_0$ and its time-shifted version $\boldsymbol{H}_1$ are constructed from the Markov parameters (samples of the impulse response) as shown in (5.28)–(5.32). By Proposition 1, the entries of these matrices decay as one moves away from the top-left corner. As a result, the Hankel matrices are increasingly low-rank and dominated by the slowest decaying (most significant) modes.

The truncated SVD of $\boldsymbol{H}_0$ in (5.33) retains only the $M_{\mathrm{h}-}$ largest singular values and associated singular vectors, which correspond to the dominant stable dynamics. The state matrix $\boldsymbol{A}$ is then recovered from the low-rank approximation of $\boldsymbol{H}_0$ and $\boldsymbol{H}_1$ via a similarity transformation, as given in (5.35). Because the impulse response is decaying, no unstable modes are encoded in the Hankel matrices. Thus, the SVD truncation preserves only stable dynamics. Consequently, the realised $\boldsymbol{A}$ matrix also has eigenvalues strictly inside the unit circle.

This stability property is further supported by classical system identification results (Juang and Pappa, 1985), which state that ERA applied to the impulse response of a stable system yields a stable state-space realisation, provided the dominant singular values are retained.

Hence, the condition $|\lambda_i(\boldsymbol{A})| < 1$ follows directly from the decay in $\boldsymbol{\theta}^{\mathrm{L}}$ and the structure-preserving nature of ERA. □

The sequence of steps involved in the nonlinear system identification of the Hammerstein system discussed in this chapter is summarised in Algorithm 3.

5.3 Results and Evaluation - Hammerstein System

This section presents a comprehensive evaluation of the proposed identification framework across simulated and measured datasets of the Hammerstein model structure. For all cases, the models are evaluated under different noise conditions using separate training, validation, and test datasets. The primary performance metric is the RRMSE in percentage, computed between the true output (noiseless, if applicable) and predicted

Algorithm 3 From Volterra kernel to state-space model: a *prior-informed*, structured identification of a Hammerstein system

Input: Input-output data $\{u_k, y_k\}_{k=1}^{N}$
Output: PNLSS Parameters $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{E}, \boldsymbol{f}\}$

1: Choose Volterra memory M_h and order D (ensure $N \geq D \cdot I$, $I = M_\mathrm{h} + 2$)
2: Build full design matrix $\boldsymbol{U} \in \mathbb{R}^{N \times I^D}$ and apply prior (5.12) to obtain reduced $\tilde{\boldsymbol{U}} \in \mathbb{R}^{N \times D(M_\mathrm{h}+1)}$
3: Estimate reduced Volterra kernel $\tilde{\boldsymbol{h}} \in \mathbb{R}^{D(M_\mathrm{h}+1)}$ (see (5.17)) and regularise via (5.19) to obtain $\tilde{\boldsymbol{h}}_\mathrm{reg}$
4: Reshape $\tilde{\boldsymbol{h}}_\mathrm{reg}$ into $\boldsymbol{R} \in \mathbb{R}^{(M_\mathrm{h}+1) \times D}$ (cf. (5.21)) and extract block parameter estimates $\{\hat{\boldsymbol{t}}, p_1, \ldots, p_D\}$ by SVD (Section 5.2.2)
5: Optionally regularise/extend ($M_{\mathrm{h}+} > M_\mathrm{h}$, $D_+ > D$) using (5.23)–(5.25) to get an extended impulse response
6: Record the feedback coefficient d from the first coefficient of impulse response
7: Form Hankel matrices $\boldsymbol{H}_0, \boldsymbol{H}_1$ (size $r \times s$, $r+s = M_{\mathrm{h}+}$, choose r, s with $r, s \geq M_{\mathrm{h}-}$) with the remaining impulse response vector. Select model order $n = M_{\mathrm{h}-}$ from singular-energy criterion (90–95%)
8: Perform SVD on $\boldsymbol{H}_0$, compute $\boldsymbol{\Gamma}_r, \boldsymbol{\Delta}_s$ (cf. (5.34)) and recover linear state-space estimates $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}\}$ via ERA formulas ((5.35))
9: Estimate nonlinear maps $\{\boldsymbol{E}, \boldsymbol{f}\}$ from $\boldsymbol{b}, d$ and polynomial coefficients $\theta_2^\mathrm{NL}, \ldots, \theta_{D_+}^\mathrm{NL}$
10: Return $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{E}, \boldsymbol{f}\}$

output on the test set. The mean and standard deviation of the output prediction error are also reported to assess the bias and variability of the models. Additionally, for the simulation dataset, we report the RRMSE between the normalised true parameters and normalised estimated parameters to assess block-level accuracy, especially for the proposed approach.

Across all experiments, we adopt a standardised naming convention for the models of the form `finalModel_intermediateModel(s)_optimiser`. Results directly related to the contributions of this thesis are highlighted with a grey background in Tables 5.2-7.4. Baseline comparisons include the classical nonlinear state-space modelling approach, where PNLSS parameters are initialised from linear estimates (e.g., BLA, N4SID) with the nonlinear mappings initially set to zero and subsequently optimised. All experiments were performed on a machine with an Intel Core i7-1165G7 CPU (2.80GHz, 11th Gen, 8 logical cores) and 16GB RAM.

5.3.1 Numerical Simulations

This section presents results from numerical simulations based on Hammerstein systems with third-order low-pass linear blocks and three distinct nonlinearities: polynomial, dead zone, and sigmoid. These serve as controlled experiments for validating the individual stages of the proposed modelling framework.

5.3.1.1 Simulation Setup

The LTI block is designed as a third-order low-pass filter with a passband frequency of 0.5 Hz, a passband ripple of 0.5 dB, and a stopband attenuation of 20 dB. Three different nonlinearities (see Figure 5.6) are considered:

1. Third-order polynomial:

$$q_k = v_k + 0.4v_k^2 + 0.2v_k^3. \tag{5.36}$$

2. Dead zone, defined for the range $[-0.3, 0.3]$:

$$q_k = \begin{cases} 0, & \text{for } -0.3 \leq v_k \leq 0.3 \\ v_k - 0.3, & \text{for } v_k > 0.3 \\ v_k + 0.3, & \text{for } v_k < -0.3. \end{cases} \tag{5.37}$$

   This specific width was chosen based on an energy ratio analysis. The energy ratio is defined as the ratio of the RMS value of the output to that of the input of the static nonlinearity, normalised by the gain. This ratio serves as an indicator of nonlinearity intensity: a value close to 1 implies nearly linear behaviour, while a lower value reflects stronger nonlinearity due to greater signal attenuation in the dead zone. In the gear backlash modelling scenario discussed in Ovy and Sun (2018), the energy ratio was approximately 0.92, indicating moderate nonlinearity. In our simulation with a dead zone of width 0.6 (i.e., the range $[-0.3, 0.3]$), the energy ratio was computed to be 0.78, indicating a stronger nonlinearity than in the backlash case. This quantitative comparison supports the chosen dead zone width, demonstrating that the simulated nonlinearity not only reflects realistic nonlinear behaviour observed in practical systems but also explores a more challenging identification scenario.

3. Sigmoid function:

$$q_k = \frac{2}{1 + e^{-0.8v_k}} - 1. \tag{5.38}$$

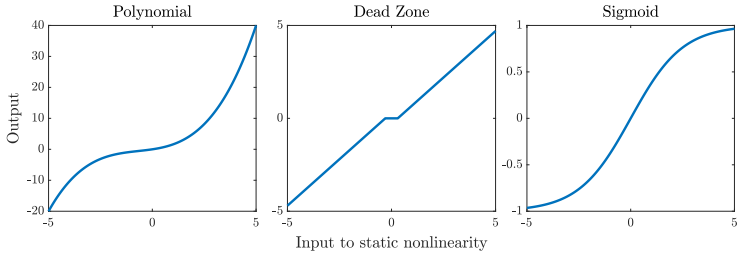


Figure 5.6: Illustration of Static Nonlinearities used in Simulation: 1. Third-order polynomial, 2. Dead zone defined for the range $[-0.3, 0.3]$, 3. Scaled and symmetric logistic sigmoid. These nonlinear functions are approximated by D^{th}-order polynomial functions in the simulations.

In this study, we assume that the input signals are noise-free, and only output measurements are affected by additive WGN. This modelling choice aligns with common practice in system identification, particularly in control-oriented settings where the input is typically a known, designed excitation (e.g., multisine or pseudorandom binary sequence), or a digital control signal, and thus can be assumed noise-free. This is often valid when the input originates from a digital controller, making it directly known and unaffected by measurement noise.

However, in identification scenarios where the goal is to extract physically meaningful parameters from naturally occurring data, the input cannot always be imposed precisely and is often subject to measurement noise. While input noise can be incorporated into certain low-order nonlinear models using specialised estimation techniques, such extensions are significantly more complex and fall outside the scope of this thesis, which focuses on general block-oriented nonlinear systems.The RRMSE between the true output $\boldsymbol{y}$ and the predicted output $\hat{\boldsymbol{y}}$ across $N_{\text{runs}} = 40$ Monte-Carlo trials is averaged as below:

$$\text{RRMSE}_1 = \frac{100}{N_{\text{runs}}} \sum_{i=1}^{N_{\text{runs}}} \left(\frac{\sqrt{\frac{1}{N-M} \sum_{k=M+1}^{N} (y_k - \hat{y}_{k,i})^2}}{\sqrt{\frac{1}{N-M} \sum_{k=M+1}^{N} y_k^2}} \right) \% \tag{5.39}$$

and compared in Table 5.2 along with the averaged training time. The parameter M equals M_{h} for the *prior-informed* Volterra model and initial block-oriented model, $M_{\text{h+}}$ for the optimised block-oriented model, $M_{\text{h-}}$ for the PNLSS models. The choice of 40 trials strikes a practical balance between statistical reliability and computational cost, as each trial involves full system identification under varying noise conditions across multiple algorithms. Increasing N_{runs} beyond 40 yields only marginal benefits, while increasing computational load.

The *prior-informed* Volterra model enables the identification of true parameters of the LTI block and the static nonlinear block (in the case of polynomial nonlinearity) as explained in Section 5.2.2. These parameters are further improved in the optimisation routine. The linear filters are designed as infinite impulse response (IIR) systems, but using the Volterra approach and LM optimisation of the block-structured model, we estimate the impulse response $\hat{\boldsymbol{t}}$. These estimates represent an approximate truncation of the IIR response. Later, in the ERA step (see Section 5.2.4), the truncated impulse response of order $M_{\mathrm{h}+}$ is used to construct a reduced order linear state-space model of order $M_{\mathrm{h}-}$, which provides a good approximation of the true system dynamics.

The comparison is performed in the frequency domain to compare the actual and estimated parameters of the linear filters. The Frequency Response Function (FRF) effectively captures the filter's behaviour in the frequency domain and are less sensitive to truncation errors. The impulse response estimate $\hat{\boldsymbol{t}}$ is first normalised using the L^2-norm. Normalisation is crucial in the block-oriented model structure because the scaling factor can be arbitrarily moved between the blocks. While this scaling does not affect the error in the predicted output, normalisation is recommended to ensure a consistent and meaningful comparison of the true filter characteristics. After normalisation, the impulse response estimates are converted into FRFs using MATLAB's `freqz` function, with the number of frequency points set to $K = 512$. The RRMSE between the FRFs of the true filter $F(\omega)$ and the estimated filter $\hat{F}(\omega)$ for K frequency points is computed as below:

$$\mathrm{RRMSE}_2 = \frac{100}{N_{\mathrm{runs}}} \sum_{i=1}^{N_{\mathrm{runs}}} \left(\frac{\sqrt{\frac{1}{K} \sum_{n=1}^{K} \left| F(\omega_n) - \hat{F}(\omega_n)_i \right|^2}}{\sqrt{\frac{1}{k} \sum_{n=1}^{K} \left| F(\omega_n) \right|^2}} \right) \%. \tag{5.40}$$

Similarly, the RRMSE between the true coefficients $p_1, \ldots, p_D$ of the static nonlinear block and the estimated coefficients $\hat{p}_1, \ldots, \hat{p}_D$, in the case of the polynomial static nonlinear block, is computed across all Monte-Carlo trials as below:

$$\mathrm{RRMSE}_3 = \frac{100}{N_{\mathrm{runs}}} \sum_{i=1}^{N_{\mathrm{runs}}} \left(\frac{\sqrt{\frac{1}{D} \sum_{j=1}^{D} (\hat{p}_{j,i} - p_j)^2}}{\sqrt{\frac{1}{D} \sum_{j=1}^{D} p_j^2}} \right) \%. \tag{5.41}$$

Additionally, to evaluate the prediction accuracy and robustness of the model across multiple Monte Carlo trials, the error signal was computed as the difference between the true and predicted outputs. For each trial, the mean and standard deviation of the error were computed. To summarise the performance over all N_{runs} trials, the mean of

the means and the mean of the standard deviations were computed as:

$$\begin{aligned}\bar{\mu} &= \frac{1}{N_{\text{runs}}}\sum_{i=1}^{N_{\text{runs}}}\left(\frac{1}{N-M}\sum_{k=M+1}^{N}\left(y_k-\hat{y}_{k,i}\right)\right),\\ \bar{\sigma} &= \frac{1}{N_{\text{runs}}}\sum_{i=1}^{N_{\text{runs}}}\left(\sqrt{\frac{1}{N-M-1}\sum_{k=M+1}^{N}\left(\left(y_k-\hat{y}_{k,i}\right)-\mu_i\right)^2}\right).\end{aligned} \tag{5.42}$$

where μ_i is the mean error of the ith trial. Here, $\bar{\mu}$ quantifies the overall bias in the predictions across trials, while $\bar{\sigma}$ reflects the average variability in the prediction errors. The Root Mean Square Error (RMSE) can be approximately estimated from the averages of the mean and standard deviation of the prediction error as follows:

$$\text{RMSE} \approx \sqrt{\bar{\mu}^2+\bar{\sigma}^2}. \tag{5.43}$$

Intuitively, the RRMSE_1 is obtained by normalising the RMSE by the RMS value of the true output signal $\boldsymbol{y}$:

$$\text{RRMSE}_1 \approx 100\times\frac{\text{RMSE}}{\text{RMS}(\boldsymbol{y})}\%. \tag{5.44}$$

5.3.1.2 Results and Analysis

The input signal $\boldsymbol{v} \sim \mathcal{N}(0,1)$ is passed through the simulated Hammerstein system, and the output $\boldsymbol{y}$ is contaminated with WGN at varying SNR levels. A total of $N = 8000$ samples were used for training, with 2000 unseen samples allocated to each of the validation and test sets, as shown in Figure 5.7. The input excites all frequencies, while the output, taken after the low-pass LTI block, has dominant low-frequency components.

Table 5.2 reports the identification performance for three types of nonlinearities (polynomial, dead zone, and sigmoid), under different SNR conditions. The table includes output prediction errors (RRMSE_1), and relative parameter errors in the linear (RRMSE_2) and nonlinear (RRMSE_3) blocks, over 40 Monte Carlo runs. For convenience and readability, we assign short labels to each model, as defined in Table 5.1. These labels will be used consistently throughout the thesis. Model IDs P1-P4 correspond to the models proposed in this work, while B1-B4 represent baseline models. In particular, B2 and B4 denote PNLSS variants that are compared against the proposed PNLSS model P4.

In Table 5.2.A, the model B4 demonstrates accurate output approximation, especially under low noise conditions. At SNR = 20 dB, it achieves a notably low RRMSE of

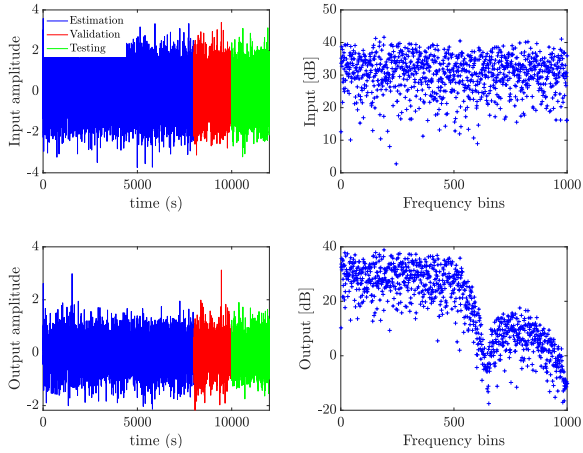


Figure 5.7: Simulation dataset showing the training, validation, and test splits. The Gaussian input excites all frequencies, and the low-pass nature of the LTI block results in an output with greater low-frequency content.

Table 5.1: List of models used throughout the thesis

Model ID	Description
P1	*Prior-informed* Volterra
P2	BS_Volterra
P3	BS_Volterra_LM
P4	PNLSS_BS_Volterra_LM
B1	BLA
B2	PNLSS_BLA_LM
B3	N4SID
B4	PNLSS_N4SID_LM

0.56%. The model B2, which is initialised using the BLA, converges to the same optimal solution as model B4 in the noise-free case. However, as noise increases, the sensitivity of the BLA-based initialisation becomes apparent, and B2 fails to attain the optimal solution, unlike models B4 and P4.

It is also worth mentioning that the dimensioning parameter r used to estimate the BLA in step 1 of Algorithm 1 (see (2.47)) is set to 10 times the model order (i.e., $r = 30$ for model order 3). This choice, while improving the fidelity of the BLa, results in higher computational cost, reflected in B2's longer training time, especially when followed by LM optimisation.

Table 5.2: Performance on Simulated Test Dataset with 3 Types of Nonlinearity - Hammerstein System

Methodology Details	M	D	No. of Parameters	RRMSE_1 (%)			RRMSE_2 in the LTI block (%)			RRMSE_3 in the SNL block (%)			$\bar{\mu}$			$\bar{\sigma}$			Training time (s)
				No noise	SNR 20 dB	SNR 5 dB	No noise	SNR 20 dB	SNR 5 dB	No noise	SNR 20 dB	SNR 5 dB	No noise	SNR 20 dB	SNR 5 dB	No noise	SNR 20 dB	SNR 5 dB	
A. Polynomial nonlinear block - Training sample: $N = 8000$, Test sample: 2000, $\mathrm{RMS}(y_{\mathrm{test}}) = 1.22$, $N_{\mathrm{runs}} = 40$																			
Prior-informed Volterra	20	3	63	1.05	1.36	4.98	0.05	0.08	0.37	–	–	–	-3e-6	-3e-4	-2e-3	0.01	0.02	0.06	0.03
BS_Volterra	20	3	23	0.07	0.70	4.04	0.05	0.07	0.30	0.02	**0.18**	**1.04**	-1e-4	1e-4	3e-3	9e-4	8e-3	0.05	0.05
BS_Volterra_LM	48	3	52	2e-4	0.33	1.99	**1e-4**	**0.04**	**0.20**	**1e-4**	**0.17**	**0.94**	1e-6	1e-4	1e-3	2e-6	4e-3	0.02	1.8
PNLSS_BS_Volterra_LM	3	3	24	**1.00**	**1.06**	**2.03**	–	–	–	–	–	–	1e-4	2e-4	8e-4	0.01	0.01	0.02	1.8
BLA	3	–	16	52.12	52.21	69.55	–	–	–	–	–	–	0.37	0.38	0.40	0.51	0.51	0.74	148
PNLSS_BLA_LM	3	3	24	**0.39**	8.80	11.25	–	–	–	–	–	–	1e-5	6e-3	5e-3	5e-3	0.11	0.14	163
N4SID	3	–	16	48.73	48.74	48.78	–	–	–	–	–	–	0.38	0.38	0.38	0.46	0.46	0.46	0.3
PNLSS_N4SID_LM	3	3	24	**0.39**	**0.56**	**2.32**	–	–	–	–	–	–	2e-5	-7e-5	3e-4	5e-3	7e-3	0.03	11
B. Dead zone nonlinear block - Training sample: $N = 8000$, Test sample: 2000, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.53$, $N_{\mathrm{runs}} = 40$																			
Prior-informed Volterra	20	3	63	13.65	13.68	14.43	**0.07**	**0.08**	**0.25**	–	–	–	3e-3	3e-3	3e-3	0.07	0.07	0.08	0.02
BS_Volterra	20	3	23	13.61	13.63	14.11	**0.07**	**0.08**	**0.24**	–	–	–	4e-3	4e-3	5e-3	0.07	0.07	0.07	0.05
BS_Volterra_LM	48	3	52	13.48	13.49	13.66	**0.05**	**0.06**	**0.20**	–	–	–	5e-3	5e-3	5e-3	0.07	0.07	0.07	1.6
PNLSS_BS_Volterra_LM	3	3	24	**13.76**	**13.76**	**13.87**	–	–	–	–	–	–	4e-3	4e-3	4e-3	0.07	0.07	0.07	1.6
BLA	3	–	16	18.51	18.60	32.68	–	–	–	–	–	–	2e-3	2e-3	3e-3	0.10	0.10	0.17	84
PNLSS_BLA_LM	3	3	24	**13.63**	**13.64**	18.38	–	–	–	–	–	–	4e-3	4e-3	4e-3	0.07	0.07	0.10	94
N4SID	3	–	16	18.55	18.55	18.63	–	–	–	–	–	–	2e-3	2e-3	2e-3	0.10	0.10	0.10	0.2
PNLSS_N4SID_LM	3	3	24	**13.63**	**13.64**	**13.82**	–	–	–	–	–	–	4e-3	4e-3	4e-3	0.07	0.07	0.07	10
C. Sigmoid function nonlinear block and Training sample, $N = 8000$, Test sample: 2000, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.25$, $N_{\mathrm{runs}} = 40$																			
Prior-informed Volterra	20	3	63	1.65	1.83	4.78	**0.05**	**0.06**	**0.19**	–	–	–	-1e-4	-2e-4	-4e-4	4e-3	5e-3	0.01	0.02
BS_Volterra	20	3	23	1.25	1.32	2.82	**0.05**	**0.06**	**0.19**	–	–	–	-2e-4	-2e-4	6e-5	3e-3	3e-3	7e-3	0.05
BS_Volterra_LM	48	3	52	1.25	1.28	2.27	**5e-3**	**0.04**	**0.20**	–	–	–	-2e-4	-2e-4	-1e-4	3e-3	3e-3	6e-3	1.5
PNLSS_BS_Volterra_LM	3	3	24	**1.36**	**1.39**	**2.15**	–	–	–	–	–	–	-1e-4	-1e-4	-1e-4	3e-3	4e-3	5e-3	1.5
BLA	3	–	16	8.15	8.26	23.71	–	–	–	–	–	–	-4e-4	-4e-4	-3e-4	0.02	0.02	0.06	61
PNLSS_BLA_LM	3	3	24	**1.58**	**1.63**	7.72	–	–	–	–	–	–	-2e-4	-2e-4	-3e-4	4e-3	4e-3	0.02	69
N4SID	3	–	16	8.13	8.13	8.29	–	–	–	–	–	–	-4e-4	-4e-4	-4e-4	0.02	0.02	0.02	0.2
PNLSS_N4SID_LM	3	3	24	**1.59**	**1.64**	**2.71**	–	–	–	–	–	–	-2e-4	-2e-4	-3e-4	4e-4	4e-3	7e-3	8

In contrast, model P1, despite using a higher number of parameters, is significantly faster to compute while achieving a comparable level of accuracy, particularly in low-noise conditions. The block-structured (BS) model P2, identified from the rank-one approximation of *prior-informed* Volterra kernel, reduces the number of parameters significantly and allows for the separate identification of the LTI dynamic block and SNL block parameters. These initial estimates from the *prior-informed* Volterra model are further improved through LM optimisation, with increased flexibility to fit a longer impulse response. The resulting model P3 achieves the lowest RRMSE in the noise-free condition and maintains relatively low errors at higher noise levels, indicating robust performance. This is further supported by consistently low relative parameter errors in both blocks across noise conditions, demonstrating its capability to model and estimate the system's internal structure and individual block dynamics.

The PNLSS model P4, realised from the optimised impulse response and nonlinearity mappings, achieves output prediction accuracy equivalent to that of the best-performing baselines across all noise conditions. Notably, this model is obtained in under 2 seconds, despite using the same number of parameters. In contrast, the baseline approaches B2 and B4 require significantly more time, primarily due to the computational burden of optimising in the full parameter space of the state-space model.

The averaged mean error $\bar{\mu}$ and standard deviation $\bar{\sigma}$ across 40 Monte-Carlo runs (as shown in Table 5.2.A) provide insight into the bias and variability of each model. Using the relations in (5.43) and (5.44), along with the values of $\bar{\mu}$, $\bar{\sigma}$ and $\mathrm{RMS}(y_{\mathrm{test}})$ provided in the table, the RRMSE_1 can be approximated analytically.

Similar observations hold for Tables 5.2.B and 5.2.C, which correspond to systems with dead zones and sigmoid nonlinearities, respectively. Across these cases, the PNLSS model P4 consistently achieves accuracy on par with the best baselines (B2 and B4), while requiring significantly less computation time, again under 2 seconds per run.

The comparable performance of all nonlinear models in Table 5.2.B can be explained by the nonlinearity induced by the dead zone. The energy ratio between the output and input of the SNL block is 0.78, indicating that only 22% of the input signal energy is suppressed by the nonlinearity. This implies that the system operates largely within a linear regime. As a result, all nonlinear models converge to a similar RRMSE of around 13%, while the linear models plateau at a higher error of 18%. The similarity in both mean and variance of the error across methods suggests that the small nonlinearity does not pose a significant modelling challenge, and that all models can capture it equally well. In such settings, differences in computational efficiency, structural transparency, or parameter robustness become more important for model selection than marginal improvements in accuracy.

Figure 5.8 summarises the output prediction accuracy across all nonlinearities at SNR 5 dB. The bar graph shows the mean of the RRMSEs across Monte-Carlo trials, while

the accompanying violin plots illustrate the full distribution (99th percentile) of RRMSE values for each nonlinear model (for clarity, the linear models B1 and B3 are omitted in the plots). To prevent misleading tails caused by rare outliers, the violin plots are clipped at the 99th percentile. The width of the violin indicates the density of error values, while the horizontal black line denotes the median. Together, these plots highlight both the average performance and the variability across runs, allowing a more complete comparison between methods.

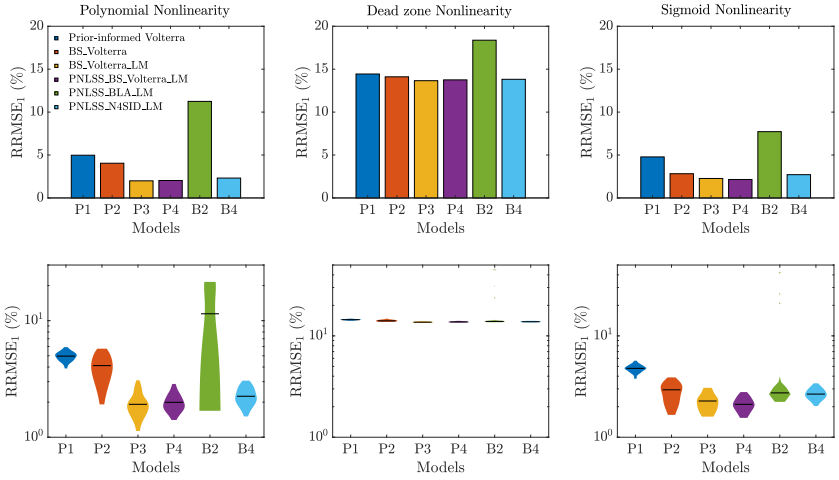


Figure 5.8: RRMSE$_1$ (%) of output prediction for all nonlinearities at SNR 5 dB. The proposed PNLSS model P4 achieves accuracy comparable to baseline B4.

Analysis of Variance (ANOVA) test: To determine whether the three identification methods (P4, B2, B4) produce statistically different mean RRMSE under various nonlinearities and SNR levels, a one-way ANOVA test is performed. Each method yields 40 RRMSE values corresponding to different noise realisations. The ANOVA evaluates the null hypothesis:

$$H_0 : \mu_{\mathrm{P2}} = \mu_{\mathrm{B2}} = \mu_{\mathrm{B4}},$$

against the alternative:

$$H_1 : \text{At least one mean differs.}$$

The ANOVA partitions the total variation of the data into two components as shown in Table 5.3. Here k is the number of groups (3 methods) and N is the total number of observations (120). If all group means are equal, the ratio F should be close to 1. A large F indicates that the between-group variance is much larger than the within-group variance, suggesting that not all means are equal.

Table 5.3: General structure of ANOVA table

Source	**Description**	**Sum of Squares (SS)**	**Degrees of Freedom (df)**	**Mean Square (MS)**	**F statistic**
Groups	Variation between the group means	$\mathrm{SS}_{\mathrm{groups}}$	$k-1$	$\mathrm{MS}_{\mathrm{groups}} = \frac{\mathrm{SS}_{\mathrm{groups}}}{k-1}$	$\mathrm{F} = \frac{\mathrm{MS}_{\mathrm{groups}}}{\mathrm{MS}_{\mathrm{error}}}$
Error	Variation within each group	$\mathrm{SS}_{\mathrm{error}}$	$N-k$	$\mathrm{MS}_{\mathrm{error}} = \frac{\mathrm{SS}_{\mathrm{error}}}{N-k}$	–
Total	Total variation in all samples	$\mathrm{SS}_{\mathrm{total}}$	$N-1$	–	–

The p-value represents the probability of observing an F-statistic as large as (or larger than) the one obtained, assuming H_0 is true:

$$p = P(F_{(k-1,N-k)} \geq F_{\mathrm{obs}}).$$

A small p-value indicates that such a large F would be extremely unlikely if all group means were equal. The significance level of p-value and its interpretation are given in Table 5.4. A complementary measure to the p-value is the effect size, which quantifies

Table 5.4: Decision rule (common thresholds) for p-values

Threshold	Criterion	Interpretation
$p \geq 0.05$	Statistically insignificant (–)	–
$p < 0.05$	Statistically significant (*)	Evidence that at least one mean differs
$p < 0.01$	Highly significant (**)	Strong evidence
$p < 0.001$	Very highly significant (***)	Very strong evidence

the magnitude of the difference between groups rather than just its statistical significance. In this study, the eta-squared (η^2) metric is used to assess the extent to which the choice of identification method influences the observed variation in RRMSE. It is defined as the ratio of the between-group sum of squares to the total sum of squares,

$$\eta^2 = \frac{\mathrm{SS}_{\mathrm{groups}}}{\mathrm{SS}_{\mathrm{total}}}.$$

The value of η^2 represents the proportion of the total variance in the data that can be attributed to the grouping factor (here, the identification method). Higher values of η^2 indicate a stronger effect of the method on model performance. According to commonly accepted interpretation guidelines, the corresponding threshold values for η^2 are summarised in Table 5.5. Hence, while the p-value tests whether the mean differences are statistically significant, η^2 complements it by describing how practically meaningful those differences are.

Table 5.5: Effect size interpretation for η^2

Range	Interpretation
$\eta^2 < 0.01$	Negligible
$0.01 \leq \eta^2 < 0.06$	Weak effect
$0.06 \leq \eta^2 < 0.14$	Moderate effect
$\eta^2 \geq 0.14$	Strong effect

Table 5.6 summarises the one-way ANOVA results of Hammerstein systems comparing the three PNLSS models (P4, B2, B4) across different nonlinearities and SNR levels. For each nonlinearity-SNR condition, the reported p-values indicate whether the differences in model performance (RRMSE) are statistically significant. At the same time, η^2 quantifies the proportion of variance explained by the model type, providing a measure of effect size.

Table 5.6: ANOVA Table - Hammerstein simulation dataset

Nonlinearity	p-value		Star rating		η^2		Effect size interpretation	
	SNR 20 dB	SNR 5 dB	SNR 20 dB	SNR 5 dB	SNR 20 dB	SNR 5 dB	SNR 20 dB	SNR 5 dB
Polynomial	$1e^{-4}$	$1e^{-4}$	***	***	0.32	0.40	Strong	Strong
Dead zone	0.45	$1e^{-4}$	-	***	0.01	0.11	Negligible	Moderate
Sigmoid	$1e^{-4}$	$1.2e^{-3}$	***	**	0.90	0.10	Strong	Moderate

For the polynomial nonlinearity under 20 dB SNR, the ANOVA yielded $p = 1e^{-4}$, which is highly significant ($p < 0.001$). This indicates that the mean RRMSE values across the three identification methods are statistically different, and the null hypothesis H_0 can be rejected. The effect size, $\eta^2 = 0.32$, corresponds to a strong effect, implying that approximately 32% of the total variation in RRMSE can be attributed to the method used. The corresponding notched box plot in Figure 5.9 visually illustrates that the proposed method (P4) achieves a consistently lower RRMSE distribution compared to the baseline methods (B2 and B4). The corresponding post-hoc multiple comparison plot in Figure 5.10 further confirms that the mean of P4 differs significantly from B2, while the difference between P4 and B4 is smaller but still significant at the 0.001 level. Overall, both the statistical and visual analyses suggest that the proposed method outperforms the baselines for this case.

Across all nonlinearities and SNR conditions, the ANOVA results consistently demonstrate that the choice of identification method has a statistically significant effect on model accuracy, except for the dead-zone case at high SNR (20 dB), where $p = 0.45$ indicates no meaningful difference between the methods.

The effect sizes (η^2) vary depending on the nonlinearity and noise level. For the polynomial and sigmoid nonlinearities, the effect sizes are strong ($\eta^2 \geq 0.3$), showing that the method type explains a substantial portion of the RRMSE variance. In contrast, for the dead-zone nonlinearity, the effect is negligible at 20 dB but becomes moderate at 5 dB, suggesting that differences between methods become more pronounced under higher noise conditions.

Overall, these results indicate that the proposed identification method provides a statistically significant improvement in accuracy for most nonlinearities, particularly under challenging (low-SNR) conditions, while maintaining performance comparable to the best baseline in simpler cases.

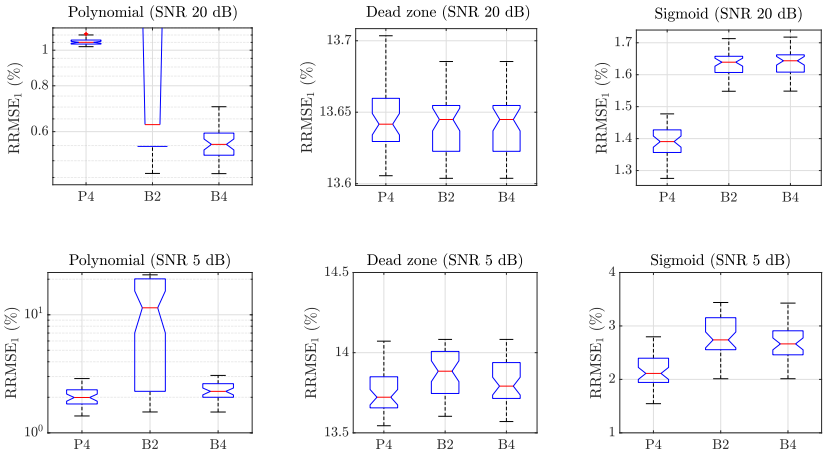


Figure 5.9: Box plots of RRMSE (%) for the three PNLSS models under different nonlinearities and SNR levels for the Hammerstein system. Each subplot corresponds to a specific nonlinearity–SNR combination. The boxes represent the distribution of model errors obtained from 40 Monte Carlo runs, illustrating both the median and spread of the RRMSE values. Lower median and narrower boxes indicate better and more consistent model performance across noise realisations.

Figure 5.11 reports parameter estimation accuracy for both the LTI and static nonlinear (SNL) blocks across the same settings. It also includes true parameter comparison for the polynomial nonlinearity case. As the nonlinearity becomes more complex, estimation accuracy slightly decreases, though the proposed method remains robust in true parameter recovery.

Figure 5.12 compares the training times and number of parameters for different modelling approaches. All PNLSS models shown here use the same number of parameters, ensuring a fair comparison in terms of model complexity. While the

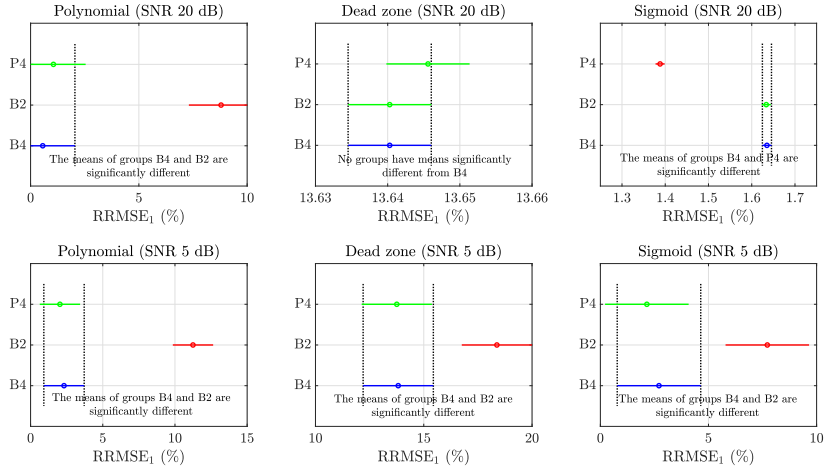


Figure 5.10: Multiple comparison results (Tukey–Kramer test) for pairwise mean differences in RRMSE among the three PNLSS models under different nonlinearities and SNR levels for the Hammerstein system. Each subplot corresponds to a specific nonlinearity–SNR combination. The blue marker represents the fixed reference model (B4), while green markers indicate models whose mean RRMSE lies within the 95% confidence interval of the reference model, implying statistically similar performance. Red markers denote models whose mean differs significantly from the reference, indicating statistically inferior or superior performance relative to B4.

BLA- and N4SID-initialised PNLSS models (B2 and B4) achieve accurate results, they require significantly longer training times due to the computational cost of optimising the nonlinear state-space parameters. This is further exacerbated in the BLA-based model (B2) by the use of a dimensioning parameter set to 10 times the model order. In contrast, the proposed Volterra-based PNLSS model P4 achieves comparable output prediction accuracy using the same number of parameters but completes training in under 2 seconds. This demonstrates the advantage of using a *prior-informed* Volterra structure and block-wise initialisation, resulting in substantially lower training time and rapid convergence.

Detailed plots for a representative case (SNR 5 dB, dead zone nonlinearity) are shown in Figures 5.13–5.14. The pole-zero map of the estimated LTI block across 40 trials is shown on the left of Figure 5.13. The estimated poles and zeros are concentrated around the true locations, although some estimated zeros extend beyond the unit circle, whereas the true zeros lie exactly on it. This behaviour arises from the transmission zero characteristics of the LTI block, which can be sensitive to estimation noise or model structure. While zeros outside the unit circle do not affect system stability

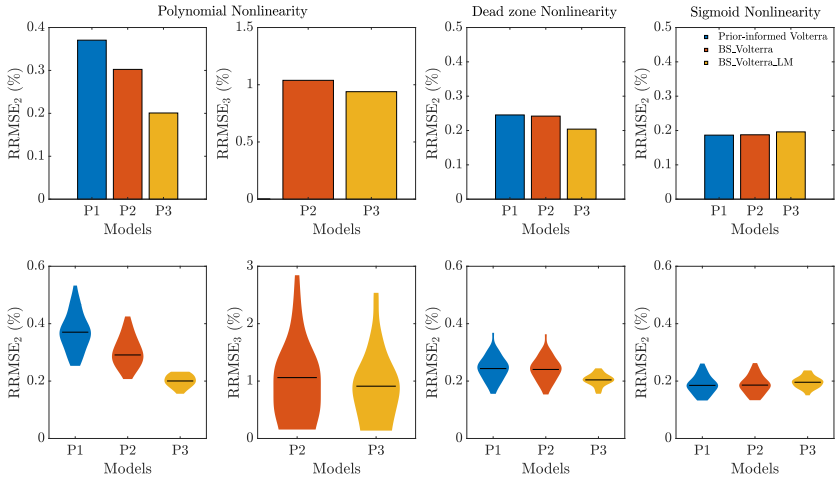


Figure 5.11: $\text{RRMSE}_{2,3}$ (%) of parameters in the LTI and SNL blocks across all nonlinearities at SNR 5 dB. Parameter errors increase slightly with the complexity of nonlinearity. The distributions over 40 trials are captured in the violin plots, clipped at the 99th percentile. The mean error of each distribution corresponds to the height of the associated bar graph.

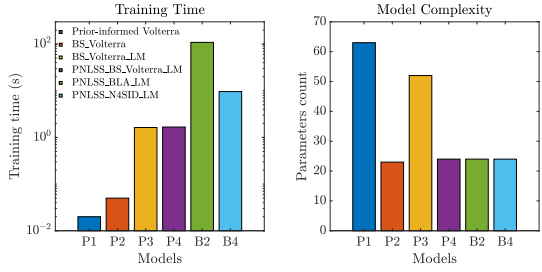
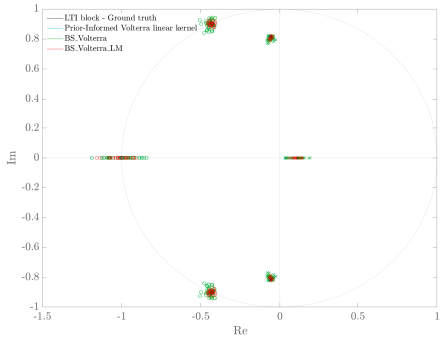
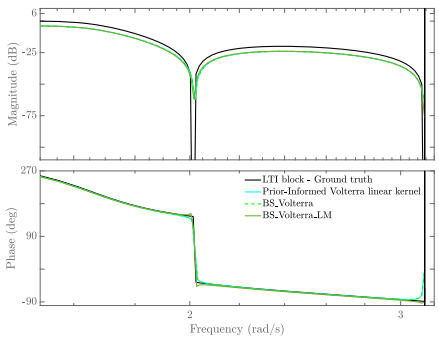


Figure 5.12: Comparison of training times and number of parameters for different PNLSS modelling approaches. All PNLSS models (P4, B2, B4) use the same number of parameters. The proposed PNLSS model P2 achieves equivalent accuracy with significantly faster training (under 2 seconds), demonstrating the advantage of structure-based initialisation.

(unlike poles), they do indicate non-minimum phase behaviour, which can alter the system's phase response and introduce delays or undershoot in the time domain. These discrepancies are important when considering system inversion or control applications, where minimum-phase characteristics are often preferred. On the right, the frequency responses from all 40 Monte Carlo trials are averaged, and the resulting FRF magnitude and phase plots closely match the true system characteristics.



Pole-zero map. Legend: ×: poles, ∘: zeros



Bode plot (Magnitude and Phase)

Figure 5.13: LTI block characteristics for the Hammerstein system with dead zone nonlinearity case (SNR 5 dB), averaged over 40 trials.

The optimisation progress is shown on the left in Figure 5.14, where each line represents the validation error at each iteration across 40 trials. The Volterra-based model P3 starts with a better initial estimate compared to that of the linear models B1 and B3. The optimisation for baseline models B2 and B4 starts with higher validation errors due to zero-initialised nonlinear mappings and requires several iterations to converge. A few BLA-based trials exhibit poor performance, primarily due to the omission of input realisation averaging during BLA estimation (see Section 2.3.5), which is critical for obtaining consistent BLA results. This step was deliberately skipped to reduce the overall simulation cost. On the right, Figure 5.14 shows the output prediction errors on the test set. Since all nonlinear models achieve similar RRMSE values ($\approx 14\%$, Table 5.2.B), the error traces for all approaches lie close to the zero line, visually confirming the similarity in their prediction performance.

5.3.2 Experimental Validation

This section focuses on an experimental evaluation using a nonlinear electronic system with a Hammerstein structure excited by a multisine input. The measurement setup and

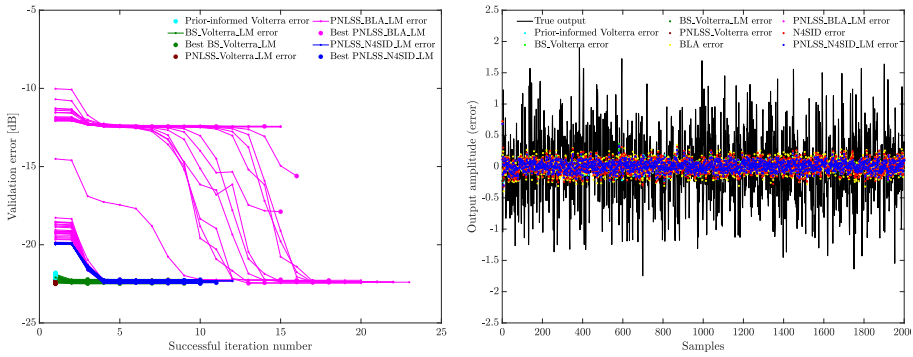


Figure 5.14: Left: Validation errors across iterations for 40 trials (Dead zone nonlinearity, SNR 5 dB). Right: Test set prediction errors. All methods achieve similar performance ($\approx 14\%$ RRMSE).

signal design are described in detail, followed by a comparative analysis of modelling performance.

5.3.2.1 Measurement Setup and Signal Design

We conducted measurements on an electronic nonlinear system described in Vandersteen and Schoukens (1999); Schoukens et al. (2009b), which has a WH structure, as shown in Figure 5.15.

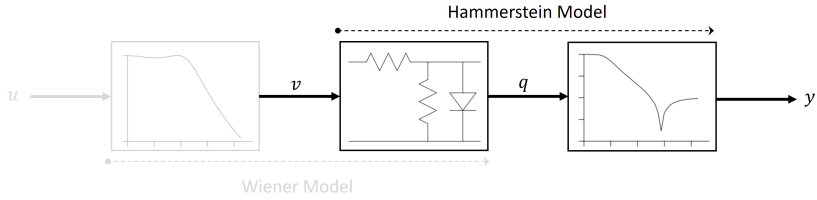


Figure 5.15: Measurement setup of Hammerstein system showing that the input to the SNL block is a coloured signal.

The first LTI block is a third-order Chebyshev (Type I) low-pass filter, designed with a cut-off frequency of 4.4 kHz and a passband ripple of 0.5 dB. This filter passes the low-frequency content of the input signal while attenuating high-frequency components, shaping the excitation to the nonlinear block. The output of this block is used as the input to the Hammerstein system for identification.

The SNL block is implemented using a diode-based circuit shown in Figure 5.16. This circuit consists of two resistors connected in series: $R_1 = 1\text{k}\Omega$ followed by $R_2 = 10\text{k}\Omega$, and a 1N4148 diode connected in shunt configuration. The anode (+) of the diode is connected at the junction between the two resistors, while the cathode (−) is connected to the output node. The input voltage v (input to the SNL) is applied at the left terminal of R_1, and the output is taken across the diode (from the cathode to ground).

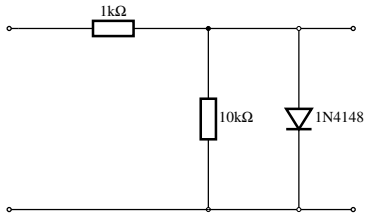


Figure 5.16: Diode circuit used in the SNL block

The resistors form a voltage divider, and the diode introduces a nonlinear switching behaviour. For small input voltages (say $V_{\text{in}} < 0.7$ V): The diode remains reverse-biased and behaves like an open circuit. The output voltage V_{out} is then given by the standard voltage divider relation:

$$V_{\text{out}} \approx \frac{R_2}{R_1 + R_2} \cdot V_{\text{in}} = \frac{10}{11} \cdot V_{\text{in}}.$$

This represents a linear region of operation. However, when the input voltage increases and the voltage at the diode's anode exceeds its cathode by approximately 0.7 V, the diode becomes forward-biased and conducts. This effectively clamps the output voltage near the diode's forward voltage (around 0.7 V), producing a saturating nonlinearity. The circuit thus exhibits piecewise behaviour: linear for small signals, and nonlinear saturation for larger ones. Because the output at any time depends solely on the instantaneous input and not on its past values, the nonlinearity is static and memoryless. The diode provides the nonlinearity, while the resistors control the gain and the activation threshold.

The LTI block following the SNL block is implemented as a third-order inverse Chebyshev (Type II) low-pass filter with a stop-band attenuation of 40 dB starting at 5 kHz. This filter exhibits a sharp roll-off and includes a transmission zero within the frequency range of interest. The presence of a transmission zero introduces a frequency at which the system completely attenuates the input, creating phase and magnitude distortions that are challenging to model or invert accurately. This complicates the identification process, particularly for techniques relying on linear approximations, because it violates the assumption of a smooth, invertible frequency

response. Identifying or inverting the dynamics of such a system requires special care, often involving regularisation or multi-stage modelling strategies.

The same nonlinear system is used for experiments across all the desired block-oriented structures. While the measurement setup was pre-existing, we performed new experiments by generating our excitation signal and acquiring input-output data, as described in Vijayan et al. (2025b). The signal $\boldsymbol{u}$ is generated using a multisine generator with a sampling frequency of 156 kHz and selective excitation of odd frequency bins, where only specific odd frequencies are excited, while others are left unexcited. This excitation signal is applied to the arbitrary signal generators in the measurement setup, and acquisition cards are used to measure the signals $\boldsymbol{v}$ and $\boldsymbol{y}$ for the Hammerstein system. As a result, the input $\boldsymbol{v}$ to the SNL block is already a filtered version of the excitation signal. This is evident in the frequency spectrum of $\boldsymbol{v}$, shown in Figure 5.17.

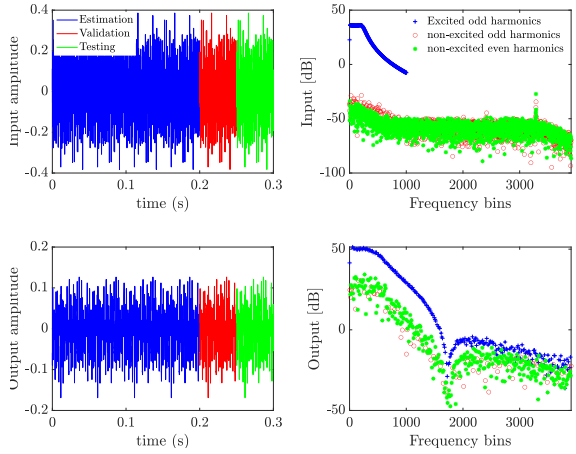


Figure 5.17: Measurement dataset showing the training, validation, and test splits. The multisine input excites selected odd frequency bins, and the third-order inverse Chebyshev filter in the LTI block results in an output with predominantly low-frequency content.

5.3.2.2 Results and Analysis

This section evaluates the identification accuracy and computational performance of various model structures using input-output measurements from a Hammerstein system, as reported in Table 5.7. $N = 31,256$ samples were used for training, with 7,814 unseen samples each for validation and testing. Since the output measurements have very

Table 5.7: Performance on the Measurement Dataset

Hammerstein System (SNR 25 dB, Training sample: $N = 31{,}256$, Test sample: $7{,}814$, $\mathrm{RMS}(y_{\mathrm{test}}) = 52.3$ mV)

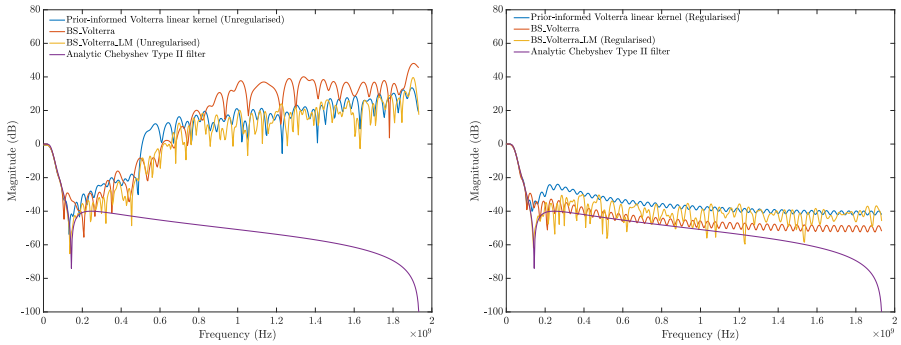
Methodology Details	M	D	No. of Parameters	RRMSE_1 (%)	$\bar{\mu}$ (mV)	$\bar{\sigma}$ (mV)	Training time (s)
Prior-informed Volterra	100	3	303	4.62	0.42	2.37	40
BS_Volterra	100	3	103	7.16	1.62	3.35	40
BS_Volterra_LM	201	3	205	3.96	0.88	1.87	188
PNLSS_BS_Volterra_LM	3	3	24	**11.63**	0.53	6.06	189
BLA	3	–	16	19.21	-3.72	9.33	26
PNLSS_BLA_LM	3	3	24	**3.66**	0.69	1.79	79
N4SID	3	–	16	40.87	-3.39	21.1	1
PNLSS_N4SID_LM	3	3	24	**8.63**	0.69	4.46	67

low noise, we performed the analysis by adding WGN with SNR 25 dB. A third-order polynomial approximates the diode-based static nonlinearity in the setup.

The modelling procedure began with a *prior-informed* Volterra kernel estimation, where knowledge about the underlying block-oriented structure was incorporated into the Volterra series. This prior knowledge enabled the suppression of irrelevant kernel entries. To confirm that the identified system's output remains bounded for any bounded input (BIBO stability), i.e., the sum of the absolute values of all samples of the resulting *prior-informed* Volterra linear kernel must be finite, the *prior-informed* Volterra kernel was regularised (5.19). Based on the knowledge about the LTI block, the weights of the regulariser are chosen, and the initial guess is the unique minimum norm solution obtained from (5.18).

The block-oriented model P2 is then obtained from the resulting regularised *prior-informed* Volterra kernel, the initial guess for the LM optimisation. However, to enforce BIBO stability in all the identified candidate models, it is recommended to include the shape-promoting optimisation step (5.25). The resulting best block-oriented model P3 with shape penalty is used to deduce an equivalent BIBO stable PNLSS model. The normalised FRFs of the final identified filters are shown in Figure 5.18, along with the normalised FRF of an analytically designed inverse Chebyshev filter that replicates the characteristics of the true filter. This comparison clearly demonstrates that the inverse Chebyshev characteristics were successfully enforced during the LM optimisation of the block-oriented model. This validates the effectiveness of the shape penalty in capturing true system behaviour and ensuring that the dynamic block remains physically consistent.

Figure 5.18 also contrasts identification results with and without the use of regularisation. In the left panel, unregularised identification approaches show significant deviation from the analytic Chebyshev reference, particularly in the high-frequency stopband. The identified frequency responses display erratic gain patterns, peaking far above the expected attenuation profile and exhibiting spurious oscillations. These distortions arise due to overfitting of noise and the lack of constraints guiding the solution towards



Unregularised kernel Regularised kernel

Figure 5.18: FRF of an analytic 3rd-order inverse Chebyshev low-pass filter with stop-band attenuation of 40 dB starting at 5 kHz compared with the estimated FRFs of Volterra and Block-structured models at SNR = 25 dB. Refer Sections 5.2.1 and 5.2.3

physically meaningful dynamics. Although the passband is captured to some extent, the models suffer from high variance and instability, especially where the SNR is low.

In contrast, the right panel shows the same identification procedures but with the inclusion of regularisation. The regularised, *prior-informed* Volterra kernel maintains a smooth, attenuated stopband, and the model P3 with shape penalty closely tracks the analytic reference across the full frequency range. This demonstrates that the use of prior-based regularisation reduces variance while preserving low-frequency accuracy, enforcing filter-like decay in the stopband, and avoiding non-physical gain amplification. While the low-pass region alone may appear sufficient for capturing dominant system behaviour, unconstrained models can yield unstable or non-generalisable representations. Therefore, incorporating regularisation and shape constraints is critical not only for high-frequency fidelity but also for numerical stability and physical interpretability.

Table 5.7 summarises the modelling results on the measurement dataset for various identification methods. The baseline model B2, which builds upon the BLA-initialised model B1 refined through LM optimisation, achieves the lowest prediction error of 3.66% with a modest number of parameters (24) and a relatively fast training time (79 seconds). This approach outperforms both its N4SID-initialised counterpart B4 (8.63%) and the proposed model P4 directly realised from the *prior-informed* Volterra kernel and achieves an RRMSE of 11.63%. Although P4 shows higher prediction error in this case, it maintains structural interpretability and is derived from a compact block-based model without requiring a full state-space optimisation.

In Figure 5.19, a comparative bar chart illustrates the output prediction accuracy, training time, and parameter count for various PNLSS models. The proposed Volterra-based

PNLSS model P4 achieves accuracy comparable to other PNLSS models (such as B4), while using the same number of parameters but requiring longer training time. This trade-off illustrates the benefit of incorporating structural insights early in the modelling framework; although it may introduce additional training overhead, it provides more informed initialisation and can guide the optimiser toward accurate solutions with fewer structural assumptions.

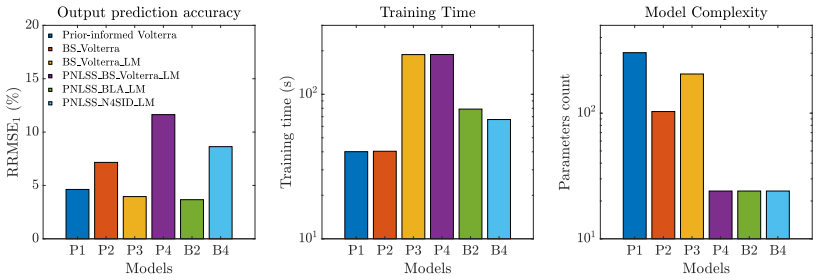


Figure 5.19: RRMSE$_1$ (%) of output prediction at SNR 25 dB, training time and number of parameters for PNLSS models. The proposed PNLSS model P4 achieves accuracy comparable to PNLSS models B2 and B4 initialised with linear methods, with similar complexity but increased training time.

The evolution of validation error during LM optimisation is depicted on the left in Fig. 5.20. Volterra-informed models (P3, P4), especially those initialised using regularised kernels, converge more quickly to the optimal solution, demonstrating better initialisation and better use of structural priors. In contrast, models based on linear initialisation (e.g., B4) require more iterations and still converge to higher final errors.

Finally, the test error profiles across all models are shown on the right in Fig. 5.20. The proposed PNLSS model P4 display low prediction errors, confirming its strong generalisation performance even under additive noise conditions. These results clearly show that *prior-informed regularisation*, coupled with structural decomposition and penalty-based optimisation, can lead to models that are both accurate and robust, particularly in scenarios involving known system structure.

5.4 Conclusion

In this chapter, we developed a systematic approach to identify Hammerstein systems by exploiting the structure of the Volterra kernel. By incorporating prior knowledge during the Volterra kernel estimation stage, we were able to regularise the solution space and guide the identification process toward physically meaningful and stable

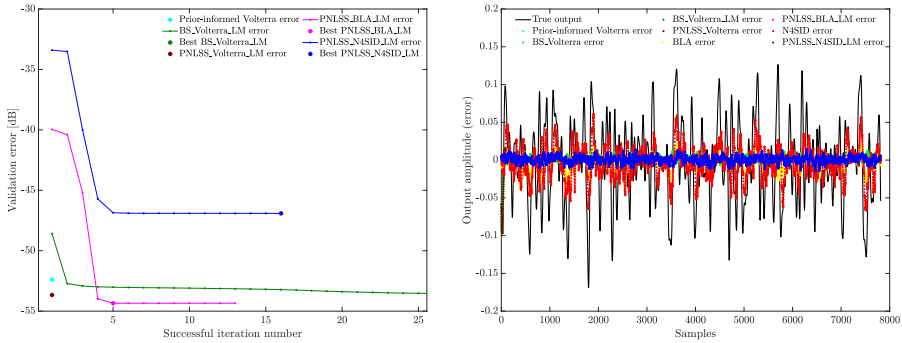


Figure 5.20: Left: Validation error over LM optimisation iterations. The Volterra-based methods (P3, P4) start closer to the optimum and converge faster. Right: Test set prediction errors in V for SNR 25 dB.

models. The decomposition of the Volterra kernel into a block-oriented structure allowed the extraction of a Hammerstein block-oriented model, which was further refined through nonlinear optimisation with shape-based penalties to enforce known filter characteristics and stability.

The modelling framework culminated in the realisation of a PNLSS model derived from the optimised Hammerstein representation. The resulting models demonstrated strong predictive accuracy, fast convergence during optimisation, and robustness to measurement noise. Simulation studies and experimental validation confirmed that the proposed framework performs in par with the classical state-space approaches, particularly in scenarios where the underlying system structure is known or can be reasonably approximated.

Overall, this chapter highlights the power of combining *prior-informed* regularisation, low-rank-based modelling, and structure-exploiting optimisation to achieve high-quality Hammerstein system identification. This approach lays the foundation for scalable and interpretable modelling frameworks in the following chapters, involving block-oriented and state-space representations.

Chapter 6

Volterra Kernel Estimation for Wiener System

Abstract: This chapter presents the proposed modelling framework for identifying Wiener structures using Volterra kernel estimation. The Wiener kernel is analytically constructed and reformulated as a structured tensor network, enabling efficient computation through the Tensor Train Matrix format. This representation supports scalable pseudoinverse computation and facilitates noise-robust identification. The reconstructed kernel is then used to extract the underlying linear filter and static nonlinearity through low-rank matrix approximation, followed by nonlinear optimisation to refine parameter estimates. From the optimised block-oriented model, a PNLSS model is realised via subspace techniques, enabling simulation-ready deployment. Regularisation and shape constraints are applied to ensure stability and physical consistency throughout the identification process. The proposed methodology is validated through numerical simulations and measurement data, and its generalisability is demonstrated on a set of benchmark nonlinear systems from process control. The results suggest that although the predictive accuracy is on par with existing approaches, the proposed method provides greater interpretability by explicitly identifying the underlying linear and nonlinear subsystems.

Note on Chapter Content: This chapter comprises both previously published and unpublished work by the PhD candidate, supplemented with selected material from existing literature to provide context and facilitate comparison. Specifically:

• Sections 6.1, 6.5, and 6.8 are based on P. Vijayan, P. Dreesen, J. Lataire and M. Ishteva, "Enhancing Block-Oriented System Identification Using *Prior-Informed* Volterra Kernels," in *IEEE Transactions on Instrumentation and Measurement*, vol. 74, pp. 1–11, 2025, Article No. 6507311. DOI: 10.1109/TIM.2025.3575968.

• Sections 6.6 and 6.7 present original, unpublished work by the PhD candidate.

• Sections 6.2, 6.3, and 6.4 are adapted from existing literature and revised to fit in the context of this thesis.

This research was conducted by the PhD candidate under the supervision of the co-authors, who served as (co-)promoters.

This chapter presents the development of an identification procedure for Wiener systems, in alignment with the overall research framework illustrated in Figure 1.6. Similar to the approach used for Hammerstein systems in the previous chapter, the identification procedure begins with the construction of the Volterra model, as described in Section 5.1. The subsequent sections focus on the derivation of the Wiener kernel, a structured representation of the Volterra kernel that corresponds to the specific form of a Wiener system.

6.1 Construction of Wiener Kernel

Similar to the Hammerstein system modelling approach, we assume a polynomial nonlinearity of degree D and an impulse response $\boldsymbol{s}$ of order M_{w} to describe the individual blocks of the Wiener system. The presence of a linear dynamic block preceding the nonlinearity (see Figure 6.1) complicates the identification process, as the impulse response coefficients are raised to powers up to D and scaled by factors $p_1, \cdots, p_D$:

$$q_k = \sum_{j=1}^{D} p_j \left(\sum_{i=0}^{M_{\mathrm{w}}} s_i u_{k-i} \right)^j . \tag{6.1}$$

To illustrate the equivalence between this Wiener model and a Dth-order finite memory Volterra model, consider the minimal working example:

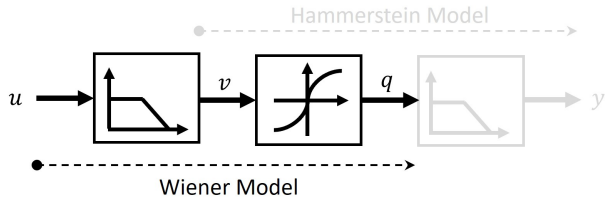


Figure 6.1: Wiener model with a linear dynamic block followed by a static nonlinear block. u is the input, q is the output, and v is the intermediate signal.

Example 13. For degree $D = 2$ and memory length $M_{\mathrm{w}} = 2$, the model (6.1) expands to:

$$q_k = p_1(s_0u_k + s_1u_{k-1} + s_2u_{k-2}) + p_2(s_0u_k + s_1u_{k-1} + s_2u_{k-2})^2$$

$$= \begin{bmatrix} u_k \\ u_{k-1} \\ u_{k-2} \\ u_k^2 \\ u_ku_{k-1} \\ u_ku_{k-2} \\ u_{k-1}^2 \\ u_{k-1}u_{k-2} \\ u_{k-2}^2 \end{bmatrix}^{\top} \begin{bmatrix} p_1s_0 \\ p_1s_1 \\ p_1s_2 \\ p_2s_0^2 \\ 2p_2s_0s_1 \\ 2p_2s_0s_2 \\ p_2s_1^2 \\ 2p_2s_1s_2 \\ p_2s_2^2 \end{bmatrix}. \tag{6.2}$$

This formulation reveals that the Wiener model includes all monomials in the model description, making the Wiener kernel dense. Thus, expanding the Volterra model (1.16) for the same hyperparameters D and M_{w}, gives:

$$q_k = \left(\begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} \otimes \begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} \right) \boldsymbol{h}. \tag{6.3}$$

It can be seen that the general Volterra model also includes all the monomials in the input. Therefore, a direct equivalence between the Wiener kernel (6.2) and the Volterra kernel can be derived.

The reshaped symmetric Volterra kernel tensor (5.7) for the Wiener structure with second-order nonlinearity is

$$\boldsymbol{H} = \left[\begin{array}{c|ccc|c} 0 & \frac{p_1 s_0}{2} & \frac{p_1 s_1}{2} & \dots & \frac{p_1 s_{M_\mathrm{w}}}{2} \\ \hline \frac{p_1 s_0}{2} & p_2 s_0^2 & p_2 s_0 s_1 & \dots & p_2 s_0 s_{M_\mathrm{w}} \\ \frac{p_1 s_1}{2} & p_2 s_0 s_1 & p_2 s_1^2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \frac{p_1 s_{M_\mathrm{w}}}{2} & p_2 s_0 s_{M_\mathrm{w}} & \dots & \dots & p_2 s_{M_\mathrm{w}}^2 \end{array}\right]. \tag{6.4}$$

A third-order symmetric Volterra kernel tensor that admits a Wiener structure has the same form as the general Volterra kernel illustrated in Figure 5.1. This kernel structure can be generalised for any Wiener model with higher-order nonlinearity. For a D-way tensor, the linear kernel components and diagonal kernel elements of orders 2 through $D-1$ are symmetrically scaled by a factor of $1/D$ along each of the D dimensions and the non-diagonal kernel elements are scaled up to $1/(D!)$ to preserve the symmetric structure of the kernel. It can be seen from (6.4) that all monomials of input u are included in the model, i.e., there is no structural prior available for Volterra kernel estimation, unlike the Hammerstein model, where the Volterra kernel is known to be diagonal (see (5.11)-(5.13)). Therefore, the size of $\boldsymbol{h} \in \mathbb{R}^{I^D}$ (where $I = M_\mathrm{w} + 2$) in $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ (5.4) increases exponentially, even for small values of M_w and D. Thus, constructing and inverting the matrix $\boldsymbol{U}$ for the Wiener structure is computationally expensive. This 'curse of dimensionality' is mitigated using a low-rank tensor-train decomposition (TTD) approach (Batselier, 2021), which solves (5.4) to obtain the symmetric Volterra kernel illustrated in Figure 5.1. Alternatively, several other techniques, such as ALS (Batselier, 2021), Bayesian Volterra tensor network (Memmel et al., 2023), Khatri-Rao product tensor network (Batselier, 2023), are also available to solve (5.4) using tensor networks. The following section introduces some basic concepts from Tensor algebra required to identify the Wiener kernel.

6.2 Tensor Basics

The fundamental concepts of tensor algebra, such as tensor representation, unfolding (matricisation), reshaping and vectorisation, tensor–matrix products, vector contractions, and the Kronecker product, are summarised in Appendix A. Readers unfamiliar with these notions are encouraged to consult that section before proceeding, as the following derivations build upon these basic operations.

6.2.1 Tensor Train Decomposition

The tensor network representation used in this thesis follows the Tensor Train Decomposition (TTD), also known as the Matrix Product State format in physics (Oseledets, 2011). The TTD expresses a D-way tensor $\mathcal{H} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ as a sequence of D third-order tensors or tensor train cores, denoted by $\mathcal{H}^{(1)}, \mathcal{H}^{(2)}, \ldots, \mathcal{H}^{(D)}$. This decomposition typically provides a low-rank approximation of the original tensor.

Each core $\mathcal{H}^{(k)}$ has dimensions $R_k \times I_k \times R_{k+1}$, where R_k and R_{k+1} are the tensor train ranks (TT-ranks). The decomposition reconstructs individual entries of $\mathcal{H}$ via a sequence of matrix multiplications:

$$h_{i_1 i_2 \ldots i_D} = \mathcal{H}^{(1)}_{i_1} \mathcal{H}^{(2)}_{i_2} \ldots \mathcal{H}^{(D)}_{i_D}, \tag{6.5}$$

where $\mathcal{H}^{(k)}_{i_k} \in \mathbb{R}^{R_k \times R_{k+1}}$ is a matrix obtained by fixing the second mode of $\mathcal{H}^{(k)}$ to index i_k. In MATLAB-style notation, this is expressed as $\mathcal{H}^{(k)}(:, i_k, :)$.

Since each entry $h_{i_1 i_2 \ldots i_D}$ is a scalar, the boundary condition require that $R_1 = R_{D+1} = 1$. An illustration of the TTD for a 3-way tensor is shown in Figure 6.2.

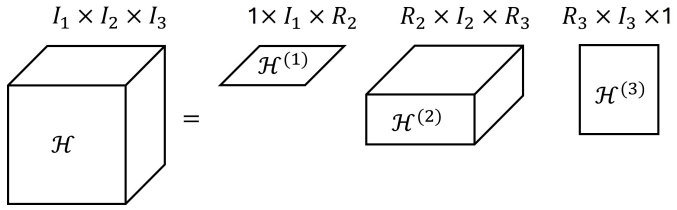


Figure 6.2: Tensor train cores of a 3-way tensor $\mathcal{H}$ of size $I_1 \times I_2 \times I_3$, where I_1, I_2, I_3 denote the mode sizes of the original tensor (rows, columns, slices). In the TT decomposition (right), each core $\mathcal{H}^{(k)}$ has size $R_k \times I_k \times R_{k+1}$, with boundary TT-ranks $R_1 = R_4 = 1$.

When all the TT-ranks are equal to a fixed value R, the total number of parameters required to store a cubic D-way tensor of dimensions I in the TT format is $O(DIR^2)$, which is significantly smaller than the $O(I^D)$. Therefore, using small TT-ranks leads to a substantial reduction in storage cost and computational savings, at the cost of approximation accuracy.

The TT-ranks required to exactly represent a given tensor $\mathcal{H}$ are referred to as its canonical TT-ranks. These ranks are upper bounded by the following result from Oseledets and Tyrtyshnikov (2010):

Theorem 2. For any tensor $\mathcal{H} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, there exists a TTD with TT-ranks satisfying

$$R_k \leq \min\left(\prod_{d=1}^{k-1} I_d, \prod_{d=k}^{D} I_d\right) = \min(I^{k-1}, I^{D-k+1}), \quad \text{for } k = 2, \ldots, D \tag{6.6}$$

When a TTD is constructed with ranks R_k that are smaller than the canonical ranks, the resulting representation is an approximation of the original tensor $\mathcal{H}$. This framework is extended to matrix-structured data using the Tensor Train Matrix format, discussed in the next section.

6.2.2 Tensor Train Matrices

In this thesis, particularly in Section 6.1, exponentially large matrices naturally arise due to the high dimensionality of the system under study. Direct manipulation or storage of such matrices is computationally expensive. To address this, the TTD framework (originally introduced for high-dimensional tensors) is extended to a matrix setting using the tensor train matrix (TTM) format (Oseledets, 2010). The TTM enables the compact representation and efficient computation of large-scale matrices, preserving the low-rank structure across both row and column modes.

Definition 6.2.1. Let $\boldsymbol{H} \in \mathbb{R}^{I^D \times J^D}$ be a large matrix whose entries are indexed as

$$\boldsymbol{H}([i_1 i_2 \ldots i_D], [j_1 j_2 \ldots j_D]). \tag{6.7}$$

where $I^D = I_1 I_2 \ldots I_D, \quad J^D = J_1 J_2 \ldots J_D$.

The TTM representation of $\boldsymbol{H}$ decomposes it into a product of D core tensors $\mathcal{H}^{(k)} \in \mathbb{R}^{R_k \times I_k \times J_k \times R_{k+1}}$, where each matrix entry (6.7) is reconstructed via a contracted product:

$$\boldsymbol{H}([i_1 i_2 \ldots i_D], [j_1 j_2 \ldots j_D]) = \sum_{r_2=1}^{R_2} \cdots \sum_{r_D=1}^{R_D} \mathcal{H}^{(1)}(1, i_1, j_1, r_2)\mathcal{H}^{(2)}(r_2, i_2, j_2, r_3) \ldots \mathcal{H}^{(D)}(r_D, i_D, j_D, 1). \tag{6.8}$$

Each $\mathcal{H}^{(k)}$ is a core tensor and the values $R_1, R_2, \ldots, R_{D+1}$ are called the TTM-ranks, with boundary conditions $R_1 = R_{D+1} = 1$ to ensure scalar output. This structured decomposition parallels the core concepts introduced in the TTD section, but extends them to represent matrices as tensor networks. A more intuitive understanding of TTM can be gained through tensor network diagrams, which will be introduced in the next section.

6.2.3 Tensor Diagrams

A tensor diagram offers a graphical representation of TTM. In these diagrams, each node corresponds to a tensor, and each edge represents an index. Index summations, as in (6.8), are visualised as edges connecting two nodes. Figure 6.3 shows the diagrams for scalars, vectors, matrices, 3-, and higher-dimensional tensors, while Figure 6.4 illustrates a TTM. For simplicity, the edges corresponding to $R_1 = R_{D+1} = 1$ are typically omitted.

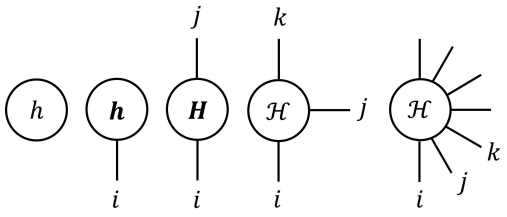


Figure 6.3: Elements of Tensor Diagrams: Graphical Representation of scalars, vectors, matrices, and higher-order tensors via node-and-edge connections. Each node represents a tensor, and each edge corresponds to a dimension or index of the tensor.

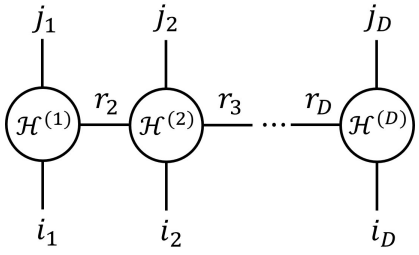


Figure 6.4: Graphical Representation of tensor train matrix in (6.8). The edges corresponding to boundary ranks indices r_1 and r_{D+1} are omitted.

High-dimensional matrices are often encountered in system identification. These matrices can grow exponentially in size with the system order D, making storage and computation infeasible. The TTM representation offers a scalable alternative by factorising such matrices into a sequence of smaller core tensors, significantly reducing the storage cost.

Example 14. Consider a matrix $\boldsymbol{H} \in \mathbb{R}^{I^D \times J^D}$. It contains $I^D J^D$ entries, resulting in exponential storage complexity. In the TTM representation, $\boldsymbol{H}$ is denoted using D core tensors $\mathcal{H}^{(k)} \in \mathbb{R}^{R_k \times I_k \times J_k \times R_{k+1}}$, where $R_1 = R_{D+1} = 1$, and R_k are the so-called TT-matrix ranks.

Assuming uniform dimensions $I_k = I, J_k = J$ and ranks $R_2 = \cdots = R_D = R$, the total number of stored parameters is:

$$\sum_{k=1}^{D} R_k I J R_{k+1} \approx DIJR^2.$$

Hence, the storage complexity becomes $O(DIJR^2)$, a drastic improvement over the exponential $O(I^D J^D)$ required for a full matrix. The efficiency depends on how small the TT-matrix ranks R_k are.

6.3 Wiener Kernel as Tensor Network

To maintain consistency and simplify notation in the remainder of this chapter, we denote the output of the Wiener system by y instead of q. This relabeling avoids the need to rewrite equations introduced in the previous chapter and does not affect the generality of the formulation.

The Volterra kernel representation in $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$(5.4) involves the matrix $\boldsymbol{U}$, whose rows consist of repeated Kronecker product $\boldsymbol{u}_k^D$ as defined in (5.2). Due to the repetition of terms in each Kronecker product, $\boldsymbol{U}$ has low rank. The following lemma provides an upper bound on the number of distinct entries in each row of $\boldsymbol{U}$.

Lemma 3. The total number of distinct entries in the vector $\boldsymbol{u}_k^D$ is upper bounded by:

$$R = \binom{D + I - 1}{I - 1}. \tag{6.9}$$

This result follows from interpreting $\boldsymbol{u}_k^D$ as the vectorisation of a symmetric rank-one tensor formed by the D-fold outer product of $\boldsymbol{u}_k$ with itself. Due to the symmetry, many entries in the tensor are repeated, and the number of unique monomials of degree D in I variables is given by (6.9).

Let us illustrate this with a concrete example.

Example 15. Consider a discrete-time SISO Volterra model of degree $D = 3$ and memory $M = 2$:

$$\begin{aligned} y_k = {} & h_0 + h_1(0)u_k + h_1(1)u_{k-1} + h_1(2)u_{k-2} + h_2(0,0)u_k^2 + 2h_2(0,1)u_k u_{k-1} \\ & + 2h_2(0,2)u_k u_{k-2} + h_2(1,1)u_{k-1}^2 + 2h_2(1,2)u_{k-1}u_{k-2} + h_2(2,2)u_{k-2}^2 \\ & + h_3(0,0,0)u_k^3 + 3h_3(0,0,1)u_k^2 u_{k-1} + 3h_3(0,0,2)u_k^2 u_{k-2} + 3h_3(0,1,1)u_k u_{k-1}^2 \\ & + 6h_3(0,1,2)u_k u_{k-1} u_{k-2} + 3h_3(0,2,2)u_k u_{k-2}^2 + h_3(1,1,1)u_{k-1}^3 \\ & + 3h_3(1,1,2)u_{k-1}^2 u_{k-2} + 3h_3(1,2,2)u_{k-1}u_{k-2}^2 + h_3(2,2,2)u_{k-2}^3, \end{aligned} \tag{6.10}$$

where the kernel is symmetric, i.e., $h_2(0,1) = h_2(1,0)$, and so on. Repeated permutations of the same variables are grouped and multiplied by the appropriate combinatorial coefficients. This expression can be simplified using the Kronecker product (5.2):

$$y_k = \left(\begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} \otimes \begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} \otimes \begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ u_{k-2} \end{bmatrix} \right)^\top \boldsymbol{h}, \tag{6.11}$$

where $\boldsymbol{h} \in \mathbb{R}^{64}$ contains the Volterra kernel coefficients. However, many entries in $\boldsymbol{u}_k^3$ are repeated due to symmetry, and hence only $R = \binom{6}{3} = 20$ of the entries are distinct.

Rewriting the coefficient vector $\boldsymbol{h}$ as a 3-way symmetric tensor $\mathcal{H} \in \mathbb{R}^{4\times4\times4}$ allows us to express the output compactly as:

$$y_k = \mathcal{H}\boldsymbol{u}_k^3, \tag{6.12}$$

using the tensor contraction operation (see Definition A.1.4 from Qi (2005)).

The implications of Lemma 3 extend to the full matrix $\boldsymbol{U}$, which stacks $\boldsymbol{u}_k^D$ for $k = 1, \ldots, N$.

Theorem 3. Let $\boldsymbol{U} \in \mathbb{R}^{N \times I^D}$ be the matrix defined by stacking $\boldsymbol{u}_k^D$ row-wise. Then:

$$\operatorname{rank}(\boldsymbol{U}) \leq R = \binom{D+I-1}{I-1}, \tag{6.13}$$

provided that $N \geq R$.

Theorem 3 follows directly from the fact that each $\boldsymbol{u}_k^D$ has only R unique entries. Thus, $\boldsymbol{U}$ can have at most R linearly independent columns, regardless of the larger ambient dimension I^D. This justifies the requirement that the input signal be persistently exciting of order D and memory M to ensure identifiability.

A consequence of this rank deficiency is that the linear-in-the-parameters system $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ in (5.4) has infinitely many solutions. However, among these, a unique minimum-norm solution exists. As shown in Batselier et al. (2017), this solution corresponds to the vectorisation of a symmetric tensor. Let us consider the Example 15 to illustrate this.

Suppose a solution $\boldsymbol{h}$ is not symmetric, i.e., the kernel coefficients corresponding to repeated monomials in $\boldsymbol{u}_k^D$ differ. For instance, the term u_k appears as the second, fifth and seventeenth entry of $\boldsymbol{u}_k^3$. One can construct a symmetrised version of $\boldsymbol{h}$, denoted $\boldsymbol{h}_{\text{symm}}$, by averaging these coefficients:

$$\boldsymbol{h}_{\text{symm},i} = \frac{h_2 + h_5 + h_{17}}{3}. \tag{6.14}$$

It follows that:

$$||\boldsymbol{h}_{\text{symm}}||_2^2 \leq ||\boldsymbol{h}||_2^2, \tag{6.15}$$

with equality if and only if $h_2 = h_5 = h_{17}$. Hence, enforcing symmetry on the Volterra kernel corresponds to selecting the unique minimum-norm solution of $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ (5.4). This is addressed in detail in the following section, accompanied by a relevant example.

6.4 Pseudoinverse in TTM Form

One way to obtain the minimum-norm solution to the linear system $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$(5.4) is via the Moore-Penrose pseudoinverse. Assume $N \geq R$ and let

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{Q}_1 & \boldsymbol{Q}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1^\top \\ \boldsymbol{V}_2^\top \end{bmatrix} = \boldsymbol{Q}_1 \boldsymbol{S} \boldsymbol{V}_1^\top, \tag{6.16}$$

be the compact SVD of $\boldsymbol{U} \in \mathbb{R}^{N \times I^D}$, where $\boldsymbol{S} \in \mathbb{R}^{R \times R}$ is diagonal with the non-zero singular values, and $\boldsymbol{Q} \in \mathbb{R}^{N \times N}$, $\boldsymbol{V} \in \mathbb{R}^{I^D \times I^D}$ are orthogonal matrices. Then the unique minimum-norm solution of $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$(5.4) is given by:

$$\boldsymbol{h}_{\text{symm}} = \boldsymbol{U}^{\dagger}\boldsymbol{y} = \boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^{\top} \boldsymbol{y}. \tag{6.17}$$

Although no explicit symmetry constraint is imposed during this process, the resulting solution is inherently symmetric. This is because the matrix $\boldsymbol{U}$ is constructed from repeated Kronecker products of the same input vector, and its column space spans only those coefficient vectors that are invariant under permutations of indices. Any asymmetric component of $\boldsymbol{h}$ belongs to $\ker(\boldsymbol{U})$, and thus it does not affect the output $\boldsymbol{y}$.

To see why the pseudoinverse picks out a symmetric solution, suppose a vector $\boldsymbol{\delta} \in \mathbb{R}^{I^D}$ satisfies:

$$\boldsymbol{U}\boldsymbol{\delta} = \boldsymbol{0}.$$

This means adding $\boldsymbol{\delta}$ to any solution $\boldsymbol{h}$ doesn't change the output:

$$\boldsymbol{U}(\boldsymbol{h} + \boldsymbol{\delta}) = \boldsymbol{U}\boldsymbol{h} + \boldsymbol{U}\boldsymbol{\delta} = \boldsymbol{y}.$$

Hence, the general solution can be written as:

$$\boldsymbol{h} = \boldsymbol{h}_{\text{symm}} + \boldsymbol{\delta}, \quad \boldsymbol{\delta} \in \ker(\boldsymbol{U}).$$

Because $\boldsymbol{h}_{\text{symm}}$ lies in the column space of $\boldsymbol{U}$, it is orthogonal to every $\boldsymbol{\delta} \in \ker(\boldsymbol{U})$. Then,

$$||\boldsymbol{h}||_2^2 = ||\boldsymbol{h}_{\text{symm}}||_2^2 + ||\boldsymbol{\delta}||_2^2 + 2\langle \boldsymbol{h}_{\text{symm}}, \boldsymbol{\delta} \rangle = ||\boldsymbol{h}_{\text{symm}}||_2^2 + ||\boldsymbol{\delta}||_2^2 > ||\boldsymbol{h}_{\text{symm}}||_2^2. \tag{6.18}$$

Since $\boldsymbol{h}_{\text{symm}}$ is orthogonal to $\boldsymbol{\delta}$, the cross term is 0. Any non-zero $\boldsymbol{\delta}$ (in particular, an asymmetric component) would increase the L^2-norm. Therefore, the unique minimum-norm solution must lie entirely in the symmetric subspace.

To illustrate this concretely, consider the following small numerical example.

Example 16. Let $D = 2$ and $I = 2$, and suppose the observed input is

$$\boldsymbol{u} = \begin{bmatrix} u_0 & u_1 & u_2 \end{bmatrix}^{\top} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^{\top},$$

so that we have $N = 3$ total samples. For a second-order Volterra model, there are $I^D = 4$ (potentially distinct) kernel elements $\{h_{11}, h_{12}, h_{21}, h_{22}\}$. However, because the Volterra kernel is symmetric, only three degrees of freedom matter: h_{11}, $h_{12} = h_{21}$, and h_{22}.

In this toy example, we form two valid "lag-2" snapshots (since $D = 2$). Using the Kronecker product, the below matrix $\boldsymbol{U} \in \mathbb{R}^{3\times 4}$ is constructed:

$$\boldsymbol{U} = \begin{bmatrix} 1 & u_1 & u_1 & u_1^2 \\ 1 & u_2 & u_2 & u_2^2 \\ 1 & u_3 & u_3 & u_3^2 \end{bmatrix}, \tag{6.19}$$

where each row corresponds to a snapshot of degree-2 monomials from the input. Now, suppose the corresponding outputs are

$$\boldsymbol{y} = \begin{bmatrix} 0.34 & 3.57 & 2.76 \end{bmatrix}^\top. \tag{6.20}$$

Solving $\boldsymbol{U}\boldsymbol{h} = \boldsymbol{y}$ via the pseudoinverse, $\boldsymbol{h} = \boldsymbol{U}^\dagger \boldsymbol{y}$, one finds numerically

$$\boldsymbol{h} = \begin{bmatrix} h_{11} & h_{12} & h_{21} & h_{22} \end{bmatrix}^\top = \begin{bmatrix} -6.93 & 4.64 & 4.64 & -2.02 \end{bmatrix}^\top. \tag{6.21}$$

Notice that $h_{12} = h_{21}$ automatically, even though we never forced that constraint. In other words, the pseudoinverse "knows" to collapse any asymmetric direction $(h_{12} - h_{21})$ into the null space of $\boldsymbol{U}$ and it picks the unique minimum-norm solution in which $h_{12} = h_{21}$. This verifies that the Volterra kernel recovered by $\boldsymbol{U}^\dagger \boldsymbol{y}$ is indeed symmetric.

Hence, the symmetric Volterra kernel arises naturally as a consequence of both the structure of $\boldsymbol{U}$ (which spans only symmetric combinations) and the minimum-norm criterion of the Moore–Penrose pseudoinverse.

6.4.1 Construction of Tensor Network

The matrix $\boldsymbol{U}$ has dimensions $N \times I^D$. As the memory length M and degree D increase, this leads to an exponential growth in the size of $\boldsymbol{U}$, making it computationally expensive to construct and invert (as shown in the previous Section). To alleviate this problem, $\boldsymbol{U}$ can be represented using a tensor network, allowing efficient computations without explicitly forming the full matrix. This mitigates the curse of dimensionality associated with the Volterra model described in $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ (5.4).

Notably, $\boldsymbol{U}$ exhibits a low-rank structure due to repeated Kronecker products, which results in repeating columns. This structure can be exploited to construct a compact tensor train representation efficiently.

Let us define a matrix $\tilde{\boldsymbol{U}} \in \mathbb{R}^{N \times I}$ as:

$$\tilde{\boldsymbol{U}} = \begin{bmatrix} \boldsymbol{u}_1^\top \\ \boldsymbol{u}_2^\top \\ \vdots \\ \boldsymbol{u}_N^\top \end{bmatrix}. \tag{6.22}$$

Then, the matrix $\boldsymbol{U}$ can be written using the row-wise Kronecker product (also known as the Khatri-Rao product) of $\tilde{\boldsymbol{U}}$ as:

$$\boldsymbol{U} = \underbrace{\tilde{\boldsymbol{U}} \odot \tilde{\boldsymbol{U}} \odot \cdots \odot \tilde{\boldsymbol{U}}}_{D \text{ times}}. \tag{6.23}$$

The operator $\odot$, called the Khatri-Rao product, is defined as:

Definition 6.4.1 (Khatri-Rao product)**.** Given two matrices $\boldsymbol{A} \in \mathbb{R}^{I \times J}$ and $\boldsymbol{B} \in \mathbb{R}^{I \times K}$, the (row-wise) Khatri-Rao product is defined as the row-wise Kronecker product.

$$\boldsymbol{A} \odot \boldsymbol{B} = \begin{bmatrix} \boldsymbol{a}_1 \\ \vdots \\ \boldsymbol{a}_I \end{bmatrix} \odot \begin{bmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_I \end{bmatrix} = \begin{bmatrix} \boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{a}_I \otimes \boldsymbol{b}_I \end{bmatrix} \in \mathbb{R}^{I \times JK}, \tag{6.24}$$

where $\boldsymbol{a}_i, \boldsymbol{b}_i, i = 1, \ldots, I$ are row vectors of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively.

The tensor network for $\boldsymbol{U}$ can be constructed efficiently using $(D-1)$ SVDs and Khatri-Rao products as shown in Algorithm 4, adapted from Batselier (2021).

Algorithm 4 constructs the tensor network by iteratively performing Khatri-Rao products followed by SVD to determine the tensor train ranks, starting from $\mathcal{U}^{(D)}$ and proceeding backwards to $\mathcal{U}^{(1)}$. The computational bottleneck is the SVD step, with complexity approximately $\mathcal{O}(R_{d+1} I^3 N^2)$ flops. A graphical illustration of this iterative construction is shown in Figure 6.5.

Moreover, we can derive an upper bound for the tensor network ranks of the $\boldsymbol{U}$:

Theorem 4. The TT-matrix ranks of the matrix $\boldsymbol{U}$ in $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ (5.4) satisfy

$$R_d \leq \binom{D-d+I}{I-1}. \tag{6.25}$$

Algorithm 4 Construction of TTM of repeated row-wise Kronecker product of matrices (Batselier, 2021)

Input: Matrix $\tilde{\boldsymbol{U}} \in \mathbb{R}^{N \times I}$ and order D
Output: Tensor network $\mathcal{U}^{(1)}, \ldots, \mathcal{U}^{(D)}$ of $\boldsymbol{U}$ in (6.23)

$\mathcal{U}^{(D)} \leftarrow \text{reshape}(\tilde{\boldsymbol{U}}, [1, N, I, 1])$
for $d = D : -1 : 2$, **do**
  $\boldsymbol{T} \leftarrow \text{reshape}(\mathcal{U}^{(d)}, [N, IR_{d+1}])$
  $\boldsymbol{T} \leftarrow \boldsymbol{T} \odot \tilde{\boldsymbol{U}}$
  $\boldsymbol{T} \leftarrow \text{reshape}(\boldsymbol{T}, [NI, IR_{d+1}])$
  $[\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{V}] \leftarrow \text{SVD}(\boldsymbol{T})$
  Truncate $\boldsymbol{Q}, \boldsymbol{S}, \boldsymbol{V}$ to rank R_d
  $\mathcal{U}^{(d)} \leftarrow \text{reshape}(\boldsymbol{V}^\top, [R_d, 1, I, R_{d+1}])$
  $\mathcal{U}^{(d-1)} \leftarrow \text{reshape}(\boldsymbol{QS}, [1, N, I, R_d])$
end for

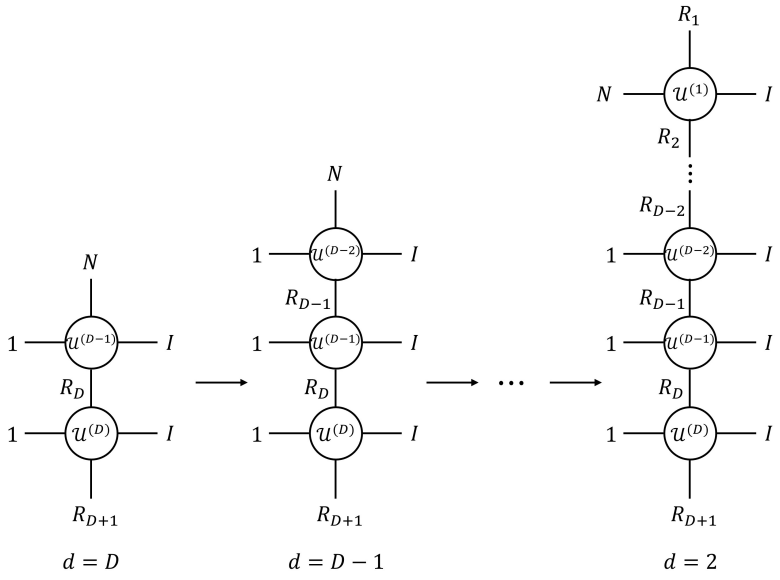


Figure 6.5: Tensor Network of repeated Khatri-Rao product of $\tilde{\boldsymbol{U}}$ in each iteration, showing the TT ranks.

The theorem has been proved for a matrix with a slightly different structure but sharing the repeated Khatri-Rao product form in Batselier et al. (2018). Below, we present the proof tailored to our specific case.

Proof. Based on Theorem 3, the number of distinct columns R_d in the $(D-d+1)$-fold Khatri-Rao products is bounded by

$$R_d \leq \binom{(D-d+1)+I-1}{I-1} = \binom{D-d+I}{I-1}, \tag{6.26}$$

instead of $I^{D-d+1}, \forall D+1 \geq d \geq 2$. This reverse ordering of d is consistent with the construction flow adopted in Algorithm 4 and illustrated in Figure 6.5. □

This bound shows that the TT-matrix ranks grow polynomially with I and D, contrasting with the exponential growth in the naive matrix representation. In practice, when inputs are persistently exciting, these upper bounds are typically attained. The following example compares the tensor network ranks of $\boldsymbol{U}$ with the conventional upper bounds of Theorem 2.

Example 17. Consider a simple SISO system with $D = 10$ and $I = 2$. The tensor network ranks of $\boldsymbol{U}$ are shown by the blue dotted line in Figure 6.6, whereas the conventional upper bounds are represented by the blue solid line. Notice the difference of one order of magnitude for R_6 (the rank of the middle core). This difference becomes ever larger as I (memory length) increases. For $I = 10, D = 10$, the modified rank $R_6 = 2002$ compared to the conventional upper bound of $R_6 = 10^5$, shown in red in Figure 6.6.

The current linear system $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ (5.4) can be expressed using a tensor diagram (ref. Section 6.2.3) as shown in Figure 6.7, where the matrix $\boldsymbol{U}$ is represented by TTMs:

$$\mathcal{U}^{(1)} \in \mathbb{R}^{1\times N\times I\times R_2}$$

with the second index of the remaining tensors $\mathcal{U}^{(2)}, \ldots, \mathcal{U}^{(D)}$ being one-dimensional. Similarly, the Volterra kernel vector $\boldsymbol{h} \in \mathbb{R}^{I^D}$ is represented using a TT format. The first tensor is given by:

$$\mathcal{H}^{(1)} \in \mathbb{R}^{1\times I\times R_2},$$

and the remaining tensors $\mathcal{H}^{(2)}, \ldots, \mathcal{H}^{(D)}$ are determined by the TT-matrix ranks.

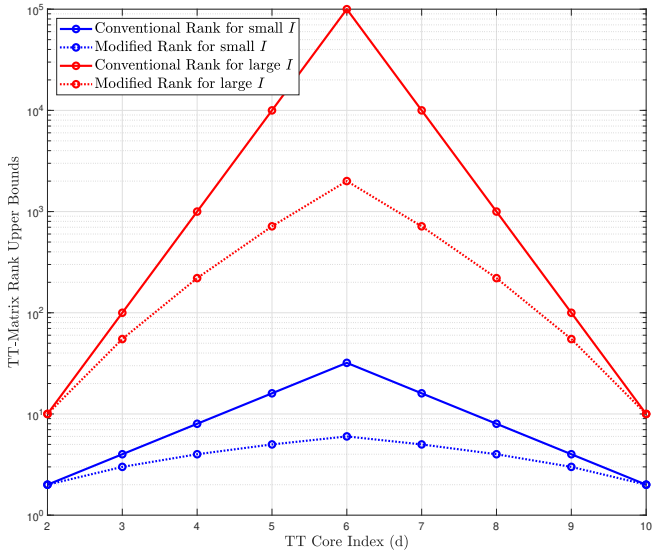


Figure 6.6: Comparison of Upper Bounds of Conventional and Modified TT-Matrix Ranks of $\boldsymbol{U}$.

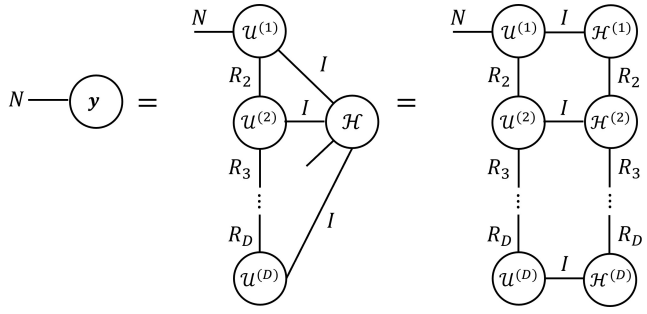


Figure 6.7: Tensor diagram of SISO Volterra model (5.4). The matrix $\boldsymbol{U}$ and the kernel vector $\boldsymbol{h}$ are represented by tensor train matrices.

Summing over all connected indices in the diagram yields the predicted output vector $\boldsymbol{y} \in \mathbb{R}^N$. This tensorised formulation reduces storage and computation, enabling scalable identification of nonlinear systems in Volterra models.

6.4.2 Unique Minimum-Norm Solution in TTM Format

Once the TTM representation of $\boldsymbol{U}$ is obtained using Algorithm 4, its pseudoinverse, and hence the unique minimum-norm solution, can be computed efficiently while staying entirely within the TTM framework.

To do this, we first compute the thin SVD of $\boldsymbol{U}$ as described in (6.16). This is achieved by performing a single SVD on the first core of the TTM, denoted as $\mathcal{U}^{(1)}$. This core is reshaped into a matrix $\boldsymbol{U}_1 \in \mathbb{R}^{N \times IR_2}$, after which its SVD is computed:

$$\boldsymbol{U} = \boldsymbol{Q}_1 \boldsymbol{S} \boldsymbol{V}_1^\top, \tag{6.27}$$

where $\boldsymbol{Q}_1 \in \mathbb{R}^{N \times R}$ contains the left singular vectors, $\boldsymbol{S} \in \mathbb{R}^{R \times R}$ is diagonal with the non-zero singular values, $\boldsymbol{V} \in \mathbb{R}^{IR_2 \times R}$ contains the right singular vectors.

If the input sequence is persistently exciting (i.e. sufficiently rich), then the rank R of $\boldsymbol{U}_1$ equals the dimension of the symmetric Volterra kernel space (ref. Lemma 3 and Theorem 3):

$$R = \binom{D + I - 1}{I - 1}. \tag{6.28}$$

After the SVD, the first core of the TTM for $\boldsymbol{V}_1$ is reconstructed by reshaping $\boldsymbol{V}$ into a 4-dimensional tensor:

$$\mathcal{V}^{(1)} \in \mathbb{R}^{1 \times R \times I \times R_2}. \tag{6.29}$$

The remaining cores of $\boldsymbol{V}_1$ are simply inherited from the original TTM of $\boldsymbol{U}$. This process is summarised in Algorithm 5.

Using the components from the thin SVD, the symmetric minimum-norm solution $\boldsymbol{h}_{\mathrm{symm}}$ can be obtained using the expression:

$$\boldsymbol{h}_{\mathrm{symm}} = \boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^\top \boldsymbol{y}. \tag{6.30}$$

This operation can be efficiently implemented in the TTM format by modifying only the first core of the TTM corresponding to $\boldsymbol{V}_1$. Specifically, the matrices $\boldsymbol{S}^{-1}$ and $\boldsymbol{Q}_1^\top$ and the vector $\boldsymbol{y}$ are contracted into the first core, effectively absorbing all the operations required to obtain $\boldsymbol{h}_{\mathrm{symm}}$. This process is visualised in Figure 6.8.

Since only the first core is modified, the TT-matrix ranks of $\boldsymbol{h}_{\mathrm{symm}}$ are initially identical to those of $\boldsymbol{V}_1$. However, due to the contraction with $\boldsymbol{y} \in \mathbb{R}^N$, the column dimension is reduced to 1, suggesting that the TT-matrix ranks of $\boldsymbol{h}_{\mathrm{symm}}$ can be significantly smaller.

Algorithm 5 Thin SVD of $\boldsymbol{U}$ in TTM form (Batselier, 2021)

Input: TTM of $\boldsymbol{U}$ from Algorithm 4
Output: Matrices $\boldsymbol{Q}_1, \boldsymbol{S}$ and TTM $\boldsymbol{V}_1$ in (6.16)

1: $\boldsymbol{U}_1 \leftarrow \text{reshape}(\mathcal{U}^{(1)}, [N, IR_2])$
2: $[\boldsymbol{Q}_1, \boldsymbol{S}, \boldsymbol{V}] \leftarrow \text{SVD}(\boldsymbol{U}_1)$
3: Truncate $\boldsymbol{Q}_1, \boldsymbol{S}, \boldsymbol{V}$ to a rank $R = \binom{D+I-1}{I-1}$
4: $\mathcal{V}^{(1)} \leftarrow \text{reshape}(\boldsymbol{V}, [1, R, I, R_2])$
5: **for** $d = 2 : D$ **do**
6: $\quad \mathcal{V}^{(d)} \leftarrow \mathcal{U}^{(d)}$
7: **end for**

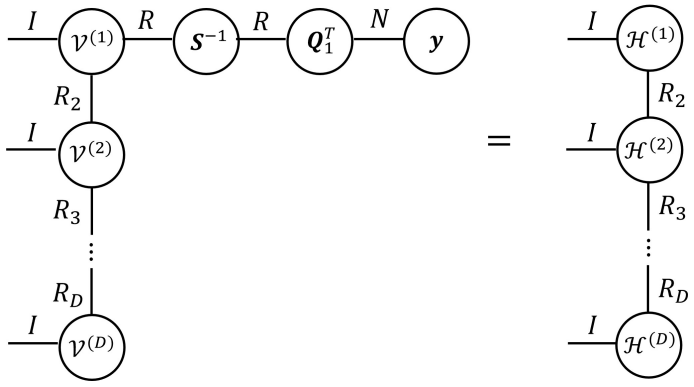


Figure 6.8: Tensor diagram of the symmetric minimum-norm solution $\boldsymbol{h}_{\text{symm}}$ (6.30) using the pseudoinverse of $\boldsymbol{U}$ in TTM form. The vector $\boldsymbol{y}$ is absorbed into the first core, resulting in the TTM of $\boldsymbol{h}$.

These ranks can be further reduced using a sequence of SVD-based truncations without any loss of accuracy (Oseledets, 2011). Moreover, upper bounds on the TT-matrix ranks can be derived from the inherent symmetry of $\boldsymbol{h}_{\text{symm}}$, as shown below.

Theorem 5. Let $\boldsymbol{h}_{\text{symm}}$ be the symmetric minimum-norm solution in TTM format. Then its TT-matrix ranks R_d satisfy:

$$R_d \leq \min\left\{\binom{d+I-1}{I-1}, \binom{D-d+I-1}{I-1}\right\}, \quad \text{for } d = 1, \ldots, D-1. \tag{6.31}$$

Proof. The above theorem can be proved using a function space dimension argument. Because the kernel is symmetric, it belongs in the vector space $\text{Sym}^D(\mathbb{R}^I)$, the space

of symmetric order-D tensors over $\mathbb{R}^I$. Its dimension is therefore given by:

$$\dim\{\mathrm{Sym}^D(\mathbb{R}^I)\} = \binom{D+I-1}{I-1}.$$

Any unfolding of this tensor (e.g., mode-d matricisation or TT-matrix unfolding) involves a bilinear map between symmetric tensors of degree d and $D-d$:

$$\mathrm{Sym}^d(\mathbb{R}^I) \otimes \mathrm{Sym}^{D-d}(\mathbb{R}^I), \quad \text{for } d = 1, \ldots, D-1,$$

whose rank is upper bounded by the product of the dimensions of the two subspaces:

$$\binom{d+I-1}{I-1} \cdot \binom{D-d+I-1}{I-1}.$$

But the matrix rank is bounded by both row and column space dimensions. So:

$$R_d \leq \min\left\{ \binom{d+I-1}{I-1}, \binom{D-d+I-1}{I-1} \right\}, \quad \text{for } d = 1, \ldots, D-1.$$

□

Example 18. Consider a symmetric third-order Volterra kernel $\boldsymbol{h}_{\mathrm{symm}}$ for degree $D = 3$ and input memory length $I = 3$, corresponding to input values $1, u_k, u_{k-1}$. The kernel is represented as a symmetric 3-dimensional tensor:

$$\boldsymbol{h}_{\mathrm{symm}}(i_1, i_2, i_3), \quad \text{with } i_1, i_2, i_3 \in \{1, 2, 3\}.$$

Without exploiting symmetry, the total number of entries in the tensor is $3 \times 3 \times 3 = 27$. However, due to full permutation symmetry, the number of unique entries is significantly reduced. Each unique value corresponds to a multiset of size 3 drawn from 3 elements, and the total number of such multisets is given by the binomial coefficient:

$$\binom{D+I-1}{I-1} = \binom{5}{2} = 10.$$

This means the 27 entries of the tensor reduce to just 10 unique values, with repeated values due to symmetric permutations.

To analyse the tensor train representation, consider unfolding the tensor along the first and second TT-cores. The rank bounds from Theorem 5 state that the TT-matrix ranks

R_1 and R_2 satisfy:

$$R_1 \leq \min\left\{\binom{3}{2}, \binom{4}{2}\right\} = \min\{3, 6\} = 3,$$

$$R_2 \leq \min\left\{\binom{4}{2}, \binom{3}{2}\right\} = \min\{6, 3\} = 3.$$

Thus, the TT-matrix ranks for $\boldsymbol{h}_{\mathrm{symm}}$ are bounded by:

$$R_0 = R_3 = 1, \quad R_1 = R_2 \leq 3,$$

and the TTM representation has the format $(1, 3, 3, 1)$. This is a significant reduction in representation complexity. For instance, the mode-1 unfolding $\boldsymbol{H}_1 \in \mathbb{R}^{3\times 9}$ or the mode-2 unfolding $\boldsymbol{H}_2 \in \mathbb{R}^{9\times 3}$ would typically require higher-rank factorisations in the absence of symmetry. The TT-format, however, efficiently leverages symmetry to achieve compact representation with reduced computational cost.

The theoretical rank bound is tight for dense, non-sparse Volterra kernels. However, in practice, many real-world systems exhibit sparse or rapidly decaying kernels, which implies that the actual TT-matrix ranks of $\boldsymbol{h}_{\mathrm{symm}}$ are often significantly lower than the upper bound. For example, the Volterra kernel of the Wiener system is known to decay in the ideal, noise-free case (Birpoutsoukis et al., 2017). Nevertheless, the presence of measurement noise can disrupt the low-rank structure and lead to inflated TT-matrix ranks. The next Subsection discusses in detail how measurement noise impacts the compressibility and rank profile of $\boldsymbol{h}_{\mathrm{symm}}$.

6.4.3 Impact of Measurement Noise on the TT-Ranks of $\boldsymbol{h}_{\mathrm{symm}}$

In practice, output measurements are often corrupted by additive noise. This affects the system identification problem because the observed system becomes:

$$\boldsymbol{y} + \boldsymbol{e} = \boldsymbol{U}\boldsymbol{h}, \tag{6.32}$$

where $\boldsymbol{e} \in \mathbb{R}^N$ denotes the output measurement noise, assumed to be zero-mean and mutually uncorrelated. Crucially, this noise does not affect the input regressor matrix $\boldsymbol{U}$, which is constructed from known input data.

The symmetric minimum-norm solution to the perturbed system (6.32) is then given by:

$$\hat{\boldsymbol{h}}_{\mathrm{symm}} = \boldsymbol{U}^{\dagger}(\boldsymbol{y} + \boldsymbol{e}) = \boldsymbol{h}_{\mathrm{symm}} + \boldsymbol{U}^{\dagger}\boldsymbol{e}. \tag{6.33}$$

Using the thin SVD representation $\boldsymbol{U} = \boldsymbol{Q}_1 \boldsymbol{S} \boldsymbol{V}_1^\top$, this can be rewritten as:

$$\hat{\boldsymbol{h}}_{\mathrm{symm}} = \boldsymbol{h}_{\mathrm{symm}} + \boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^\top \boldsymbol{e}. \tag{6.34}$$

The correction term $\boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^\top \boldsymbol{e}$ is a linear transformation of the noise vector $\boldsymbol{e}$ and, importantly, lies in the column space of $\boldsymbol{V}_1$. Since $\boldsymbol{V}_1$ is a TTM with TT-matrix ranks at their maximal values (determined by the structure of $\boldsymbol{U}$), this implies that the noise-induced perturbation can fully excite all degrees of freedom of the TT-format. That is, the correction term is symmetric (because $\boldsymbol{V}_1$ is symmetric) but of high rank.

Even if the true kernel $\boldsymbol{h}_{\mathrm{symm}}$ is low-rank in TT-format (for example, due to sparsity or decaying dynamics), the added noise introduces full-rank components within the symmetric space. This leads to an observed solution $\hat{\boldsymbol{h}}_{\mathrm{symm}}$ whose TT-matrix ranks are pushed to their upper bounds, as defined in Theorem 5.

Example 19. Consider the earlier Example 18 with $D = 3$ and $I = 3$. Suppose the true kernel $\boldsymbol{h}_{\mathrm{symm}}$ has TT-Matrix ranks $\{1, 2, 2, 1\}$ due to its sparsity. If a small amount of white noise $\boldsymbol{e}$ is added to the output, then the correction term $\boldsymbol{V}_1 \boldsymbol{S}^{-1} \boldsymbol{Q}_1^\top \boldsymbol{e}$ will typically have TT-Matrix ranks $\{1, 3, 3, 1\}$, that is, at the maximum permitted by symmetry. Hence, the perturbed estimate $\hat{\boldsymbol{h}}_{\mathrm{symm}}$ also has full TT-Matrix rank.

This phenomenon has serious implications for practical Volterra kernel identification:

- The loss of low-rank structure makes storage and computation more expensive.
- The sensitivity of TT-Matrix ranks to noise highlights the ill-conditioning of the inverse problem.
- Low-rank identification approaches must incorporate regularisation or denoising strategies to mitigate the effects of measurement noise.

This behaviour is particularly pronounced in systems such as Wiener models, where the true kernels are known to decay (Birpoutsoukis et al., 2017). Despite this decay, the presence of measurement noise effectively masks the low-rank structure unless explicit measures are taken to preserve it.

6.5 Reconstructing Wiener Structure using Wiener Kernel

This subsection, similar to Section 5.2.2, marks a key intermediate step in the identification framework of this thesis (see Figure 1.6)

Once the Volterra kernel $\boldsymbol{h}$ has been estimated using the TT-format approach, it can be used to extract the block parameters of the underlying Wiener system. This involves identifying the impulse response $\boldsymbol{s}$ of the linear dynamic block and the coefficients $p_1, \ldots, p_D$ of the static nonlinearity.

To estimate the impulse response $\boldsymbol{s}$ of the linear block, we rely on the second-order Wiener kernel $\boldsymbol{H}^{(2)}$:

$$\boldsymbol{H}^{(2)} = \begin{bmatrix} p_2 s_0^2 & p_2 s_0 s_1 & \cdots & p_2 s_0 s_{M_\mathrm{w}} \\ p_2 s_0 s_1 & p_2 s_1^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ p_2 s_0 s_{M_\mathrm{w}} & \cdots & \cdots & p_2 s_{M_\mathrm{w}}^2 \end{bmatrix} = p_2 \cdot \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{M_\mathrm{w}} \end{bmatrix} \begin{bmatrix} s_0 & s_1 & \cdots & s_{M_\mathrm{w}} \end{bmatrix}. \tag{6.35}$$

This is a symmetric, rank-one matrix when $p_2 \neq 0$, reflecting the outer product structure of the linear block's impulse response.

In the **ideal noiseless case**, $\boldsymbol{H}^{(2)}$ is exactly rank-one due to the inherent modularity of Wiener systems, provided $\boldsymbol{s} \neq 0$ and $p_2 \neq 0$. The Volterra linear kernel $\boldsymbol{h}^{(1)}$ equals the impulse response $\boldsymbol{s}$ scaled by p_1. However, in the **noisy case**, the matrix $\boldsymbol{H}^{(2)}$ is typically full-rank but remains approximately low-rank. This low-rank structure enables us to extract the model parameters using SVD as follows:

$$\boldsymbol{H}^{(2)} = \sum_{i=1}^{r} \sigma_i \boldsymbol{\alpha}_i \boldsymbol{\beta}_i^*, \tag{6.36}$$

where $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are the singular vectors, $\boldsymbol{\sigma}_i$ are the singular values, and r is the rank of $\boldsymbol{H}^{(2)}$. The impulse response estimate $\hat{\boldsymbol{s}}$ is identified as the first singular vector corresponding to the largest singular value σ_1. Since the pair of corresponding columns $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ can differ by a sign for a symmetric $\boldsymbol{H}^{(2)}$ (i.e., $\boldsymbol{\alpha}_i = \pm\boldsymbol{\beta}_i$), the impulse response estimate $\hat{\boldsymbol{s}}$ is invariant up to a sign. The coefficient p_2 is then estimated from the first singular value σ_1, and the sign is corrected as follows:

$$\hat{p}_2 = \sigma_1 \boldsymbol{\beta}_1^* \boldsymbol{\alpha}_1. \tag{6.37}$$

To estimate the coefficient p_1, we use the Volterra linear kernel $\boldsymbol{h}^{(1)}$ and the impulse response estimate $\hat{\boldsymbol{s}}$. From (6.4), we observe that $\boldsymbol{h}^{(1)}$ is a scaled version of the impulse response $\boldsymbol{s}$. Given that the estimate $\hat{\boldsymbol{s}}$ is normalised, i.e., using the inner product $\hat{\boldsymbol{s}}^\top \hat{\boldsymbol{s}} = 1$, the coefficient p_1 is estimated as follows:

$$\hat{p}_1 = \hat{\boldsymbol{s}}^\top \boldsymbol{h}^{(1)}. \tag{6.38}$$

Similarly, higher-order coefficients $p_3, p_4, \ldots, p_D$ are extracted from the corresponding higher-order Volterra kernels. For instance, the third-order coefficient p_3 is estimated

using the third-order Volterra kernel $\mathcal{H}^{(3)}$ as follows:

$$\hat{p}_3 = \mathrm{vec}(\mathcal{S})^\top \mathrm{vec}(\mathcal{H}^{(3)}), \tag{6.39}$$

where $\mathcal{S} = \hat{\boldsymbol{s}} \circ \hat{\boldsymbol{s}} \circ \hat{\boldsymbol{s}}$, the operator $\circ$ denotes the outer product (tensor product). The operator $\mathrm{vec}(\cdot)$ vectorises the tensor. Following this approach, the coefficient p_D is derived from the Dth-order kernel.

This approach exploits the low-rank structure of the symmetric Volterra kernel induced by the Wiener model. It provides closed-form, interpretable estimates of the block parameters, which serve as effective initialisation for further refinement. However, in the presence of measurement noise or model mismatch, these estimates may deviate from the true system parameters. Notably, the rank-one assumption of $\boldsymbol{H}^{(2)}$ can be violated by noise, leading to inflated rank and loss of structure. Therefore, to achieve a better approximation of system behaviour, these initial parameters are subsequently refined by solving a (shape-enforced) nonlinear optimisation problem that directly minimises the output prediction error, while preserving the physical characteristics of the impulse response.

6.6 Shape-enforced Optimisation of Wiener Kernel

To enhance the interpretability and robustness of the Wiener model in all intermediate stages of the modelling framework in Figure 1.6, we optionally apply a two-stage shape-regularised optimisation procedure, following the same structure as the Hammerstein case (see Equation (5.25)). This involves refining the initial low-rank estimates $(\hat{\boldsymbol{s}}, \hat{p}_1, \ldots, \hat{p}_D)$ to better align the model output with measurements, while promoting physically meaningful impulse response shapes.

Stage 1: Low-Rank Structure-Constrained Refinement The first stage operates in a compact parameter space informed by the Volterra kernel structure, enforcing smoothness and decay on the impulse response vector $\boldsymbol{s}$. The regularised cost function balances data fit, curvature penalisation, and decay:

$$\min_{\boldsymbol{s},\boldsymbol{p}} J(\boldsymbol{s}, \boldsymbol{p}) = \underbrace{\|\hat{\boldsymbol{y}}(\boldsymbol{u}; \boldsymbol{s}, \boldsymbol{p}) - \boldsymbol{y}\|_2^2}_{\text{data fit}} + \underbrace{\mu \, \|\boldsymbol{w} \odot \boldsymbol{s}\|_2^2}_{\text{decay}} + \underbrace{\lambda \left\|\boldsymbol{\Delta}^2 \boldsymbol{s}\right\|_2^2}_{\text{curvature}}. \tag{6.40}$$

where μ and λ are regularisation weights controlling decay and smoothness, respectively; $\boldsymbol{\Delta}^2$ is the second-order difference operator and $\boldsymbol{w} = \begin{bmatrix} 0 & 1 & \cdots & M_{\mathrm{w}} \end{bmatrix}^\top$ is a linear ramp weighting later taps more heavily but not aggressively, unlike exponential weights. This weighting vector ensures that later lags in the kernel are penalised more strongly while preserving flexibility in the tail to accommodate varying decay patterns across different system dynamics.

This shape-enforced step is optional and particularly beneficial in measurement scenarios with high noise or unknown dynamics. In simulation studies, where the true system is known and initial estimates are already accurate, this refinement may be omitted to reduce training time without compromising model quality. Further refinements in an extended parameter space can follow this step if needed.

Stage 2: Shape-enforced Optimisation In the second step, the refined impulse response and polynomial coefficients are embedded into a longer parameter vector $\boldsymbol{\theta}$ to allow additional flexibility during model fitting. The only difference between the shape enforcement in stage 1 and stage 2 is that in stage 2, we allow the freedom to fit a longer impulse response and higher degree nonlinearity, unlike in stage 1, where the memory and degree of nonlinearity are the same as those used in the Volterra kernel estimation. This step uses the block-oriented model formulation described in Section 5.2.3, adapted for the Wiener structure, as shown below:

$$\Phi\left(\boldsymbol{u}_k, \begin{bmatrix}\boldsymbol{\theta}^{\mathrm{NL}} \\ \boldsymbol{\theta}^{\mathrm{L}}\end{bmatrix}\right) = \sum_{j=1}^{D_+} \hat{p}_j \left(\sum_{i=0}^{M_{\mathrm{w}+}} \hat{s}_i u_{k-i}\right)^j = \sum_{j=1}^{D_+} \theta_j \left(\sum_{i=0}^{M_{\mathrm{w}+}} \theta_{D_+ + 1 + i} u_{k-i}\right)^j, \tag{6.41}$$

where D_+ and $M_{\mathrm{w}+}$ denote extended degree and memory length introduced during optimisation to improve the approximation of the block characteristics. We then solve the regularised cost function (5.25) directly based on the input–output data.

This final step ensures accurate output prediction while continuing to enforce smoothness in the LTI dynamics and compactness in the polynomial representation. The optimised block parameter estimates are subsequently used in the next section to realise the PNLSS form of the Wiener structure, which marks the final stage of the modelling framework.

6.7 Realising PNLSS Model from the optimised Wiener Kernel

The procedure for realising a PNLSS model from the optimised Wiener kernel follows the same principles as in the Hammerstein case (see Section 5.2.4), with the key difference being the placement of the nonlinearity after the linear dynamic block. Since the Wiener model parameters have already been optimised to capture extended memory and nonlinear effects, this step focuses on structurally mapping the block-oriented model into a PNLSS-compatible form using the estimated linear and nonlinear coefficients.

The initial step involves reconstructing a discrete-time LSS model from the optimised impulse response vector $\boldsymbol{\theta}^{\mathrm{L}}$. In the ideal, noiseless conditions, where the LTI block is an FIR model of order $M_{\mathrm{w}+}$ and the SNL has degree D_+, the PNLSS state dimension is taken as $M_{\mathrm{w}-} = M_{\mathrm{w}+}$. For the Wiener case, the following theoretical structure of

the PNLSS parameters (4.43) is obtained by appropriately adjusting the coefficient mappings to reflect the position of the nonlinearity at the output:

$$\boldsymbol{A} = \begin{bmatrix} 0 & \dots & \dots & \dots & 0 \\ 1 & \ddots & & & \vdots \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \boldsymbol{c} = \theta_1^{\mathrm{NL}} \begin{bmatrix} \theta_2^{\mathrm{L}} \\ \vdots \\ \theta_{M_{\mathrm{w}+}}^{\mathrm{L}} \end{bmatrix}^\top, \quad d = \theta_1^{\mathrm{NL}} \theta_1^{\mathrm{L}},$$

$$\boldsymbol{E} = \emptyset, \quad \boldsymbol{f} = \begin{bmatrix} \theta_2^{\mathrm{NL}} (\theta_2^{\mathrm{L}})^2 & 2\theta_2^{\mathrm{NL}} \theta_2^{\mathrm{L}} \theta_3^{\mathrm{L}} & \dots & D_+ \theta_{D_+}^{\mathrm{NL}} \theta_{M_{\mathrm{w}+}}^{\mathrm{L}} (\theta_1^{\mathrm{L}})^{D_+ - 1} & \theta_{D_+}^{\mathrm{NL}} (\theta_1^{\mathrm{L}})^{D_+} \end{bmatrix}, \tag{6.42}$$

In this representation, the matrix $\boldsymbol{A}$ implements a tapped delay line that propagates past inputs through the state. The vector $\boldsymbol{b}$ injects the current input, while the output map $\boldsymbol{c}$ and scalar d capture the linear dynamics prior to the static nonlinearity. The nonlinear term $\boldsymbol{f}$ encodes the contribution of higher-order polynomial terms applied to the linear output. Unlike in the Hammerstein case, the nonlinearity here operates on the output of the dynamic block, which changes the structure of $\boldsymbol{c}$ and $\boldsymbol{f}$. As before, this representation is theoretical, and in practice, a reduced-order approximation using ERA is employed for numerical realisation.

As in the Hammerstein case (see Section 5.2.4), the LTI block in practical Wiener systems may deviate from an FIR assumption, and the static nonlinearity may not be strictly polynomial. In the presence of model mismatch and noise, it is beneficial to realise a reduced-order state-space model with dimension $M_{\mathrm{w}-} \ll M_{\mathrm{w}+}$. For consistency, this work adopts the same approach based on the ERA, rather than more general subspace methods such as N4SID or MOESP.

Employing the same ERA approach outlined previously in Section 5.2.4, Equations (5.27)-(5.35), the LSS parameters $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d)$ can be extracted from the optimised impulse response $\boldsymbol{\theta}^{\mathrm{L}}$, assuming a known system order $M_{\mathrm{w}-}$.

Since the nonlinear mapping in the Wiener structure acts solely at the output (ref. Section 4.5.3.2, Paduart et al. (2007)), the nonlinear state matrix $\boldsymbol{E}$ is omitted. The nonlinear output vector $\boldsymbol{f}$ is then derived using the estimated polynomial coefficients $\boldsymbol{\theta}^{\mathrm{NL}}$, together with the impulse response $\boldsymbol{\theta}^{\mathrm{L}}$, as defined in Equation (4.43).

6.8 Results and Evaluation - Wiener System

This section presents results from both numerical simulations and measurement-based system identification for Wiener systems, following a similar approach to the one

Algorithm 6 From Volterra kernel to state-space model: a structured identification of a Wiener system

Input: Input-output data $\{u_k, y_k\}_{k=1}^{N}$
Output: PNLSS Parameters $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{f}\}$

1: Choose Volterra memory M_{w} and order D (ensure $N \geq \binom{D+I-1}{I-1}$ (cf. Theorem 3), $I = M_{\mathrm{w}} + 2$)
2: Build matrix $\tilde{\boldsymbol{U}} \in \mathbb{R}^{N \times I}$, as shown in (6.22)
3: Construct TTM of the repeated row-wise Kronecker product of $\tilde{\boldsymbol{U}}$, as in Algorithm 4
4: Compute thin SVD of $\boldsymbol{U}$ in TTM form, as in Algorithm 5
5: Estimate Volterra kernel $\tilde{\boldsymbol{h}} \in \mathbb{R}^{I^D}$ (see (6.30))
6: Reshape the kernel $\tilde{\boldsymbol{h}}$ into a D-way tensor and extract the block-oriented model parameters estimates $\{\hat{\boldsymbol{s}}, p_1, \ldots, p_D\}$ using rank-one approximation of the kernel, as given in (6.35)-(6.39)
7: Optionally regularise/extend ($M_{\mathrm{w}+} > M_{\mathrm{w}}$, $D_+ > D$) using (6.40)–(6.41) to get an extended impulse response
8: Record the feedback coefficient d from the first coefficient of impulse response
9: Form Hankel matrices $\boldsymbol{H}_0, \boldsymbol{H}_1$ (size $r \times s$, $r+s = M_{\mathrm{h}+}$, choose r, s with $r, s \geq M_{\mathrm{h}-}$) with the remaining impulse response vector. Select model order $n = M_{\mathrm{h}-}$ from singular-energy criterion (90–95%)
10: Perform SVD on $\boldsymbol{H}_0$, compute $\boldsymbol{\Gamma}_r, \boldsymbol{\Delta}_s$ (cf. (5.34)) and recover linear state-space estimates $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}\}$ via ERA formulas ((5.35))
11: Estimate nonlinear output map $\boldsymbol{f}$ from $\boldsymbol{c}, d$ and polynomial coefficients $\theta_2^{\mathrm{NL}}, \ldots, \theta_{D_+}^{\mathrm{NL}}$
12: Return $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{f}\}$

outlined for Hammerstein systems in the previous chapter. In addition to systems that conform to the Wiener structure, we also evaluate the performance of the proposed modelling algorithm on a selection of benchmark process industry systems, including pH neutralisation, heat exchangers, and continuous stirred-tank reactors (CSTRs). These systems do not necessarily adhere to a strict Wiener block structure, but are included to assess the flexibility and generalisation capability of the framework when applied to more complex or loosely structured nonlinear dynamics.

6.8.1 Numerical Simulations

Similar to the numerical experiments conducted on the simulated Hammerstein systems in Section 5.3.1.1, the Wiener system used here consists of a third-order low-pass filter as the LTI block, with a passband frequency of 0.5 Hz, a passband ripple of 0.5 dB, and a stopband attenuation of 20 dB. The SNL block is chosen to be either a polynomial, dead zone or sigmoid function. The parameters of the polynomial and

sigmoid nonlinearities are identical to those used in Section 5.3.1.1, while the range of dead zone nonlinearity has been widened from $[-0.3, 0.3]$ to $[-0.75, 0.75]$. All other simulation settings, including the input signal, noise characteristics, and performance metrics, remain consistent with those described previously.

The input signal, $\boldsymbol{u} \sim \mathcal{N}(0, 1)$, is passed through the simulated Wiener system. The output, $\boldsymbol{y}$, is subsequently corrupted by WGN at varying SNR levels. A total of $N = 8{,}000$ samples are used for training, with $2{,}000$ unseen samples allocated to each of the validation and test sets, as illustrated in Figure 6.9.

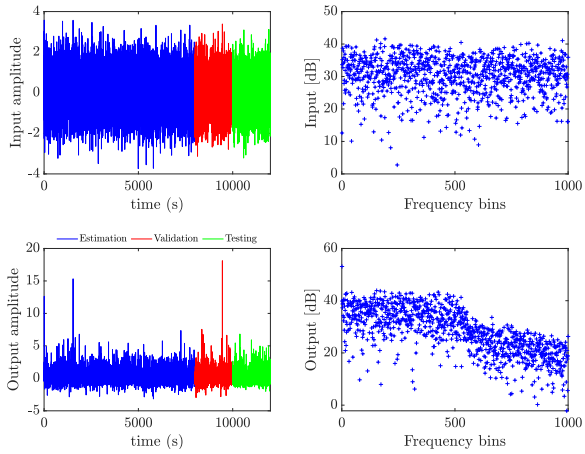


Figure 6.9: Simulation dataset showing the training, validation, and test splits. The Gaussian input excites a broad frequency range. The output exhibits dominant low-frequency components as well as residual higher-frequency content due to nonlinear distortion.

Since the nonlinearity follows the linear filter in a Wiener structure, the system output is shaped by both the low-pass dynamics and the nonlinear transformation. Although the linear filter attenuates high-frequency components, the static nonlinearity introduces frequency mixing and distortion, resulting in output signals that retain not only strong low-frequency content but also contain higher-frequency components, particularly harmonics and intermodulation products. This contrasts with the Hammerstein system, where the nonlinearity precedes the linear block and thus the output spectrum remains strictly low-pass filtered.

Table 6.1 reports the identification performance of Wiener systems with three types of nonlinearities, under varying SNR conditions. It includes output prediction errors (RRMSE_1), and relative parameter errors in the linear (RRMSE_2) and

nonlinear (RRMSE_3) blocks, over 40 Monte Carlo runs. The model IDs used in this table follow the same convention as those defined earlier in Table 5.1 (Section 5.3.1.2).

In Table 6.1.A, for the polynomial nonlinearity, the block-oriented model P3 and its corresponding PNLSS model P4 achieve the highest output prediction accuracy, with RRMSE_1 values as low as 0.7–2.4% even at an SNR of 5 dB. The block-oriented model also shows excellent recovery of the LTI and nonlinear parameters, reflected in low RRMSE_2 and RRMSE_3 values. However, this performance comes at the cost of higher training time due to solving a large Volterra regression problem, with matrix $\boldsymbol{U} \in \mathbb{R}^{8000\times 2023}$. The baseline PNLSS model B4 also performs well in terms of output prediction, but it lacks the interpretability offered by the block-oriented structure. Another baseline, the BLA-based model B2, exhibits higher prediction errors compared to P4 and B4.

The dead zone nonlinearity introduces discontinuities and non-smooth behaviour, while the sigmoid nonlinearity introduces smooth saturation effects, making identification more challenging compared to the polynomial case. Tables 6.1.B and C show performance of the models under slightly higher SNR conditions than those used in the polynomial nonlinearity case (Table 6.1.A). This adjustment was necessary because, in the context of Wiener system identification, approximating complex nonlinearities like the dead zone and sigmoid functions using a third-order polynomial led to higher errors in some Monte Carlo trials, thereby increasing the average RRMSE. As a result, higher SNR levels were used to ensure meaningful identification results for the given hyperparameters M and D.

In Table 6.1.B, model P3 demonstrates consistently strong parameter estimation across all 40 trials, which enables the corresponding PNLSS model P4 to outperform the baselines B2 and B4 in terms of prediction accuracy. In Table 6.1.C, all PNLSS models (proposed and baselines) show comparable performance. However, model P4 requires significantly more computation time, primarily due to the time spent solving the Volterra regression problem associated with a dense Wiener kernel.

Figure 6.10 compares the output prediction error (RRMSE_1) across six modelling approaches for three different nonlinearities, evaluated at SNR_2 (see Table 6.1 for SNR values). The top row presents the mean errors on a logarithmic scale to emphasise relative differences, while the bottom row shows the distribution of errors across 40 Monte Carlo runs using violin plots (clipped at the 99th percentile to reduce the influence of outliers). Across all three nonlinearity types, the PNLSS model P4, initialised using the Volterra-based block-oriented model, consistently outperforms the baseline models B2 and B4 in terms of lower mean RRMSE_1. Additionally, all models. including the baselines B2 and B4, exhibit relatively low variability across Monte Carlo runs, as reflected in the compact shape of the violin plots and the alignment of the median (indicated by the black horizontal lines). Nevertheless, model P4 consistently achieves lower median and mean errors across all three nonlinearities, indicating that

Table 6.1: Performance on Simulated Test Dataset with 3 Types of Nonlinearity - Wiener System

Methodology Details	M	D	No. of Parameters	RRMSE_1 (%)			RRMSE_2 in the LTI block (%)			RRMSE_3 in the SNL block (%)			$\bar{\mu}$			$\bar{\sigma}$			Training time (s)
				No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	
A. Polynomial nonlinear block - Training sample: $N = 8000$, Test sample: 2000, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.98$, $N_{\mathrm{runs}} = 40$, SNR_1 = 20 dB, SNR_2 = 5 dB																			
Volterra	20	3	2023	1.34	6.03	33.14	0.05	0.16	0.80	–	–	–	2e-4	8e-4	4e-3	0.01	0.06	0.32	214
BS_Volterra	20	3	24	0.16	1.16	6.64	0.05	0.08	0.37	0.10	0.75	4.14	8e-5	-3e-4	-1e-3	2e-3	0.01	0.07	215
BS_Volterra_LM	48	3	52	1e-4	0.32	1.85	**1e-4**	**0.03**	**0.18**	**0.03**	**0.33**	**1.86**	1e-8	-1e-4	-3e-4	1e-6	3e-3	0.02	218
PNLSS_BS_Volterra_LM	3	3	46	**0.74**	**0.85**	**2.38**	–	–	–	–	–	–	-4e-5	-4e-4	-3e-3	7e-3	8e-3	0.02	218
BLA	3	–	16	43.87	43.89	49.98	–	–	–	–	–	–	0.22	0.22	0.23	0.37	0.37	0.43	172
PNLSS_BLA_LM	3	3	46	21.07	15.75	26.19	–	–	–	–	–	–	5e-3	5e-3	0.01	0.21	0.15	0.26	192
N4SID	3	–	16	38.60	38.60	38.64	–	–	–	–	–	–	0.20	0.20	0.20	0.32	0.32	0.32	1
PNLSS_N4SID_LM	3	3	46	**1.42**	**1.55**	**3.82**	–	–	–	–	–	–	1e-4	2e-4	2e-3	0.01	0.02	0.04	11
B. Dead zone nonlinear block $[-0.75, 0.75]$ - Training sample, $N = 8000$, Test sample: 2000, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.24$, $N_{\mathrm{runs}} = 40$, SNR_1 = 30 dB, SNR_2 = 20 dB																			
Volterra	20	3	2023	32.58	32.63	33.11	1.84	1.82	1.85	–	–	–	-2e-3	-2e-3	-1e-3	0.08	0.08	0.08	217
BS_Volterra	20	3	24	75.93	75.78	76.28	2.70	2.70	2.74	–	–	–	6e-3	6e-3	6e-3	0.18	0.18	0.19	218
BS_Volterra_LM	48	3	52	26.13	26.13	26.12	**0.34**	**0.34**	**0.34**	–	–	–	2e-3	2e-3	2e-3	0.06	0.06	0.06	221
PNLSS_BS_Volterra_LM	3	3	46	**35.55**	**36.24**	**37.88**	–	–	–	–	–	–	2e-3	2e-3	2e-3	0.09	0.09	0.09	221
BLA	3	–	16	71.27	71.30	71.39	–	–	–	–	–	–	3e-3	3e-3	3e-3	0.17	0.17	0.17	141
PNLSS_BLA_LM	3	3	46	63.92	63.93	63.94	–	–	–	–	–	–	3e-3	3e-3	3e-3	0.16	0.16	0.16	157
N4SID	3	–	16	61.05	61.06	61.06	–	–	–	–	–	–	3e-3	3e-3	3e-3	0.15	0.15	0.15	0.2
PNLSS_N4SID_LM	3	3	46	44.01	44.00	43.98	–	–	–	–	–	–	2e-3	2e-3	2e-3	0.11	0.11	0.11	10
C. Sigmoid function nonlinear block - Training sample, $N = 8000$, Test sample: 2000, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.26$, $N_{\mathrm{runs}} = 40$, SNR_1 = 40 dB, SNR_2 = 30 dB																			
Volterra	20	3	2023	1.47	1.57	2.18	**0.05**	**0.05**	**0.06**	–	–	–	-4e-5	-5e-6	6e-5	4e-3	4e-3	6e-3	198
BS_Volterra	20	3	24	36.24	43.91	71.57	2.59	2.87	4.18	–	–	–	-1e-3	-2e-3	-3e-3	0.10	0.12	0.19	198
BS_Volterra_LM	48	3	52	0.60	0.60	0.60	**2e-3**	**4e-3**	**0.01**	–	–	–	-2e-4	-2e-4	-2e-4	2e-3	2e-3	2e-3	201
PNLSS_BS_Volterra_LM	3	3	46	**0.91**	**0.91**	**0.85**	–	–	–	–	–	–	-2e-4	-2e-4	-2e-4	2e-3	2e-3	2e-3	201
BLA	3	–	16	4.77	4.76	4.76	–	–	–	–	–	–	-5e-4	-5e-4	-5e-4	0.01	0.01	0.01	64
PNLSS_BLA_LM	3	3	46	**1.03**	**1.03**	**1.05**	–	–	–	–	–	–	-2e-4	-2e-4	-1e-4	3e-3	3e-3	3e-3	72
N4SID	3	–	16	4.71	4.71	4.71	–	–	–	–	–	–	-5e-4	-5e-4	-5e-4	0.01	0.01	0.01	0.4
PNLSS_N4SID_LM	3	3	46	**1.03**	**1.03**	**1.05**	–	–	–	–	–	–	-2e-4	-2e-4	-1e-4	3e-3	3e-3	3e-3	8

the proposed Volterra-based initialisation improves the overall accuracy of the identified PNLSS models under moderate noise conditions.

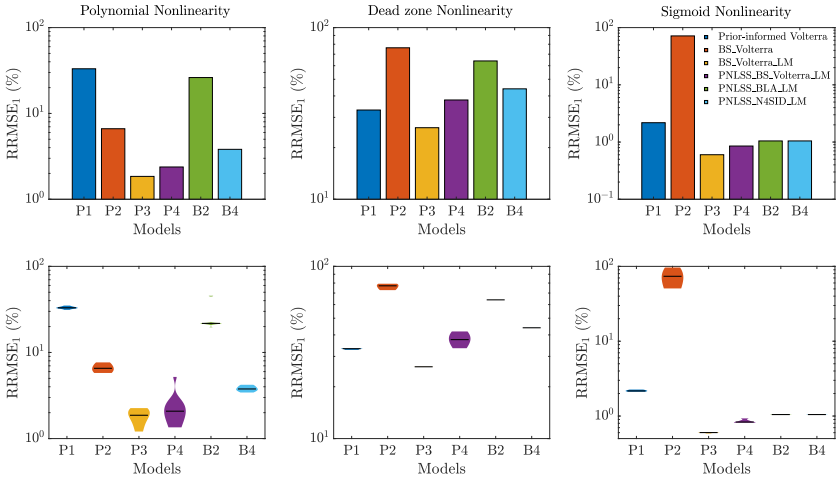


Figure 6.10: Comparison of output prediction errors (RRMSE_1 %) across six modelling approaches for Wiener systems with three different nonlinearities, evaluated at SNR_2. The top row shows the mean errors on a logarithmic scale, while the bottom row displays the distribution of errors over 40 Monte Carlo runs using violin plots (clipped at the 99th percentile). PNLSS model P4, initialised using the proposed Volterra-based method, achieves consistently lower prediction error compared to baseline models B2 and B4.

ANOVA - Report: A one-way ANOVA was performed to compare the performance of the three PNLSS models for the Wiener system under SNR_1 (high), and SNR_2 (low) conditions and the results are reported in Table 6.2. Across all nonlinearities and SNR levels, the p-values were extremely small ($p \leq 1e^{-7}$), indicating statistically significant differences in model performance. The corresponding effect sizes were consistently strong ($\eta^2 > 0.65$), suggesting that the choice of identification method has a substantial impact on modelling accuracy regardless of noise level. These results are further supported by the box plots in Figure 6.11 and Tukey–Kramer multiple comparison plots in Figure 6.12, which clearly distinguish the proposed model from the literature-based alternatives.

Figure 6.13 highlights the parameter estimation performance of models P1 to P3 for each nonlinearity. The top row shows mean relative errors for LTI and SNL blocks ($\text{RRMSE}_{2,3}$) as bar plots, and the bottom row shows their distributions across trials using violin plots. In the polynomial case, this is supported by Figure 6.14, which compares true and estimated poles/zeros (left), polynomial coefficients (centre) and the

Table 6.2: ANOVA Table - Wiener simulation dataset

Nonlinearity	p-value		Star rating		η^2		Effect size interpretation	
	SNR_1	SNR_2	SNR_1	SNR_2	SNR_1	SNR_2	SNR_1	SNR_2
Polynomial	$1e^{-7}$	$1e^{-8}$	***	***	0.67	0.79	Strong	Strong
Dead zone	$1e^{-9}$	$1e^{-8}$	***	***	0.99	0.98	Strong	Strong
Sigmoid	$1e^{-9}$	$1e^{-7}$	***	***	0.98	0.93	Strong	Strong

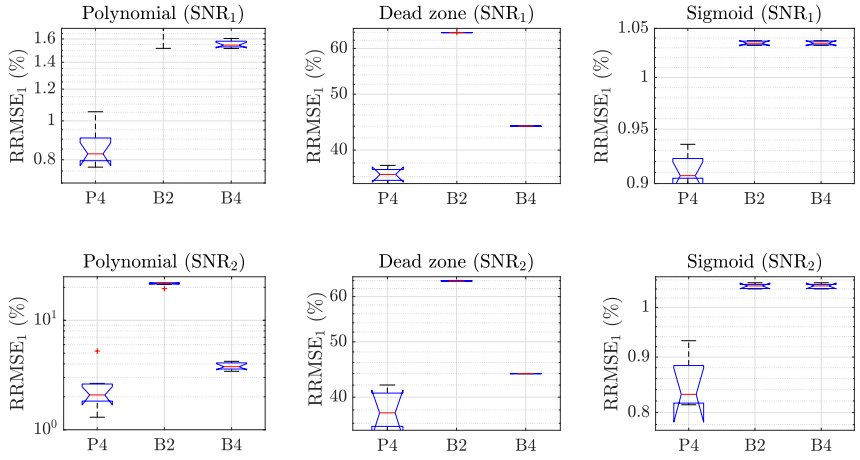


Figure 6.11: Box plots of RRMSE (%) for the three PNLSS models under different nonlinearities and SNR levels for the Wiener system. Each subplot corresponds to a specific nonlinearity–SNR combination. The boxes represent the distribution of model errors obtained from 40 Monte Carlo runs, illustrating both the median and spread of the RRMSE values. Lower median and narrower boxes indicate better and more consistent model performance across noise realisations.

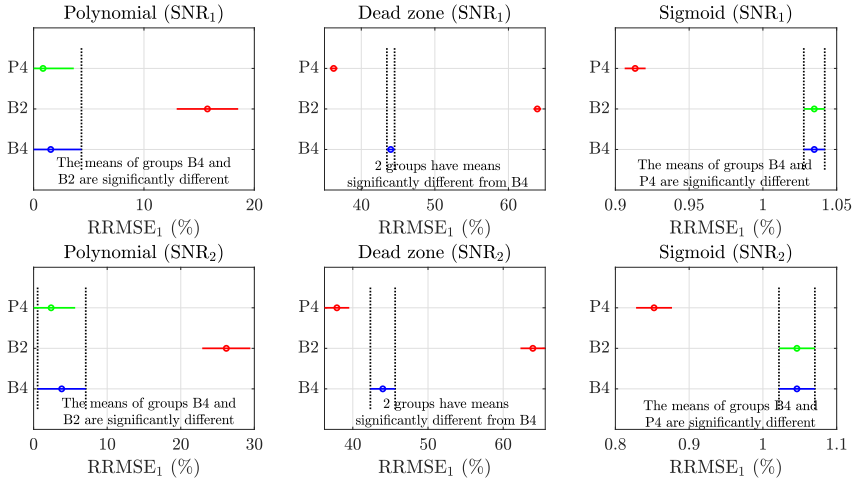


Figure 6.12: Multiple comparison results (Tukey–Kramer test) for pairwise mean differences in RRMSE among the three PNLSS models under different nonlinearities and SNR levels for the Wiener system. Each subplot corresponds to a specific nonlinearity–SNR combination. The blue marker represents the fixed reference model (B4), while green markers indicate models whose mean RRMSE lies within the 95% confidence interval of the reference model, implying statistically similar performance. Red markers denote models whose mean differs significantly from the reference, indicating statistically inferior or superior performance relative to B4.

Bode magnitude and phase plots (right). The spread inthe Volterra model P1 indicates high variance in initial parameter estimates, which improves significantly after local refinement in model P3. The averaged Bode plot confirms accurate recovery of LTI characteristics.

Figure 6.15 compares model complexity and training time. The proposed framework requires more time, primarily due to the Volterra kernel estimation step, where no structural prior is available (unlike the Hammerstein case). Although model complexity reduces in intermediate and final PNLSS stages, training time remains a limitation for the Wiener case.

Finally, Figure 6.16 shows validation and test set prediction errors for the polynomial case. On the left, validation error per iteration is plotted for 40 trials, showing that Volterra-based initialisations often start closer to the optimum. On the right, test set prediction errors for the best candidate models confirm that all PNLSS variants perform comparably in this setting, consistent with results in Table 6.1.A.

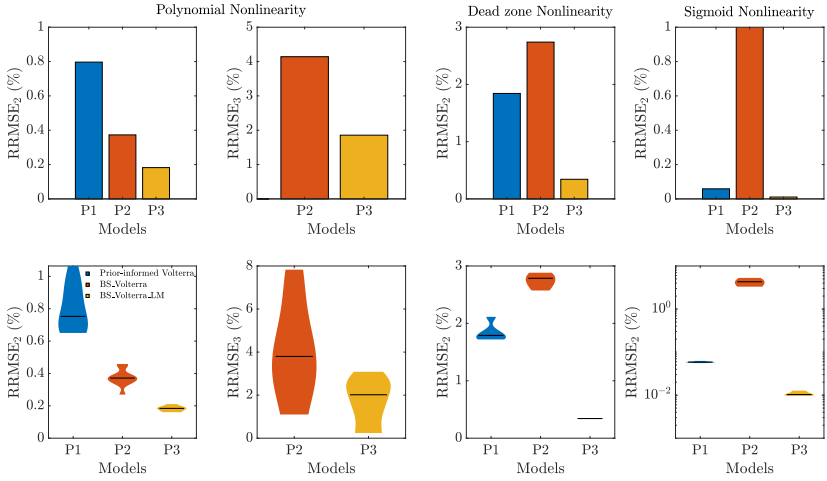


Figure 6.13: Relative parameter errors ($RRMSE_{2,3}$ %) of the LTI and SNL blocks for three types of nonlinearities at SNR_2. Top row: Mean errors. Bottom row: Error distributions across 40 runs. The model P2 and its LM extension P3 show consistent performance and tight error distributions.

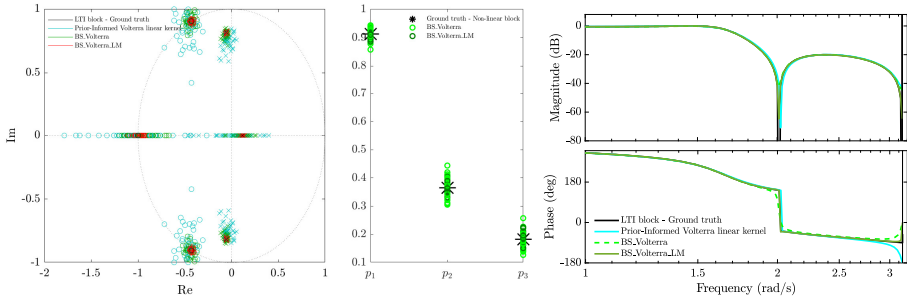


Figure 6.14: Pole-zero map and SNL block coefficients for the Wiener system with polynomial nonlinearity (SNR 5 dB). The Bode plot on the right shows averaged results over 40 trials.

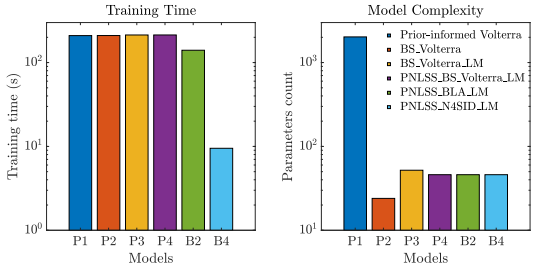


Figure 6.15: Training time and model complexity comparison for various PNLSS modelling strategies. High parameter count in the Volterra kernel leads to significantly increased training time in the Wiener system.

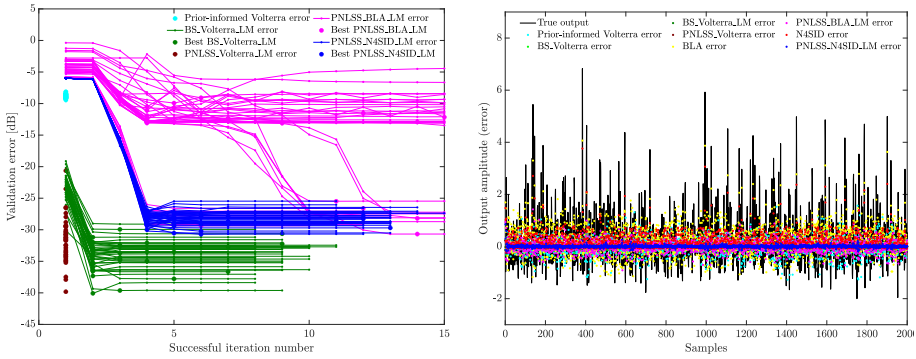


Figure 6.16: Left: Validation errors across iterations for 40 trials (polynomial nonlinearity, SNR 5 dB). Right: Test set prediction errors. PNLSS models initialised from Volterra and N4SID (P4 and B4) achieve similar final performance.

6.8.2 Experimental Validation

This section evaluates the proposed identification framework on experimental data collected from a nonlinear electronic system configured in a Wiener structure. The measurement setup, detailed in Section 5.3.2, remains the same: a nonlinear system with a WH configuration, selectively observed here at a point that yields a Wiener structure, i.e., the output is taken directly after the SNL block (see Figure 6.17).

The system again consists of a third-order Chebyshev (Type I) low-pass filter preceding a diode-based static nonlinearity (see Figure 5.16), as discussed in Section 5.3.2. The excitation is a multisine input with selective excitation of odd frequency bins, and measurements were acquired at a sampling rate of 156 kHz. The resulting dataset, shown in Figure 6.18, is split into training (N = 31,256), validation, and test (7,814

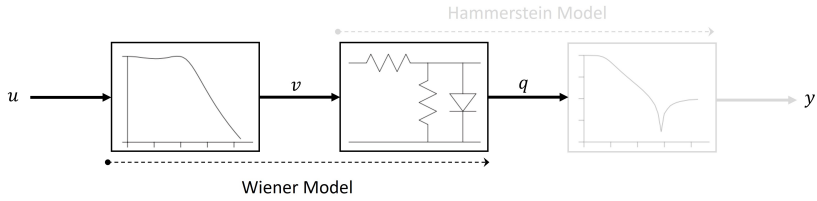


Figure 6.17: Measurement setup of Wiener system. The output is taken after the static nonlinearity, forming a Wiener configuration.

samples each) sets.

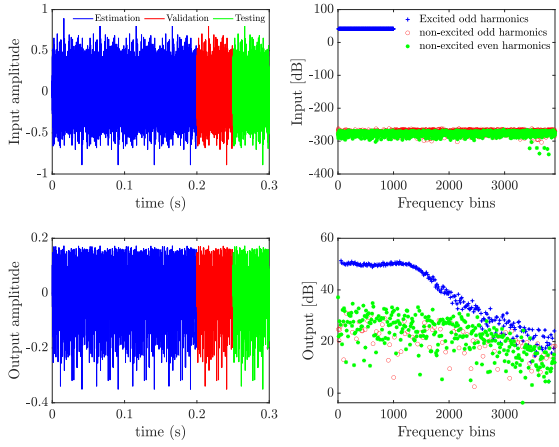


Figure 6.18: Training, validation, and test splits of the Wiener system measurement dataset. The input excites selected odd frequency bins, and the output, taken after the diode nonlinearity, retains frequency-distorted components due to static nonlinear transformation.

Table 6.3 summarises the identification results across various model structures for additive noise of SNR 25 dB. The best performance is achieved by the baseline models B2 and B4 with an RRMSE of approximately 4.5%, followed closely by the block-oriented model P3 (5.35%). The PNLSS model P4 realised from this block-oriented structure shows 14% relative RMS error. In contrast to the Hammerstein case, the Volterra model here required significantly more parameters to achieve comparable accuracy (8.34%). This is due to the dense structure of the Wiener kernel.

Figure 6.19 compares the estimated filter frequency responses with the analytic Chebyshev filter. Unlike the Hammerstein case, where shape enforcement significantly

Table 6.3: Performance on the Measurement Dataset

Wiener System (SNR 25 dB, Training sample: $N = 31{,}256$, Test sample: $7{,}814$, $\mathrm{RMS}(y_{\mathrm{test}}) = 94.5$ mV)

Methodology Details	M	D	No. of Parameters	RRMSE_1 (%)	$\bar{\mu}$ (mV)	$\bar{\sigma}$ (mV)	Training time (s)
Volterra	40	3	13243	8.34	3.63	7.01	2215
BS_Volterra	40	3	44	17.72	-2.80	16.52	4130
BS_Volterra_LM	81	3	85	5.35	2.74	4.25	4238
PNLSS_BS_Volterra_LM	3	3	46	13.78	-0.47	13.01	4239
BLA	3	–	16	14.53	-2.89	13.43	10
PNLSS_BLA_LM	3	3	46	**4.47**	1.38	4.00	59
N4SID	3	–	16	14.78	-2.93	13.65	1
PNLSS_N4SID_LM	3	3	46	**4.63**	1.70	4.03	61

improved fidelity, the Chebyshev filter structure is more naturally captured here due to the LTI block being placed before the nonlinearity. However, the model without shape enforcement shows significant deviation in the stopband. Figure 6.20 illustrates the shape-enforced Wiener kernel up to third order, demonstrating how the optimisation promotes smoothness and decay, resulting in diminishing significance of distant kernel entries.

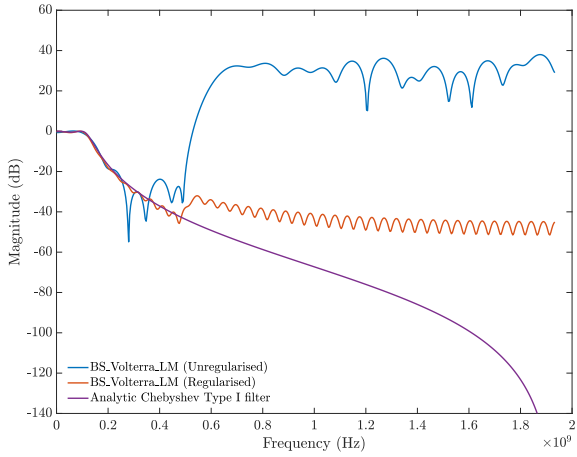


Figure 6.19: FRF of an analytic 3rd-order Chebyshev low-pass filter with a cutoff frequency of 4.4 kHz and a passband ripple of 0.5 dB compared with the estimated FRFs of Block-structured models (with and without shape-enforced optimisation) at SNR 25 dB. The shape enforcement promotes smoothness and decay of the kernel, resulting in approximating the true impulse response characteristics (refer to Section 6.6).

The output prediction accuracy and computational load of PNLSS models are compared

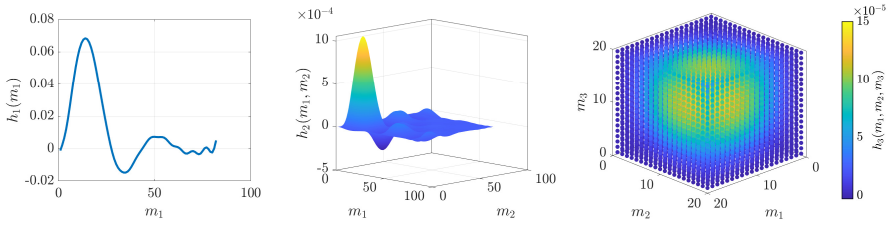


Figure 6.20: Visualisation of the shape-enforced Wiener kernel: First-, second-, and third-order components are shown separately. The third-order kernel is zoomed in on the first 20 lags to highlight its dominant region; distant entries are less significant (refer to Section 6.6).

in Figure 6.21. The proposed PNLSS model P4, initialised using the Volterra-based approach, achieves higher prediction accuracy (RRMSE = 14%) compared to the baselines, albeit with significantly longer training time. Notably, the parameter count remains the same across all models. The increased computational cost arises from the absence of *prior-informed* kernel simplification in the initial Volterra regression step. In contrast, the baseline PNLSS models B2 and B4, initialised using linear methods such as N4SID and BLA, achieve competitive performance with considerably shorter training times.

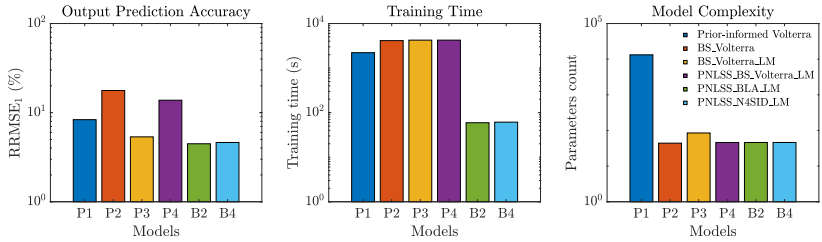


Figure 6.21: RRMSE$_1$ (%) of output prediction at SNR 25 dB. PNLSS models B2 and B4 initialised from N4SID/BLA outperform the Volterra-based PNLSS P2 in terms of efficiency and output prediction accuracy.

The evolution of validation error during LM optimisation (Figure 6.22, left) reveals that Volterra-initialised block-oriented model P3 starts with a higher initial error compared to linear-initialised models B2 and B4. However, P3 converges effectively during optimisation and reaches a final validation error comparable to those of the baseline models. In contrast, the PNLSS model P4, which is realised from P3, does not retain the same level of accuracy after transformation; it settles at a slightly higher error, closer to the performance of the linear-initialised models. On the right, the test set

prediction error confirms that all optimised PNLSS models generalise well, with B2 and B4 continuing to yield the lowest error. This suggests that while Volterra-based initialisation can converge well at the block-oriented model stage, the subsequent transformation into a PNLSS model introduces additional approximation error.

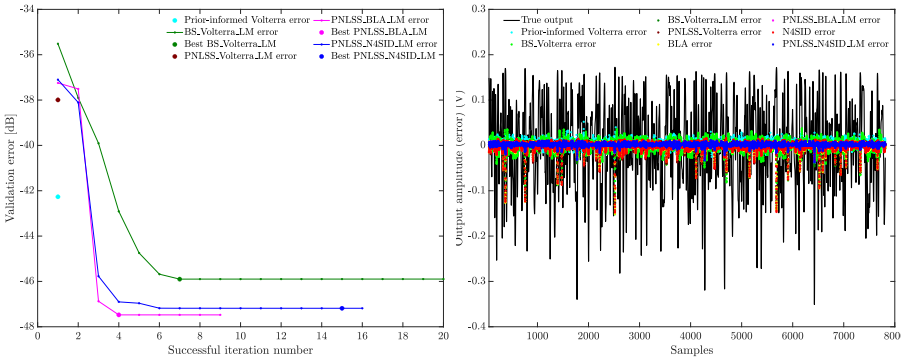


Figure 6.22: Left: Validation errors over LM optimisation iterations for the Wiener system. Although the Volterra-based block-oriented model P3 show optimal performance, the PNLSS realisation P4 introduces approximation error. Right: Output prediction error for SNR 25 dB on test set.

6.8.3 Nonlinear Dynamical Systems beyond Block-oriented Structures

To further evaluate the generality and robustness of the proposed modelling framework, we test it on a set of complex, highly nonlinear benchmark datasets from the Database for the Identification of Systems (DaISy) (De Moor, 2000). These systems are classified as general nonlinear systems in the sense that their underlying dynamics do not explicitly conform to standard block-oriented structures such as Hammerstein, Wiener, or WH models.

The selected case studies originate from the process industry and include a pH neutralisation process, a liquid-saturated steam heat exchanger, and a continuous stirred-tank reactor (CSTR). These systems are well-known in the system identification literature for their strong nonlinearities and dynamic coupling. While some studies have modelled these applications using simplified block-oriented assumptions (Gómez and Jutan, 2003; Zou et al., 2013; Jencia et al., 2024; Garcés et al., 2019; Han et al., 2017; Hong and Cheng, 2011), such structural priors may not accurately reflect the actual physical or measurement setup, particularly when unmodelled dynamics or sensor nonlinearities are present.

In this thesis, each of these systems is modelled using a Wiener structure. This choice is motivated by the absence of precise prior knowledge about the nonlinear kernel and the need for a flexible framework capable of capturing complex input–output behaviour. The Wiener model offers a balance between structural interpretability and functional flexibility, allowing the identification algorithm to approximate the dynamics more accurately without imposing restrictive assumptions.

The following subsections present detailed results of the proposed algorithm applied to each of the three datasets, highlighting both its accuracy and adaptability in modelling nonlinear dynamical systems that lie beyond conventional block-oriented model formulations, with the RMSEs between the true outputs and the outputs predicted by various modelling approaches are reported in Tables 6.4-6.6.

6.8.3.1 pH Neutralisation Process in a Stirring Tank

This case study evaluates the performance of the Volterra kernel-based PNLSS model P4 on a pH neutralisation process characterised by a nonlinear measurement function. The system consists of a 1100-litre (constant volume) stirred tank where a neutralisation reaction occurs between a strong acid and a strong base. Experimental data, obtained from this setup, are assumed to follow a Wiener model structure.

The output of interest is the pH of the effluent solution, which exhibits strong nonlinearity, particularly in the pH range of 4 to 6, as previously studied in Romanenko et al. (2004). In this dataset, the pH varies between 3.69 and 11.79, and measurements are sampled at a rate of 1 second.

This is a multi-input single-output (MISO) system, where the inputs are the flow rates (in litres per second) of acid and base solutions, and the output is the pH level of the resulting solution. The acid and base concentrations are 0.0032 mol/l and 0.05 mol/l, respectively. Given that the base flow has a dominant effect on the neutralisation dynamics, the acid flow is held constant throughout the identification process.

A total of $2{,}001$ samples are available, which are partitioned into training ($N = 801$), validation (600 samples), and test (600 samples) sets. Each subset includes portions of the pH neutralisation zone, ensuring that the dataset represents the full dynamics of the neutralisation process, including the highly nonlinear transition region between acidic and basic pH levels.

The RMSE between the true pH values and those predicted by various modelling approaches are reported in Table 6.4. Among the methods evaluated, PNLSS models P4 and B4 demonstrate nearly identical performance, achieving the lowest RMSEs in the range of 1.4, with RRMSEs in $\approx 50\%$. This confirms the consistency and robustness of the proposed PNLSS modelling framework. In comparison, the baseline model

B2 yields a slightly higher pH prediction error of 1.9. Although the absolute RMSE difference between models P4/B4 and B2 is only 0.5, this corresponds to a relative error increase from 50% to 68%. Given the highly sensitive and nonlinear nature of the pH scale, especially near the neutralisation point, such differences can have a substantial impact on control performance and system safety.

Table 6.4: Performance on Process Industry System I - Wiener Model

pH Neutralisation Process in a Stirring Tank (Training sample: $N = 801$, Test sample: 600, $\mathrm{RMS}(y_{\mathrm{test}}) = 2.79$)

Methodology Details	M	D	No. of Parameters	RMSE (no unit) ($\mathrm{RRMSE}_1\%$)	$\bar{\mu}$ (no unit)	$\bar{\sigma}$ (no unit)	Training time (s)
Volterra	7	3	164	1.77 (63.33)	-0.06	1.78	0.3
BS_Volterra	7	3	11	1.82 (64.94)	0.87	1.61	3
BS_Volterra_LM	15	3	19	1.32 (47.35)	0.35	1.28	6
PNLSS_BS_Volterra_LM	3	3	46	**1.40 (50.02)**	0.47	1.32	6
BLA	3	–	16	2.11 (75.44)	-0.16	2.11	1
PNLSS_BLA_LM	3	3	46	1.90 (67.99)	-0.10	1.90	3
N4SID	3	–	16	1.84 (65.75)	-0.16	1.83	0.2
PNLSS_N4SID_LM	3	3	46	**1.42 (50.84)**	0.10	1.42	3

This trend is further visualised in Figure 6.23, which shows the predicted and true pH trajectories. The left panel shows the full test output; the middle panel provides a zoomed-in view near the acid-base mixing region; and the right panel focuses on the neutralisation zone. Across all views, the PNLSS models P4 and B4 exhibit nearly indistinguishable tracking performance, supporting the quantitative results in Table 6.4. In contrast, the baseline model B2 exhibits visible deviations, particularly in the mixing zone, where the system exhibits highly nonlinear behaviour, while still providing a reasonable approximation near the neutralisation zone.

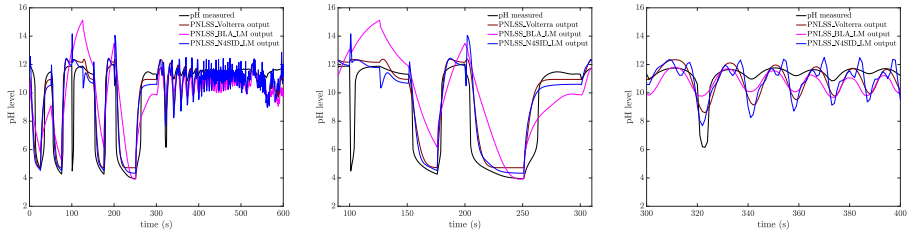


Figure 6.23: Predicted vs. true pH values for the test set in the pH neutralisation process. Left: full trajectory; middle: zoomed-in view near the acid-base mixing region; right: zoomed-in view around the neutralisation zone. The PNLSS models P2 and B4 exhibit nearly identical tracking accuracy across all regions.

6.8.3.2 Liquid-Saturated Steam Heat Exchanger

This case study investigates the performance of the Volterra kernel-based PNLSS model P4 on a heat exchanger process involving liquid heating by saturated steam. The system consists of a copper-tube heat exchanger where water is heated by pressurised saturated steam. The output of interest is the outlet liquid temperature, while the input considered for identification is the liquid flow rate. The steam temperature and inlet liquid temperature are held constant at their nominal values throughout the experiment.

The dataset, contributed by Bittanti and Piroddi (1997), is well-known in the nonlinear system identification literature. It is particularly relevant for nonlinear control design due to its non-minimum phase behaviour, which poses challenges for both identification and control. The data consists of $4{,}000$ samples collected at a sampling interval of 1 second.

In this work, we consider a SISO setting by varying only the liquid flow rate as the input, with the outlet liquid temperature as the measured output. A total of $4{,}000$samples are used, with 3000 samples allocated for training and 500 each for validation and testing.

The RMSE between the true and predicted outlet temperatures, obtained from various modelling approaches, is summarised in Table 6.5. Among all evaluated methods, the PNLSS baseline models B2 and B4 yield the best performance, with RMSEs of 0.24 deg C (corresponding to relative error below 18%). Interestingly, the direct PNLSS model P4 initialised from the block-structured Volterra model performs similarly (RMSE = 0.41 deg C), but does not outperform the linearly initialised PNLSS variants in this case.

Table 6.5: Performance on Process Industry System II - Wiener Model

Liquid-Saturated Steam Heat Exchanger (Training sample: $N = 3{,}000$, Test sample: 500, $\mathrm{RMS}(y_{\mathrm{test}}) = 1.37$ deg C)

Methodology Details	M	D	No. of Parameters	RMSE (deg C) ($\mathrm{RRMSE}_1\%$)	$\bar{\mu}$ (deg C)	$\bar{\sigma}$ (deg C)	Training time (s)
Volterra	10	3	363	0.17 (12.56)	-0.02	0.17	0.3
BS_Volterra	10	3	14	0.49 (35.76)	-0.32	0.37	3
BS_Volterra_LM	21	3	25	0.41 (30.23)	-0.23	0.34	6
PNLSS_BS_Volterra_LM	3	3	46	0.41 (29.80)	-0.23	0.34	6
BLA	3	–	16	0.49 (36.14)	-0.14	0.48	1
PNLSS_BLA_LM	3	3	46	**0.24 (17.57)**	-0.03	0.24	3.4
N4SID	3	–	16	0.39 (28.27)	-0.18	0.34	0.2
PNLSS_N4SID_LM	3	3	46	**0.24 (17.61)**	-0.03	0.24	3

This performance trend is illustrated in Figure 6.24, which shows the predicted and true outlet temperatures over the test set. The PNLSS-based models closely follow the true output, including during dynamic transitions, which supports the numerical results shown in the table. These findings confirm the effectiveness of PNLSS models even in

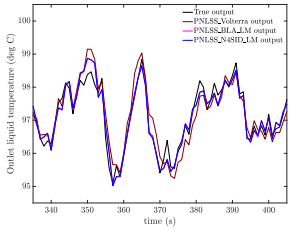


Figure 6.24: Predicted vs. true outlet liquid temperature for the test set in the liquid-saturated steam heat exchanger.

systems with non-minimum phase dynamics, as originally emphasised in Bittanti and Piroddi (1997).

6.8.3.3 Continuous Stirred Tank Reactor

This case study examines the performance of the Volterra kernel-based PNLSS model P4 on a Continuous Stirred Tank Reactor (CSTR), a benchmark system frequently used in nonlinear system identification and control. The process involves an exothermic chemical reaction, where the concentration of the reactant is controlled by manipulating the coolant flow rate. The outputs of interest are the concentration of the reactant in mol/l and the temperature of the mixture in Kelvin.

The dataset was contributed by Jairo Espinosa from KU Leuven (Morningred et al., 1992; Espinosa et al., 2004) and has been widely used in studies of nonlinear predictive control and fuzzy/neural model-based approaches (Naikwad and Dudul, 2009). The data consists of 7,500 samples collected at a sampling interval of 0.1 minutes. For identification purposes, we consider the system as a SISO model for each output, using coolant flow as the input and analysing concentration and temperature independently.

A total of 7,500 samples are available. The first 5,500 are used for training, while 1,000 samples each are allocated for validation and testing. The data reflect the nonlinear dynamics of the reactor, including effects such as energy accumulation, reaction kinetics, and disturbances induced by the coolant.

The RMSEs between the true values and those predicted by various models are reported in Table 6.6. For the concentration output, the best performance is achieved by the PNLSS model B2 initialised using BLA, with an RMSE of 1.2 mmol/l (RRMSE = 7.53%). For the temperature output, the BLA-initialised PNLSS B2 again achieves the lowest RMSE of 0.14 K (RRMSE = 3.82%).

The predictive performance is further visualised in Figure 6.25, which displays the

Table 6.6: Performance on Process Industry System III - Wiener Model

A. Tank Reactor - Concentration (Training sample: $N = 5{,}500$, Test sample: $1{,}000$, $\mathrm{RMS}(y_{\mathrm{test}}) = 16$ mmol/l)

Methodology Details	M	D	No. of Parameters	RMSE (mmol/l) ($\mathrm{RRMSE}_1\%$)	$\bar{\mu}$ (mmol/l)	$\bar{\sigma}$ (mmol/l)	Training time (s)
Volterra	10	3	363	2.0 (12.65)	-0.7	1.87	3
BS_Volterra	10	3	14	2.8 (17.71)	-1.8	2.14	18
BS_Volterra_LM	21	3	25	1.8 (11.49)	-0.9	1.59	60
PNLSS_BS_Volterra_LM	3	3	46	2.8 (17.68)	0.4	2.77	60
BLA	3	–	16	4.0 (25.07)	0.5	3.93	210
PNLSS_BLA_LM	3	3	46	**1.2 (7.53)**	-0.6	1.04	236
N4SID	3	–	16	2.6 (16.65)	0.5	2.59	2
PNLSS_N4SID_LM	3	3	46	2.6 (16.23)	0.4	2.53	22

B. Stirred Tank Reactor - Temperature (Training sample: $N = 5{,}500$, Test sample: $1{,}000$, $\mathrm{RMS}(y_{\mathrm{test}}) = 3.62$ K)

Methodology Details	M	D	No. of Parameters	RMSE (K) ($\mathrm{RRMSE}_1\%$)	$\bar{\mu}$ (K)	$\bar{\sigma}$ (K)	Training time (s)
Volterra	10	3	363	0.34 (9.25)	0.05	0.33	2
BS_Volterra	10	3	14	0.46 (12.69)	0.16	0.43	4
BS_Volterra_LM	21	3	25	0.30 (8.25)	0.07	0.29	16
PNLSS_BS_Volterra_LM	3	3	46	0.42 (11.63)	0.17	0.39	16
BLA	3	–	16	0.61 (16.93)	-0.03	0.61	130
PNLSS_BLA_LM	3	3	46	**0.14 (3.82)**	0.02	0.14	151
N4SID	3	–	16	0.33 (9.18)	-0.02	0.33	0.1
PNLSS_N4SID_LM	3	3	46	**0.19 (5.25)**	0.02	0.19	13

true and predicted outputs over the test set. The left panel shows the concentration trajectories, and the right panel depicts the temperature evolution. All three PNLSS-based models exhibit closely overlapping predictions, consistent with the quantitative results in the table.

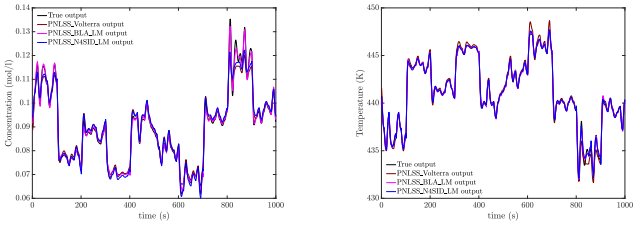


Figure 6.25: Predicted vs. true concentration and temperature of the mixture in the reactor for the test set. Left: Concentration; right: Temperature. The three PNLSS-based models exhibit nearly identical tracking accuracy across all regions.

Although the concentration and temperature outputs exhibit an inverse shape relationship (see Figure 6.25), a typical feature of exothermic reactions, the identification results differ significantly when each output is modelled independently using a SISO framework. This suggests that while the outputs are physically coupled, modelling them as separate SISO systems prevents the models from exploiting shared structural dynamics. As a

result, the prediction error does not mirror between the two outputs as might be expected. This observation highlights a limitation of the current SISO approach and suggests that a MIMO modelling framework could better exploit inter-output dependencies in future work.

6.9 Conclusion

This chapter presented a structured framework for Wiener system identification using Volterra kernel estimation and tensor-based modelling. Unlike the Hammerstein case, the non-diagonal structure of the Wiener kernel requires more involved construction and computational effort. To address this, we leveraged tensor network representations, specifically, tensor train matrices, to efficiently represent and compute the Wiener kernel, even in high-dimensional settings.

The kernel structure was then used to extract an interpretable block-oriented Wiener model, which was further optimised using nonlinear least-squares techniques. From the refined block-oriented model, a compact PNLSS model was realised using subspace methods. While the predictive accuracy of the final PNLSS models was found to be comparable to that of classical approaches, the proposed modelling framework offered a key advantage: structural interpretability via explicit identification of the linear and nonlinear subsystems.

Simulation results confirmed that although the method involves a higher computational cost due to the complexity of the Wiener kernel, it provides consistent parameter recovery and interpretability not present in conventional PNLSS models. Furthermore, by incorporating structure-aware regularisation and low-rank tensor formats, the approach lays the groundwork for generalising to more complex nonlinear systems, including those beyond strict block-oriented architectures.

Chapter 7

Prior-Informed Volterra Kernel Estimation for WH System

Abstract: This chapter presents the proposed Volterra kernel-based framework for identifying WH systems by exploiting prior knowledge about the model structure. A structured Volterra kernel is constructed that embeds the known WH topology, enabling the separation of the two LTI blocks and the static nonlinearity. To ensure robustness and interpretability, a two-stage shape-enforced optimisation is introduced, promoting smoothness and decay in the kernel while preserving the block-oriented structure. The optimised model is then transformed into a reduced-order PNLSS model that respects the WH configuration. The proposed methodology is evaluated through numerical simulations, experimental measurements, and a standard nonlinear system identification benchmark, demonstrating its accuracy, efficiency, and practical relevance.

Note on Chapter Content: This chapter comprises both previously published and unpublished work by the PhD candidate, supplemented with selected material from existing literature to provide context and facilitate comparison. Specifically:

- Sections 7.1, 7.2, and parts of Section 7.5 are based on P. Vijayan, P. Dreesen, J. Lataire and M. Ishteva, "Enhancing Block-Oriented System Identification Using *Prior-Informed* Volterra Kernels," in *IEEE Transactions on Instrumentation*

and Measurement, vol. 74, pp. 1–11, 2025, Article No. 6507311. DOI: 10.1109/TIM.2025.3575968.

- Sections 7.3 and 7.4 present original, unpublished work by the PhD candidate.

This research was conducted by the PhD candidate under the supervision of the co-authors, who served as (co-)promoters.

7.1 Construction of Wiener-Hammerstein Kernel

The WH model extends the flexibility of block-oriented structures by combining two LTI blocks with a static nonlinearity, allowing it to capture more complex system dynamics than pure Wiener or Hammerstein models (see Figure 1.5). As before, we assume a polynomial static nonlinearity of degree D, with impulse responses $\boldsymbol{s}$ and $\boldsymbol{t}$ of orders M_{w} and M_{h}, respectively, for the input and output LTI blocks. The resulting model structure is given by:

$$y_k = \sum_{l=0}^{M_{\mathrm{h}}} t_l \left\{ \sum_{j=1}^{D} p_j \left(\sum_{i=0}^{M_{\mathrm{w}}} s_i u_{k-i-l} \right)^j \right\}, \tag{7.1}$$

To illustrate the equivalence between this WH model and the Dth-order finite memory Volterra model, consider the minimal working example:

Example 20. For degree $D = 2$ and memory lengths of two LTI blocks $M_{\mathrm{w}} = M_{\mathrm{h}} = 2$, the model (7.1) expands to:

$$\begin{aligned} y_k &= t_0 \Big[p_1 (s_0 u_k + s_1 u_{k-1} + s_2 u_{k-2}) + p_2 (s_0 u_k + s_1 u_{k-1} + s_2 u_{k-2})^2 \Big] \\ &\quad + t_1 \Big[p_1 (s_0 u_{k-1} + s_1 u_{k-2} + s_2 u_{k-3}) + p_2 (s_0 u_{k-1} + s_1 u_{k-2} + s_2 u_{k-3})^2 \Big] \\ &\quad + t_2 \Big[p_1 (s_0 u_{k-2} + s_1 u_{k-3} + s_2 u_{k-4}) + p_2 (s_0 u_{k-2} + s_1 u_{k-3} + s_2 u_{k-4})^2 \Big] \end{aligned} \tag{7.2}$$

$$y_k = \sum_{i=0}^{2} t_i \cdot \begin{bmatrix} p_1 z_i \\ p_2 z_i^2 \end{bmatrix}, \qquad z_i = \boldsymbol{s}^\top \begin{bmatrix} u_{k-i} \\ u_{k-i-1} \\ u_{k-i-2} \end{bmatrix}.$$

This formulation reveals that the WH model imposes a specific structure on the parameters. Since the nonlinear transformation is sandwiched between two linear blocks, there are some cross-terms absent. In the above Example, there are no cross-terms between u_k and $\{u_{k-3}, u_{k-4}\}$ and between u_{k-1} and u_{k-4}. This forms a special block diagonal structure in the nonlinear kernel when expanding the Volterra model (1.16) for the same hyperparameters $D = 2$ and $M = M_{\mathrm{w}} + M_{\mathrm{h}}$, gives:

$$y_k = \left(\begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ \vdots \\ u_{k-4} \end{bmatrix} \otimes \begin{bmatrix} 1 \\ u_k \\ u_{k-1} \\ \vdots \\ u_{k-4} \end{bmatrix} \right) \boldsymbol{h}. \tag{7.3}$$

It can be seen that the general Volterra model includes all the monomials in the input. Therefore, the parameters corresponding to the cross-terms that are not included in the WH model (7.2) are zero.

To derive the block-oriented model parameters from the Volterra kernel, we expand and rewrite (7.1) for order $D = 2$ in the form of the Volterra model $\boldsymbol{y} = \boldsymbol{U}\boldsymbol{h}$ (5.4). Thus, we get the reshaped symmetric kernel tensor in Figure 7.1, which exhibits a banded structure in the second-order kernel.

The linear kernel is a convolution of the two impulse responses $\boldsymbol{s}$ and $\boldsymbol{t}$, scaled by the coefficient p_1. The higher-order WH kernels exhibit a structured and interpretable form, which, in special cases, reduce to well-known kernel structures: for example, when $t_0 = 1$ and $t_1 = \cdots = t_{M_{\mathrm{h}}} = 0$, the structure simplifies to a Wiener system kernel (see (6.4)); and when $s_0 = 1$ and $s_1 = \cdots = s_{M_{\mathrm{w}}} = 0$, it reduces to a Hammerstein system kernel (see (5.11)).

The second-order kernel matrix $\boldsymbol{H}^{(2)}$ is constructed by sequentially placing a series of rank-one blocks of the form $p_2 t_{k-1} \boldsymbol{s}\boldsymbol{s}^\top$, each of size $(M_{\mathrm{w}} + 1) \times (M_{\mathrm{w}} + 1)$, where $\boldsymbol{s} \in \mathbb{R}^{M_{\mathrm{w}}+1}$ is fixed, and t_{k-1} is a scalar weight for the kth block.

Each such block is inserted diagonally starting at position (k, k), for $k = 1, 2, ..., M_{\mathrm{h}}+1$, effectively forming a banded pattern along the main diagonal. Specifically,

- the first block spans the submatrix $\boldsymbol{H}^{(2)}_{1:M_{\mathrm{w}}+1,1:M_{\mathrm{w}}+1}$,
- the second block spans $\boldsymbol{H}^{(2)}_{2:M_{\mathrm{w}}+2,2:M_{\mathrm{w}}+2}$,
- and so on, until the last block fills $\boldsymbol{H}^{(2)}_{M_{\mathrm{h}}+1:M+1,M_{\mathrm{h}}+1:M+1}$.

All entries outside this diagonal band are zero.

$$
\boldsymbol{H} = \left[\begin{array}{c|cccccccc}
0 & \frac{p_1 s_0 t_0}{2} & \frac{p_1(s_0 t_1 + s_1 t_0)}{2} & \cdots & \cdots & \cdots & \cdots & \frac{p_1 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} \\
\hline
\frac{p_1 s_0 t_0}{2} & p_2 s_0^2 t_0 & \frac{2p_2 s_0 s_1 t_0}{2} & \cdots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_0}{2} & 0 & \cdots & 0 \\
\frac{p_1(s_0 t_1 + s_1 t_0)}{2} & \frac{2p_2 s_0 s_1 t_0}{2} & p_2(s_0^2 t_1 + s_1^2 t_0) & & \vdots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_1}{2} & \ddots & \vdots \\
\vdots & \vdots & & \ddots & \vdots & \ddots & \ddots & 0 \\
\vdots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_0}{2} & \cdots & \cdots & p_2(s_0^2 t_{M_\mathrm{h}} + \cdots + s_{M_\mathrm{w}}^2 t_0) & & \ddots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} \\
\vdots & 0 & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_1}{2} & \ddots & & \ddots & & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & & \ddots & \vdots \\
\frac{p_1 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} & 0 & \cdots & 0 & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} & \cdots & \cdots & p_2 s_{M_\mathrm{w}}^2 t_{M_\mathrm{h}}
\end{array}\right]
$$

Figure 7.1: Reshaped symmetric second-order Wiener-Hammerstein kernel $\boldsymbol{H}$, showing banded structure.

This construction results in a structured second-order kernel matrix composed of diagonally shifted, overlapping rank-one blocks. Each block captures both temporal structure (through $\boldsymbol{s}$) and the dynamic weighting (through $\boldsymbol{t}_k$), while the global scaling p_2 encodes the contribution of the second-order nonlinearity.

Similarly, the third-order kernel $\mathcal{H}^{(3)}$ can be visualised as a banded tensor, where each block is a rank-one third-order tensor of the form $p_3 t_{k-1}(\boldsymbol{s} \circ \boldsymbol{s} \circ \boldsymbol{s})$, diagonally shifted and repeated for $k = 1, 2, ..., M_\mathrm{h} + 1$. Figure 7.2 illustrates individual components of a third-order WH tensor, highlighting the banded structure in the kernel orders 2 and 3.

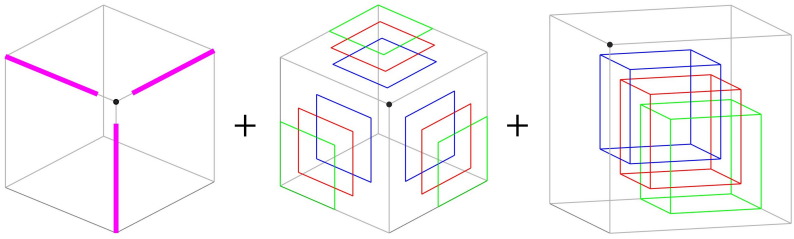


Figure 7.2: Component-wise visualisation of the Wiener-Hammerstein kernel tensor, illustrating the first-, second-, and third-order structures across different orders. The element at position $(1, 1, 1)$ corresponds to the constant h_0 and is highlighted with a black marker. The vectors $\boldsymbol{h}^{(1)}$ in all directions represent the coefficients of linear terms. The banded submatrix $\boldsymbol{H}^{(2)}$ in all directions represents the coefficients of second-order monomials. The banded sub-tensor $\mathcal{H}^{(3)}$ represents the coefficients of third-order monomials.

This structure generalises naturally to any higher-order kernel, where the dth-order kernel is composed of diagonally overlapping rank-one tensors of the form $p_d t_{k-1} \boldsymbol{s}^{\circ d}$, each encoding a homogeneous polynomial nonlinearity of order d shaped by the system's memory. The representation

$$(\boldsymbol{s}^{\circ d})_{i_1 . i_2, \ldots, i_d} = \boldsymbol{s}_{i_1} \boldsymbol{s}_{i_2} \ldots \boldsymbol{s}_{i_d}$$

denotes the symmetric d-way outer product of the vector $\boldsymbol{s}$, resulting in a rank-one tensor of order d.

Overall, this formulation provides a compact, structured, and interpretable representation of the WH kernels, where $\boldsymbol{s}$ captures the input dynamics, $\boldsymbol{t}$ controls the shifting and weighting of nonlinear terms, and the polynomial coefficients p_d govern the strength of each nonlinear order.

The zero entries outside the diagonal band in the Volterra kernel tensor represent structural prior knowledge of the WH model, analogous to the Hammerstein case. This

structured sparsity significantly reduces the number of parameters to be estimated. In concrete terms, not all nonlinear cross-terms appear in the WH model due to its sandwiched architecture, and this can be formalised as follows:

- Owing to the cascade of two linear dynamic blocks surrounding the static nonlinearity, the output at time step k depends nonlinearly only on past inputs that have passed through the first dynamic block.
- This results in a Volterra representation at time k where only monomials involving time-shifted filtered inputs $u_{k-l}, u_{k-1-l}, \ldots, u_{k-M_\mathrm{w}-l}$ for $l = 0, \ldots, M_\mathrm{h}$ contribute to the output..
- Consequently, cross-terms involving inputs from incompatible time shifts, such as $u_i u_j$ with $i \in \{k, \ldots, (k-M_\mathrm{w})\}$ and $j \in \{(k-M_\mathrm{w}-1), \ldots, (k-M_\mathrm{w}-M_\mathrm{h})\}$, do not appear in the Volterra series expansion. Their corresponding coefficients in the kernel are structurally zero.

This prior knowledge enables elimination of the corresponding columns in the Volterra regression matrix $\boldsymbol{U}$, allowing estimation only over the non-zero entries of the kernel. The reduced problem formulation in (5.14) thus becomes more tractable, though the number of unknowns remains greater than in the Hammerstein case due to the added complexity of the WH structure.

7.2 Reconstructing WH Structure using WH Kernel

This section, similar to Section 6.5, discusses the identification of the *prior-informed* WH kernel from the reduced problem (5.14), and the subsequent steps involved in extracting the WH block-oriented model parameters from the identified *prior-informed* kernel.

The estimation of the WH kernel follows the same principle as for the Hammerstein and Wiener models, by solving a reduced linear system of the form:

$$\boldsymbol{y}^{\mathrm{meas}} = \tilde{\boldsymbol{U}}\tilde{\boldsymbol{h}} + \boldsymbol{b}, \tag{7.4}$$

where $\tilde{\boldsymbol{U}}$ is the structured input matrix incorporating the known structure of the WH kernel, and $\tilde{\boldsymbol{h}}$ is the vector of unknown kernel coefficients. Recall that the solution is obtained using the pseudoinverse:

$$\tilde{\boldsymbol{h}} = \tilde{\boldsymbol{U}}^{\dagger}\boldsymbol{y}^{\mathrm{meas}}, \tag{7.5}$$

which can be computed via truncated SVD to mitigate noise sensitivity or ill-conditioning. This approach leverages the prior knowledge embedded in the WH kernel structure to

reduce the number of estimated parameters and improve numerical robustness. The result is a non-iterative, efficient estimation of the reduced kernel $\tilde{\boldsymbol{h}}$, which forms the basis for subsequent block-oriented model identification.

Since all Volterra kernels of the WH model can be expressed as convolutions of the two linear impulse responses $\boldsymbol{s}$ and $\boldsymbol{t}$, scaled by the corresponding polynomial coefficients $p_1, p_2, \ldots, p_D$, the identification of the individual block parameters becomes inherently challenging. These convolutions span increasing dimensions with higher kernel orders, resulting in overlapping, structured, and scale-ambiguous representations that complicate the direct recovery of the underlying system components.

$$\boldsymbol{H} = \left[\begin{array}{c|cccccccc}
0 & \frac{p_1 s_0 t_0}{2} & \frac{p_1(s_0 t_1 + s_1 t_0)}{2} & \cdots & \cdots & \cdots & \cdots & \frac{p_1 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} \\
\hline
\frac{p_1 s_0 t_0}{2} & p_2 s_0^2 t_0 & \frac{2p_2 s_0 s_1 t_0}{2} & \cdots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_0}{2} & 0 & \cdots & 0 \\
\frac{p_1(s_0 t_1 + s_1 t_0)}{2} & \frac{2p_2 s_0 s_1 t_0}{2} & p_2(s_0^2 t_1 + s_1^2 t_0) & & \vdots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_1}{2} & \ddots & \vdots \\
\vdots & \vdots & & \ddots & \vdots & \ddots & \ddots & 0 \\
\vdots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_0}{2} & \cdots & \cdots & p_2(s_0^2 t_{M_\mathrm{h}} + \cdots + s_{M_\mathrm{w}}^2 t_0) & & \ddots & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} \\
\vdots & 0 & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_1}{2} & \ddots & & \ddots & & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & & \ddots & \vdots \\
\frac{p_1 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} & 0 & \cdots & 0 & \frac{2p_2 s_0 s_{M_\mathrm{w}} t_{M_\mathrm{h}}}{2} & \cdots & \cdots & p_2 s_{M_\mathrm{w}}^2 t_{M_\mathrm{h}}
\end{array}\right] \tag{7.6}$$

To address this challenge, the following procedure is proposed:

1. Fix the scale ambiguity: Set $p_1 = 1$ and fix $s_0 = \hat{s}_0 = 1$ to eliminate scale redundancy.

2. Identify $\hat{t}_0$ using the red-highlighted element in the first-order kernel in (7.6) and (5.7):
$$\hat{t}_0 = h_1(0).$$

3. Estimate $\hat{p}_2$ from the second-order kernel element indicated in black rectangle in (7.6):
$$\hat{p}_2 = \frac{h_2(0,0)}{\hat{s}_0^2 \hat{t}_0}.$$

4. Estimate the remaining elements of $\hat{\boldsymbol{s}}$, i.e., $\hat{s}_1, \ldots, \hat{s}_{M_\mathrm{w}}$, using the first row of $\boldsymbol{H}^{(2)}$ (blue box in (7.6)), from the kernel elements $h_2(0,1), \cdots, h_2(0, M_\mathrm{w})$.

5. Estimate $\hat{\boldsymbol{t}}$ by solving the structured system formed by the diagonal entries of higher-order kernels (highlighted in yellow in (7.6)). For $j = 2$, solve:

$$\hat{p}_j \begin{bmatrix} \hat{s}_0^j & & \\ \vdots & \ddots & \\ \hat{s}_{M_\mathrm{w}}^j & \ddots & \hat{s}_0^j \\ & \ddots & \vdots \\ & & \hat{s}_{M_\mathrm{w}}^j \end{bmatrix} \begin{bmatrix} \hat{t}_0 \\ \vdots \\ \hat{t}_{M_\mathrm{h}} \end{bmatrix} = \begin{bmatrix} h_j(0, \cdots, 0) \\ h_j(1, \cdots, 1) \\ \vdots \\ h_j(M, \cdots, M) \end{bmatrix}. \tag{7.7}$$

6. Estimate remaining $\hat{p}_j$ coefficients using least-squares:

$$\hat{p}_j = \frac{(\boldsymbol{S}.\hat{\boldsymbol{t}})^\top \boldsymbol{h}_j}{(\boldsymbol{S}.\hat{\boldsymbol{t}})^\top (\boldsymbol{S}.\hat{\boldsymbol{t}})}, \tag{7.8}$$

where $\boldsymbol{h}_j$ collects the diagonal elements of the jth kernel and $\boldsymbol{S}$ is the Toeplitz matrix constructed using powers of $\hat{\boldsymbol{s}}$ as in (7.7) in step 5.

This approach exploits the inherent structure of the symmetric Volterra kernel induced by the WH model and yields closed-form, interpretable parameter estimates. These estimates serve as a strong initialisation for further refinement. However, due to measurement noise or model imperfections, the estimated kernels may no longer strictly follow the convolutional form, leading to deviations in parameter recovery. To address this, the next section formulates a nonlinear optimisation procedure that refines $\hat{\boldsymbol{s}}, \hat{\boldsymbol{t}}, \hat{p}_1, \ldots, \hat{p}_D$ by minimising the output prediction error, while enforcing desirable structural properties such as stability, decay and smoothness.

7.3 Shape-enforced Optimisation of WH Kernel

To enhance the robustness and interpretability of the WH model in all intermediate steps, we perform regularised optimisation in two stages, similar to those in Chapters 5 and 6, which enforce prior structural knowledge and physical plausibility of the block-oriented model components. In both stages, the same optimisation procedure is applied, including regularisation terms that promote stability and shape conformity, to ensure that not only the final but also the intermediate models provide meaningful and faithful representations of the underlying system. Stage 1 corresponds to a more constrained setting, with lower model complexity defined by hyperparameters D, M_w and M_h, applied directly after extracting the block-oriented model estimates from the Volterra kernel of memory length $M = M_\mathrm{w} + M_\mathrm{h}$. Stage 2 repeats the same optimisation but with increased flexibility, using extended hyperparameters $D_+, M_{\mathrm{w}+}$ and $M_{\mathrm{h}+}$.

This section describes both stages of the procedure, which refine the initial parameter estimates extracted from the Volterra kernel while maintaining physically meaningful intermediate representations.

Stage 1: Structure-Constrained Refinement
In this step, the WH initial estimates $(\hat{\boldsymbol{s}}, \hat{\boldsymbol{t}}, \hat{\boldsymbol{p}})$, extracted analytically from the structured WH kernel (as discussed previously), are used to initialise a nonlinear optimisation that jointly refines all parameters by minimising the following regularised least-squares cost:

$$\begin{aligned}\min_{\boldsymbol{s},\boldsymbol{p},\boldsymbol{t}} J(\boldsymbol{s},\boldsymbol{p},\boldsymbol{t}) = & \underbrace{\|\hat{\boldsymbol{y}}(\boldsymbol{u};\boldsymbol{s},\boldsymbol{p},\boldsymbol{t}) - \boldsymbol{y}\|_2^2}_{\text{data fit}} \\ & + \underbrace{\mu_s \|\boldsymbol{w}_s \odot \boldsymbol{s}\|_2^2 + \lambda_s \left\|\boldsymbol{\Delta}^2 \boldsymbol{s}\right\|_2^2}_{\text{shape constraints on } \boldsymbol{s}} \\ & + \underbrace{\mu_t \|\boldsymbol{w}_t \odot \boldsymbol{t}\|_2^2 + \lambda_t \left\|\boldsymbol{\Delta}^2 \boldsymbol{t}\right\|_2^2}_{\text{shape constraints on } \boldsymbol{t}},\end{aligned} \tag{7.9}$$

where

- $\hat{\boldsymbol{y}}$ is the model prediction reconstructed from the WH parameters $\boldsymbol{s}, \boldsymbol{p}, \boldsymbol{t}$,
- Δ^2 denotes the second-order finite difference operator to promote smoothness in $\boldsymbol{s}$ and $\boldsymbol{t}$,
- $\boldsymbol{w}_s = [0, 1, \ldots, M_{\mathrm{w}}]^\top$ and $\boldsymbol{w}_t = [0, 1, \ldots, M_{\mathrm{h}}]^\top$ are ramp vectors that enforce softer decay by penalising later taps (than aggressive decay by exponential weights). The linear weights ensure that the optimiser penalises later lags more strongly and still preserves flexibility in the tail to identify varying decay patterns across different system dynamics.
- λ_s, λ_t and μ_s, μ_t are regularisation weights balancing curvature and decay, respectively.

A key feature of this refinement is that it preserves the structural prior of the WH kernel while aligning the model output with noisy measurements. This results in improved generalisation and physical interpretability of the estimated parameters.

The refined parameters $(\hat{\boldsymbol{s}}, \hat{\boldsymbol{t}}, \hat{\boldsymbol{p}})$ then serve as a high-quality initialisation for the second stage of optimisation, which lifts the parameter constraints to allow further flexibility during output error minimisation in the full block-oriented model space.

Stage 2: Shape-enforced Optimisation
In the second step, the WH model is refined in the full block-oriented model parameter

space $\boldsymbol{\theta}$ by directly minimising the output prediction error. This formulation extends the memory lengths of both LTI blocks and the degree of the static nonlinearity, enabling more flexible and accurate modelling of the system dynamics. The use of longer memory lengths and higher-degree nonlinearity marks the difference between stage 1 and stage 2 of shape enforcement. To preserve interpretability and robustness, smoothness and decay priors are imposed on the impulse responses. The output is modelled as:

$$\begin{aligned}\Phi\left(\boldsymbol{u}_k, \begin{bmatrix}\boldsymbol{\theta}^{\mathrm{NL}} \\ \boldsymbol{\theta}^{\mathrm{L1}} \\ \boldsymbol{\theta}^{\mathrm{L2}}\end{bmatrix}\right) &= \sum_{l=0}^{M_{\mathrm{h}+}} \hat{t}_l \left\{ \sum_{j=1}^{D_+} \hat{p}_j \left(\sum_{i=0}^{M_{\mathrm{w}+}} \hat{s}_i u_{k-i-l} \right)^j \right\} \\ &= \sum_{l=0}^{M_{\mathrm{h}+}} \boldsymbol{\theta}^{\mathrm{L2}}_{D_++1+M_{\mathrm{w}+}+l} \left\{ \sum_{j=1}^{D_+} \theta^{\mathrm{NL}}_j \left(\sum_{i=0}^{M_{\mathrm{w}+}} \theta^{\mathrm{L1}}_{D_++1+i} u_{k-i-l} \right)^j \right\},\end{aligned} \tag{7.10}$$

where D_+, $M_{\mathrm{w}+}$ and $M_{\mathrm{h}+}$ denote the extended nonlinearity degree and memory lengths of the first and second LTI blocks.

The model parameters $\boldsymbol{\theta}$ are estimated by solving a regularised nonlinear least-squares problem that penalises high curvature and slow decay in both LTI impulse responses and shrinkage of polynomial terms. This ensures that the final WH model remains compact, stable, and well-aligned with the physical characteristics of the underlying system.

7.4 Realising PNLSS Model from the optimised WH kernel

The procedure for realising a PNLSS model from the optimised WH kernel closely follows the methodology applied in the Hammerstein and Wiener cases (see Sections 5.2.4 and 6.7). However, a key distinction in the WH case is that the two LTI blocks must be treated separately when constructing the LSS dynamics.

The optimised WH model provides the impulse responses $\boldsymbol{\theta}^{\mathrm{L1}}$ and $\boldsymbol{\theta}^{\mathrm{L2}}$ of the front and back LTI blocks, respectively. These impulse responses are used to extract two independent LSS realisations $(\{\boldsymbol{A}_1, \boldsymbol{b}_1, \boldsymbol{c}_1, d_1\}, \{\boldsymbol{A}_2, \boldsymbol{b}_2, \boldsymbol{c}_2, d_2\})$ using the ERA. Each of these LTI blocks is realised with model orders $M_{\mathrm{w}-} \ll M_{\mathrm{w}+}$ and $M_{\mathrm{h}-} \ll M_{\mathrm{h}+}$, reflecting a parsimonious approximation of the truncated impulse response estimates obtained during block-oriented model optimisation.

Once both LTI blocks are realised, they are assembled into a composite state-space structure consistent with the WH topology. The total state vector consists of the

concatenation of the internal states from the two LTI blocks:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \end{bmatrix},$$

where $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ correspond to the states of the first and second LTI systems, respectively. The nonlinear mapping in the WH structure is realised as a polynomial nonlinearity based on the optimised coefficients $\boldsymbol{\theta}^{\mathrm{NL}} = [\hat{p}_1, \ldots, \hat{p}_D]$, using the structures in (4.50) and (4.52).

In the ideal, noiseless conditions, where the LTI blocks are FIR models of orders $M_{\mathrm{w}+}$ and $M_{\mathrm{h}+}$, and the SNL has degree D_+, the PNLSS state dimension is taken as $M_{\mathrm{w}-} + M_{\mathrm{h}-} = M_{\mathrm{w}+} + M_{\mathrm{h}+}$. For the WH case, the theoretical structure of the PNLSS parameters (4.50) and (4.52) is obtained, where the individual parameters are:

$$\boldsymbol{A}_{1,2} = \begin{bmatrix} 0 & \dots & \dots & \dots & 0 \\ 1 & \ddots & & & \vdots \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix}, \quad \boldsymbol{b}_{1,2} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \boldsymbol{c}_1 = \begin{bmatrix} \theta_2^{\mathrm{L1}} \\ \vdots \\ \theta_{M_{\mathrm{w}+}}^{\mathrm{L1}} \end{bmatrix}^\top, \quad \boldsymbol{c}_2 = \begin{bmatrix} \theta_2^{\mathrm{L2}} \\ \vdots \\ \theta_{M_{\mathrm{h}+}}^{\mathrm{L2}} \end{bmatrix}^\top, \quad d = \theta_1^{\mathrm{NL}} \theta_1^{\mathrm{L1}} \theta_1^{\mathrm{L2}}. \tag{7.11}$$

While in theory, the WH block structure can be exactly represented as the above PNLSS model with structured polynomial nonlinearities and FIR dynamics, in practice, the reduced-order LSS realisation via ERA provides better noise robustness and model compactness. The nonlinear mapping, derived directly from the optimised WH kernel, preserves the structure and interpretability of the original block-oriented model.

This transformation from block-oriented model to PNLSS thus enables leveraging standard tools for system simulation, control, and analysis, while retaining the insight and regularisation advantages from the WH kernel identification stage.

7.5 Results and Evaluation - WH System

This section reports the performance of the proposed identification methodology for WH systems through both numerical simulations and experimental validation. The evaluation follows the structure used for the Hammerstein and Wiener cases, enabling a consistent comparison across model types. In addition, the approach is benchmarked on a widely studied WH system from the system identification literature to demonstrate its practical effectiveness and generalisability.

Algorithm 7 From Volterra kernel to state-space model: a *prior-informed*, structured identification of a WH system

Input: Input-output data $\{u_k, y_k\}_{k=1}^{N}$
Output: PNLSS Parameters $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{E}, \boldsymbol{f}\}$

1: Choose Volterra memory $M = M_\mathrm{h} + M_\mathrm{w}$ and order D (ensure $N \geq I^D$, $I = M + 2$)
2: Build full design matrix $\boldsymbol{U} \in \mathbb{R}^{N \times I^D}$ and apply prior to obtain reduced $\tilde{\boldsymbol{U}}$
3: Estimate reduced Volterra kernel $\tilde{\boldsymbol{h}}$
4: Reshape the kernel $\tilde{\boldsymbol{h}}$ into a D-way tensor and extract the block-oriented model parameters estimates $\{\hat{\boldsymbol{s}}, \hat{\boldsymbol{t}}, p_1, \ldots, p_D\}$, as explained in Section 7.2
5: Optionally regularise/extend ($M_{\mathrm{h}+} > M_\mathrm{h}, M_{\mathrm{w}+} > M_\mathrm{w}, D_+ > D$) using (7.9)–(7.10) to get extended impulse responses
6: Realise the LSS model parameters from the regularised kernels $\hat{\boldsymbol{s}}, \hat{\boldsymbol{t}}$ individually, as explained in Section 7.4, by selecting model orders $M_{\mathrm{w}-}, M_{\mathrm{h}-}$ from singular-energy criterion (90–95%)
7: Assemble the full PNLSS model along with the nonlinear mappings
8: Return $\{\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, d, \boldsymbol{E}, \boldsymbol{f}\}$

7.5.1 Numerical Simulations

The simulation setup for the WH system closely follows that of the Hammerstein and Wiener case, with key differences highlighted here. The system includes two third-order low-pass LTI blocks: the first with a passband frequency of 0.5 Hz, and the second with 0.4 Hz. All other settings, including the choice of static nonlinearity (polynomial, dead zone, or sigmoid) and noise characteristics, remain unchanged. The input–output data consists of 10,000 training samples, with 2,500 samples each for validation and testing, as illustrated in Figure 7.3.

In the WH structure, the input passes through a linear low-pass filter, followed by a static nonlinearity and another low-pass filter. While the first filter and nonlinearity introduce frequency distortion and mixing, the second filter strongly attenuates high-frequency components, resulting in an output that predominantly exhibits low-pass characteristics with reduced nonlinear spectral content compared to Wiener systems. This dual filtering masks nonlinear effects in the output spectrum, making it harder to distinguish and identify the individual dynamics of each block.

Table 7.1 summarises the identification performance for WH systems with three different nonlinearities, evaluated under varying SNR conditions. It reports the output prediction error (RRMSE_1), and the relative parameter errors for the two linear blocks (RRMSE_2) and the nonlinear block (RRMSE_3), averaged over 20 Monte Carlo runs. Compared to the Hammerstein and Wiener system simulations in earlier chapters, where 40 Monte Carlo runs were used, we limit the WH system experiments to 20 runs. This

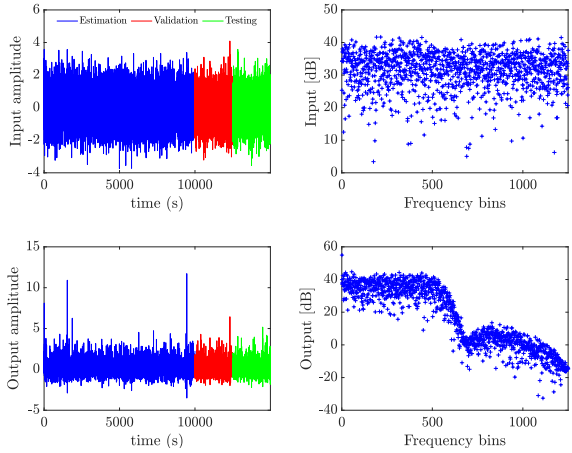


Figure 7.3: Simulation dataset showing the training, validation, and test splits. The Gaussian input excites a broad frequency range, while the output exhibits strong low-pass characteristics due to the cascaded linear filtering around the static nonlinearity.

choice is motivated by the significantly larger problem size: accurate Volterra-based representations of WH systems require longer memory lengths, resulting in larger model dimensions and higher computational demands. Additionally, the baseline identification approaches exhibit considerably longer execution times under these settings. Reducing the number of Monte Carlo runs to 20 thus provides a practical trade-off between statistical reliability and computational feasibility.

The model IDs used in this chapter follow the same convention as those defined earlier in Table 5.1 (Section 5.3.1.2). In Table 7.1.A, for the polynomial nonlinearity, the block-structured model P3 and its corresponding PNLSS realisation P4 deliver the highest output prediction accuracy, achieving RRMSE_1 values as low as 3.57% even at 5 dB SNR. The model P3 also exhibits strong parameter recovery performance, reflected in low RRMSE_2 and RRMSE_3 values for the linear and nonlinear blocks, respectively. The use of a *prior-informed* WH kernel leads to a compact representation, faster convergence, and efficient PNLSS realisation. In contrast, the baseline PNLSS models B2 and B4 show higher variability in output prediction accuracy. The asterisks (*) next to the reported RRMSE, mean, and standard deviation values indicate that, in some of the 20 Monte Carlo runs, unstable models (with NaN-valued RRMSE) were produced - this occurred only for model B2. As a result, the mean and standard deviation are computed only over the subset of stable realisations.

For dead zone ($[-0.3, 0.3]$) and sigmoid nonlinearities, Table 7.1.B and C, longer memory lengths were required for the two LTI blocks to capture the relevant dynamics in the Volterra model. This resulted in increased training time. Moreover, the inherent

Table 7.1: Performance on Simulated Test Dataset with 3 Types of Nonlinearity - Wiener-Hammerstein System

Methodology Details	M	D	No. of Parameters	RRMSE_1 (%)			RRMSE_2 in the LTI block 1 (%)			RRMSE_3 in the SNL block (%)			RRMSE_2 in the LTI block 2 (%)			$\bar{\mu} \times 10^{-3}$			$\bar{\sigma} \times 10^{-3}$			Training time (s)
				No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	No noise	SNR_1	SNR_2	
A. Polynomial nonlinear block - Training sample: $N = 10{,}000$, Test sample: $2{,}500$, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.88$, $N_{\mathrm{runs}} = 20$, SNR_1 = 20 dB, SNR_2 = 10 dB																						
Prior-informed Volterra	10+10	3	1143	2.89	4.70	11.96	–	–	–	–	–	–	–	–	–	-0.4	0.5	-0.1	26	41	106	3
BS_Volterra	10+10	3	25	6.02	9.25	23.48	0.36	0.54	1.35	**1.06**	**1.35**	4.67	0.42	0.51	1.02	5.5	5.9	14.9	53	81	207	3
BS_Volterra_LM	49+49	3	103	0.002	1.01	3.21	**.0001**	**0.11**	**0.40**	**0.93**	**1.42**	**3.94**	**.0001**	**0.13**	**0.42**	0.001	0.6	-.01	0.02	9	28	17
PNLSS_BS_Volterra_LM	6	3	160	**1.04**	**1.23**	**3.57**	–	–	–	–	–	–	–	–	–	0.1	0.9	3.3	9	11	31	17
BLA	6	–	49	38.31	40.64	39.64	–	–	–	–	–	–	–	–	–	210	214	211	265	287	279	110
PNLSS_BLA_LM	6	3	833	**14.31**	**19.02**	**16.96***	–	–	–	–	–	–	–	–	–	13.6	60.3	40.7*	126	152	137*	442
N4SID	6	–	49	36.51	36.52	36.54	–	–	–	–	–	–	–	–	–	209	209	209	245	246	246	1
PNLSS_N4SID_LM	6	3	833	**3.91**	**5.29**	**9.03**	–	–	–	–	–	–	–	–	–	2	3.1	3.3	34	47	79	413
B. Dead zone nonlinear block - Training sample, $N = 10{,}000$, Test sample: $2{,}500$, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.44$, $N_{\mathrm{runs}} = 20$, SNR_1 = 20 dB, SNR_2 = 10 dB																						
Prior-informed Volterra	20+20	3	7083	28.19	33.42	62.06	–	–	–	–	–	–	–	–	–	2.7	2.5	1.8	124	147	274	433
BS_Volterra	20+20	3	45	99.93	99.83	99.78	5.39	6.05	5.77	–	–	–	6.25	6.29	6.26	-3.5	-3.4	-4.0	441	440	440	434
BS_Volterra_LM	49+49	3	103	13.81	13.88	14.24	**0.66**	**2.31**	**1.77**	–	–	–	**0.50**	**3.85**	**6.13**	3.7	3.9	3.9	61	61	63	470
PNLSS_BS_Volterra_LM	6	3	160	**14.20**	**14.33**	**17.14**	–	–	–	–	–	–	–	–	–	2.6	2.7	2.3	62	63	75	470
BLA	6	–	49	20.94	21.38	22.05	–	–	–	–	–	–	–	–	–	0.05	-1.2	-1.2	92	94	97	104
PNLSS_BLA_LM	6	3	833	**14.39**	**15.25***	**17.86***	–	–	–	–	–	–	–	–	–	-0.7	-1*	-1*	63	67*	79*	489
N4SID	6	–	49	20.24	20.24	20.29	–	–	–	–	–	–	–	–	–	0.2	0.2	0.2	89	89	89	1
PNLSS_N4SID_LM	6	3	833	**10.70**	**12.05**	**14.13**	–	–	–	–	–	–	–	–	–	1.1	2.0	0.5	47	53	62	409
C. Sigmoid function nonlinear block - Training sample, $N = 10{,}000$, Test sample: $2{,}500$, $\mathrm{RMS}(y_{\mathrm{test}}) = 0.24$, $N_{\mathrm{runs}} = 20$, SNR_1 = 20 dB, SNR_2 = 10 dB																						
Prior-informed Volterra	20+20	3	7083	1.35	17.77	54.45	–	–	–	–	–	–	–	–	–	-0.1	-1.1	-0.4	3	43	131	429
BS_Volterra	20+20	3	45	90.77	99.73	99.62	4.39	6.57	5.92	–	–	–	2.98	6.65	6.19	-2.6	-2.4	-2.7	218	239	239	430
BS_Volterra_LM	49+49	3	103	0.60	1.37	3.62	**0.20**	**8.82**	**3.51**	–	–	–	**0.18**	**8.82**	**6.08**	-0.2	-0.3	0	1	3	9	490
PNLSS_BS_Volterra_LM	6	3	160	**1.44**	**3.25**	23.20	–	–	–	–	–	–	–	–	–	-0.2	-0.3	-0.2	3	8	56	490
BLA	6	–	49	4.12	4.53	8.32	–	–	–	–	–	–	–	–	–	0.05	0.02	0.01	10	11	20	136
PNLSS_BLA_LM	6	3	833	**1.08**	**2.75**	**7.22**	–	–	–	–	–	–	–	–	–	0.1	-0.1	0.3	3	7	17	569
N4SID	6	–	49	4.12	4.13	4.34	–	–	–	–	–	–	–	–	–	0.04	0.03	0.02	10	10	10	1
PNLSS_N4SID_LM	6	3	833	**1.04**	**2.1**	**3.64**	–	–	–	–	–	–	–	–	–	0.1	0	-0.2	3	5	9	374

model mismatch led to higher prediction errors in low SNR conditions compared to the baseline models B2 and B4.

Figure 7.4 compares RRMSE_1 across six modelling methods and three types of nonlinearities at 10 dB SNR. The top row displays the mean errors on a logarithmic scale, while the bottom row shows the distribution over 20 Monte Carlo runs using violin plots. Unlike in the Wiener case, the PNLSS model P4 initialised from the Volterra structure demonstrates consistent performance across all trials. In contrast, the model B2 exhibits occasional instability, leading to a few trials producing unstable models. Note that the violin plots include only those trials that resulted in stable models.

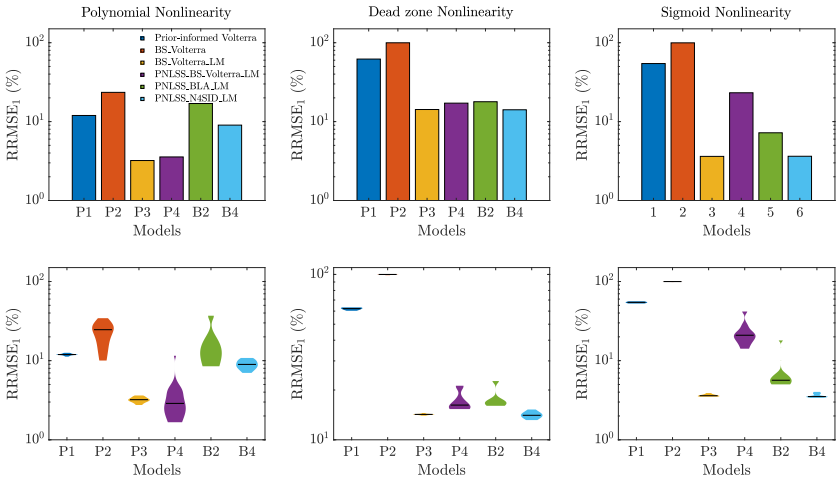


Figure 7.4: Comparison of output prediction errors (RRMSE_1 %) across six modelling approaches at SNR 10 dB, for three types of nonlinearities. The top row shows the mean errors on a log scale. The bottom row shows the error distribution over 20 Monte Carlo runs. PNLSS model P4, realised via the proposed Volterra-based approach, shows comparable performance across trials.

ANOVA - Report: Table 7.2 summarises the ANOVA results for the WH system under two SNR conditions (20 dB and 10 dB). At 20 dB SNR, the polynomial nonlinearity exhibits statistically significant differences across models ($p = 0.002$, $\eta^2 = 0.64$), indicating a strong effect. For the dead-zone and sigmoid cases, however, the differences are not statistically significant ($p > 0.05$), suggesting that at higher SNR levels the models perform comparably despite varying nonlinear structures.

At 10 dB SNR, significance becomes more pronounced for all nonlinearities ($p \leq 0.003$), with effect sizes remaining strong ($0.43 \leq \eta^2 \leq 0.72$). This demonstrates that the WH

Table 7.2: ANOVA Table - WH simulation dataset

Nonlinearity	p-value		Star rating		η^2		Effect size interpretation	
	SNR_1	SNR_2	SNR_1	SNR_2	SNR_1	SNR_2	SNR_1	SNR_2
Polynomial	0.002	$1e^{-9}$	**	***	0.64	0.50	Strong	Strong
Dead zone	0.19	0.003	-	**	0.26	0.43	Strong	Strong
Sigmoid	0.75	$1e^{-6}$	-	***	0.05	0.72	Weak	Strong

system's nonlinear dynamics are more clearly distinguishable under noisy conditions, as the reduced SNR accentuates the nonlinear distortions differently across models.

The corresponding box plots (Figure 7.5) and multiple-comparison plots (Figure 7.6) further illustrate these findings: model separation is more evident in the 10 dB case, particularly for the polynomial and sigmoid nonlinearities, where the confidence intervals between models show minimal overlap, confirming the statistical significance observed in the ANOVA test.

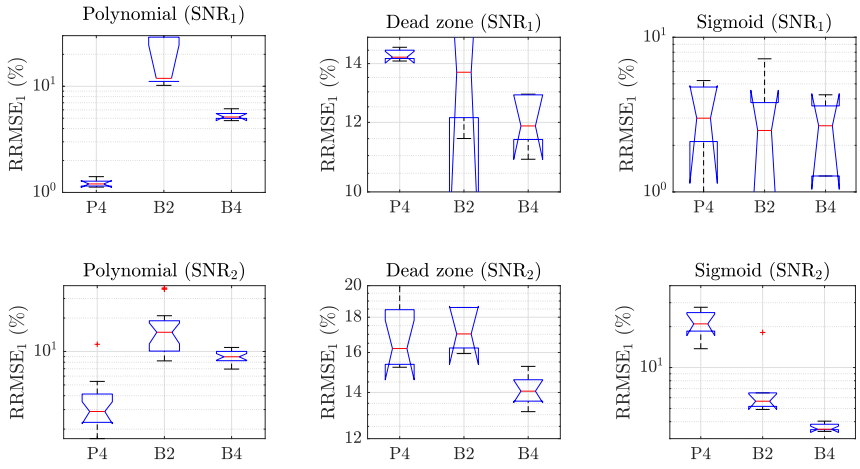


Figure 7.5: Box plots of RRMSE (%) for the three PNLSS models under different nonlinearities and SNR levels for the WH system. Each subplot corresponds to a specific nonlinearity–SNR combination. The boxes represent the distribution of model errors obtained from 40 Monte Carlo runs, illustrating both the median and spread of the RRMSE values. Lower median and narrower boxes indicate better and more consistent model performance across noise realisations.

Figure 7.7 illustrates the parameter accuracy after extracting from the WH kernels

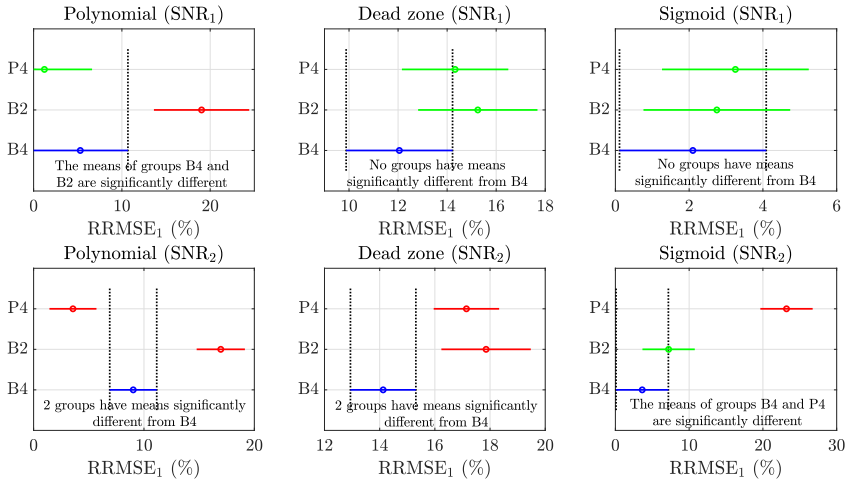


Figure 7.6: Multiple comparison results (Tukey–Kramer test) for pairwise mean differences in RRMSE among the three PNLSS models under different nonlinearities and SNR levels for the WH system. Each subplot corresponds to a specific nonlinearity–SNR combination. The blue marker represents the fixed reference model (B4), while green markers indicate models whose mean RRMSE lies within the 95% confidence interval of the reference model, implying statistically similar performance. Red markers denote models whose mean differs significantly from the reference, indicating statistically inferior or superior performance relative to B4.

for each nonlinearity. The top row presents the mean relative errors for the LTI and SNL blocks ($\mathrm{RRMSE}_{2,3}$) as bar plots, while the bottom row shows their distributions across 20 Monte Carlo runs using violin plots. For the polynomial case, the accuracy of parameter recovery is further illustrated in Figures 7.8 and 7.9, which compare the true and estimated pole-zero locations, frequency responses, and SNL coefficients.

Figure 7.10 compares model complexity and training time across methods. In the polynomial nonlinearity case, the proposed framework achieves fast convergence (≈ 17 s, refer Table 7.1.A) due to shorter memory lengths in the LTI blocks. For more complex nonlinearities, longer memory lengths increase the Volterra kernel size and training time (P1), despite the use of structural priors. However, the number of parameters is significantly reduced in the block-structured (P2, P3) and final PNLSS (P4) models. Notably, although all PNLSS variants use a 6th-order, 3rd-degree structure, the model P4 has far fewer parameters than the baselines B2 or B4, since the WH-specific structure is preserved during its realisation, unlike the unconstrained LM optimisation used in the others.

Figure 7.11 presents validation and test set prediction errors for the polynomial WH case.

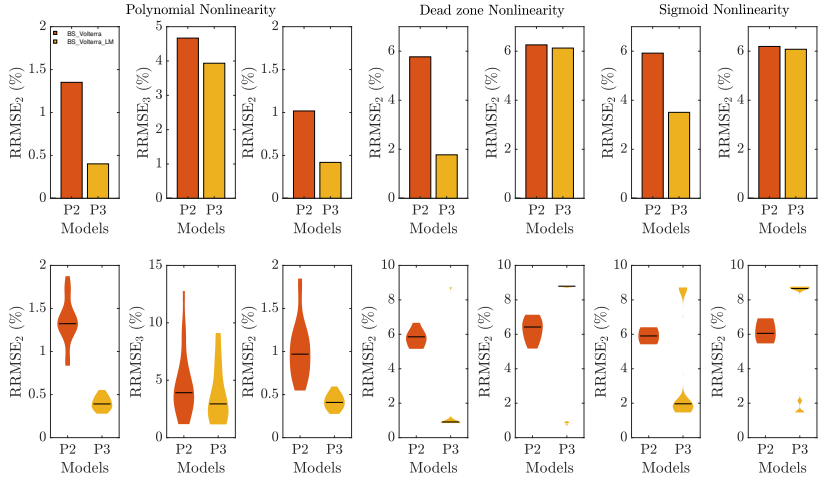


Figure 7.7: Relative parameter errors ($RRMSE_{2,3}$ %) of the LTI and SNL blocks for three types of nonlinearities at SNR 10 dB. Top row: Mean errors. Bottom row: Error distributions across 20 runs.

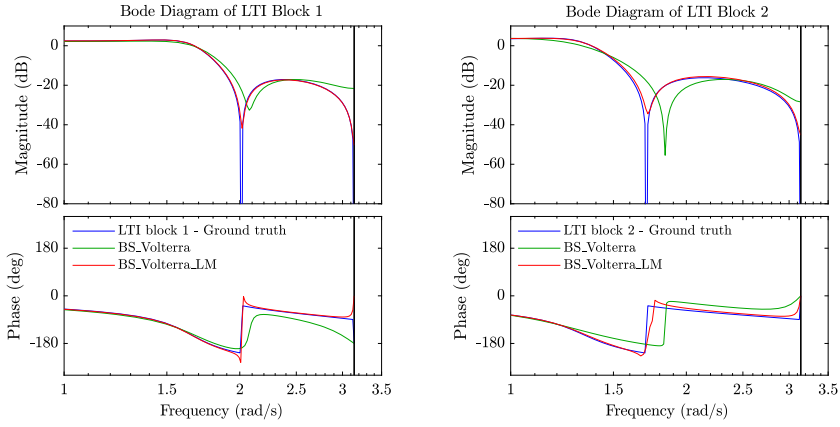


Figure 7.8: The Bode plots compare the true Linear filter characteristics with those of the averaged estimated filter characteristics by the block-structured models.

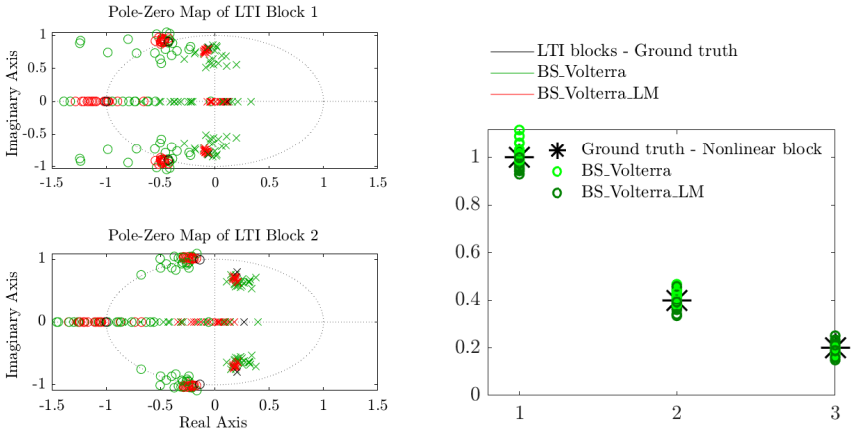


Figure 7.9: Pole-zero map (Legend: ×: poles, ∘: zeros) and SNL block coefficients for the WH system with polynomial nonlinearity (SNR 10 dB).

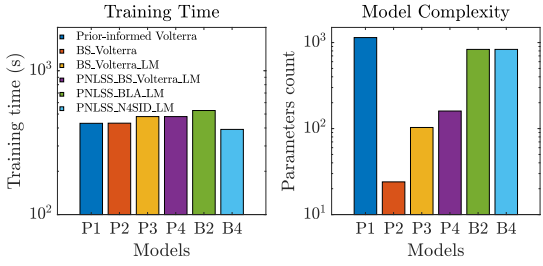


Figure 7.10: Training time and model complexity comparison for various nonlinear modelling strategies.

The left panel shows the validation error across LM iterations for all Monte Carlo trials, revealing that Volterra-based initialisations often begin closer to the optimum. The right panel displays test set prediction errors of the best models per method, confirming that the model P4 achieves better final accuracy than the baselines B2 and B4, consistent with Table 7.1.A.

7.5.2 Experimental Validation

This section evaluates the proposed identification framework using experimental data from a nonlinear electronic system configured in a WH structure. The measurement setup is identical to that described in Section 5.3.2, where the output is taken at the

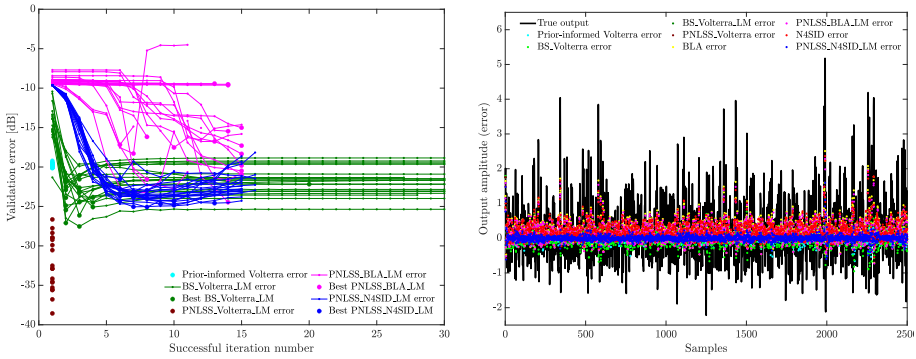


Figure 7.11: Left: Validation errors across iterations (polynomial nonlinearity, SNR 10 dB). Right: Test set prediction errors. PNLSS models P4 initialised from the Volterra model outperforms the baselines B2 and B4.

terminal stage of the full WH system (see Figure 7.12). The same setup forms the basis for the WH benchmark in the next section.

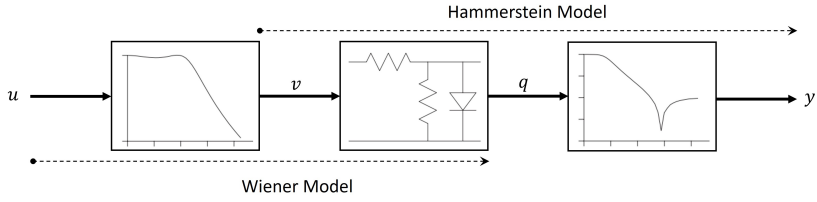


Figure 7.12: Measurement setup of Wiener-Hammerstein system. The full input–nonlinearity–output chain is measured.

The physical system again consists of a third-order Chebyshev Type I low-pass filter, a diode-based static nonlinearity, and a second low-pass filter, as introduced earlier. A multisine input with odd-bin excitation is used to drive the system, and measurements are sampled at 156 kHz. The resulting dataset is divided into a training set ($N = 31{,}256$) and validation and test sets (7,814 each).

Table 7.3 summarises the performance of different modelling approaches on the WH measurement dataset at 25 dB SNR. The proposed framework (P3 and P4) achieves low prediction errors with significantly fewer parameters than the full Volterra model, despite requiring longer training times. The `PNLSS_N4SID_LM` model also performs well in terms of accuracy, but at the cost of a much larger parameter count.

The initial *Prior-informed* Volterra model P1 achieves moderate prediction accuracy

Table 7.3: Performance on the Measurement Dataset

WH System (SNR 25 dB, Training sample: $N = 31{,}256$, Test sample: $7{,}814$, $\mathrm{RMS}(y_{\mathrm{test}}) = 52.4$ mV)

Methodology Details	M	D	No. of Parameters	RRMSE_1 (%)	$\bar{\mu}$ (mV)	$\bar{\sigma}$ (mV)	Training time (s)
Prior-informed Volterra	25+25	3	13103	17.83	0.14	9.32	403
BS_Volterra	25+25	3	55	46.25	-2.16	24.09	2203
BS_Volterra_LM	57+57	3	119	9.94	0.61	5.14	2472
PNLSS_BS_Volterra_LM	6	3	160	43.53	-0.88	22.78	2474
BLA	6	–	49	14.49	-4.01	6.45	35
PNLSS_BLA_LM	6	3	833	**0.87**	0.02	0.46	1425
N4SID	6	–	49	66.99	-3.53	34.90	1
PNLSS_N4SID_LM	6	3	833	17.02	0.15	8.91	1428

(RRMSE_1 = 17.83%), which improves significantly after block-oriented structure extraction and LM-based refinement in P3 (RRMSE_1 = 9.94%). However, the subsequent realisation into a PNLSS model P4 leads to performance degradation (RRMSE_1 = 43.53%), likely due to model mismatch or overfitting during the nonlinear state-space fitting. In contrast, the baselines B2 and B4 yield substantially better results, with B2 achieving the best accuracy (RRMSE_1 = 0.87%) and B4 reaching 17.02%.

7.5.3 Nonlinear System Identification Benchmark

The Wiener–Hammerstein benchmark dataset introduced in Schoukens et al. (2009b) serves as a standard reference for evaluating nonlinear system identification methods. Past results using various modelling techniques are summarised in Wiener-Hammerstein Benchmark Results (2025). As per the benchmark protocol, $N = 100{,}000$ samples are allocated for training (see Figure 7.13), and performance is assessed using the root mean square error (RMSE) on a test set of $88{,}000$ samples, measured in millivolts. In this work, we adopt the same evaluation metric and use $78{,}000$ test samples to report results in Table 7.4 for comparison.

Table 7.4 summarises the performance of various modelling approaches on the WH benchmark dataset. Among all methods, the baseline model B2, based on BLA, achieves the best performance with an RMSE of 2 mV. The proposed PNLSS model P4 and the baseline B4, initialised using N4SID, exhibit comparable accuracy with RMSE values around 15 mV. Notably, the intermediate model P2 maintains the interpretable block-oriented structure with a significantly reduced parameter count, while still achieving acceptable prediction accuracy.

Figure 7.14 compares the estimated frequency responses of the two LTI blocks from the model P3 with the ground truth Chebyshev Type I and II filters. Due to the use of shape-penalised optimisation, the estimated filters closely match the true responses

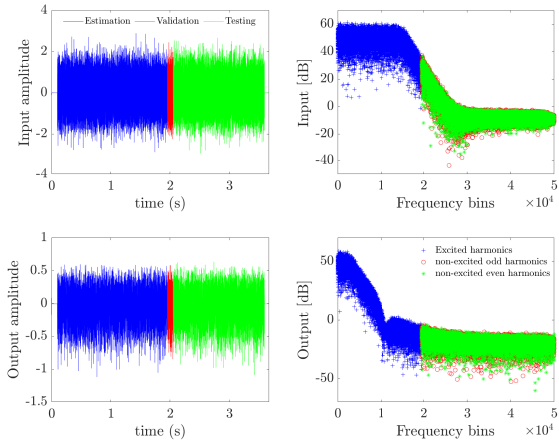


Figure 7.13: Training, validation, and test splits of the benchmark dataset.

Table 7.4: Performance on the Benchmark Dataset

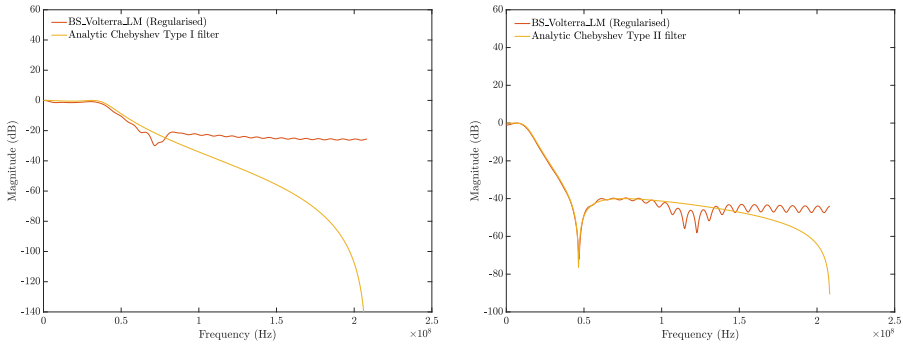
Wiener-Hammerstein System (Training sample: $N = 100{,}000$, Test sample: $79{,}000$, $\mathrm{RMS}(y_{\mathrm{test}}) = 247$ mV)

Methodology Details	M	D	No. of Parameters	RMSE (mV) ($\mathrm{RRMSE}_1\%$)	$\bar{\mu}$ (mV)	$\bar{\sigma}$ (mV)	Training time (s)
Prior-informed Volterra	20+20	3	7083	38 (15.54)	0.87	38.43	235
BS_Volterra	20+20	3	45	63 (25.37)	-36.71	50.91	3057
BS_Volterra_LM	52+52	3	109	10 (4.22)	3.27	9.91	3865
PNLSS_BS_Volterra_LM	6	3	160	**16 (6.39)**	4.83	15.06	3866
BLA	6	–	49	58 (23.46)	-37.65	44.15	837
PNLSS_BLA_LM	6	3	833	**2 (0.88)**	-0.03	2.18	7735
N4SID	6	–	49	57 (23.18)	-37.57	43.32	6
PNLSS_N4SID_LM	6	3	833	**15 (5.92)**	1.44	14.57	7267

across the entire frequency band, including in the stopband where deviations are typically more pronounced.

Figure 7.15 visualises the shape-enforced WH kernel up to third order. The first-order kernel, which corresponds to the convolution of the two LTI impulse responses, exhibits a smooth profile with a clear decay away from the diagonal, reflecting the system's finite memory. The second- and third-order kernels are also formed through structured convolutions involving the LTI blocks and the nonlinear function, resulting in banded, low-rank patterns with dominant values around specific lag combinations. This structure, reinforced by *prior-informed* regularisation, not only reduces the number of parameters but also enhances interpretability by aligning the estimated kernel with the true WH system dynamics.

The output prediction accuracy, training time, and model complexity of PNLSS models



LTI block 1 LTI block 2

Figure 7.14: FRFs of analytic 3rd-order Chebyshev low-pass type I and II filters compared with the estimated FRFs of Block-structured model P3. Refer to Section 7.3.

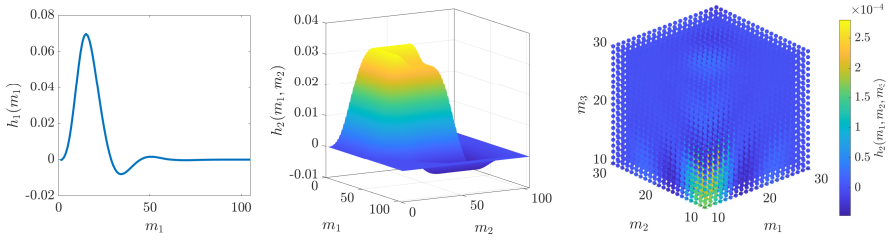


Figure 7.15: Visualisation of the shape-enforced WH kernel: First-, second-, and third-order components are shown separately. The third-order kernel is zoomed in on the lags between 10 and 30 to highlight its dominant region; distant entries are less significant (refer to Section 7.3).

are compared in Figure 7.16. The proposed Volterra-initialised PNLSS model P4 achieves prediction accuracy comparable to the PNLSS model B4 initialised from N4SID (RMSE = 15–16 mV), but with significantly fewer parameters and reduced training time. This efficiency is attributed to the *prior-informed* kernel structure that enables more compact and interpretable model realisations. However, the best performance is observed in the BLA-based model with 2 mV RMSE on the test set.

Figure 7.17 compares model performance on the benchmark system. The left plot shows validation error evolution over LM iterations, where the Volterra-initialised PNLSS model P4 shows minimal error. The right plot reports final test set prediction errors, confirming comparable accuracy across all PNLSS variants.

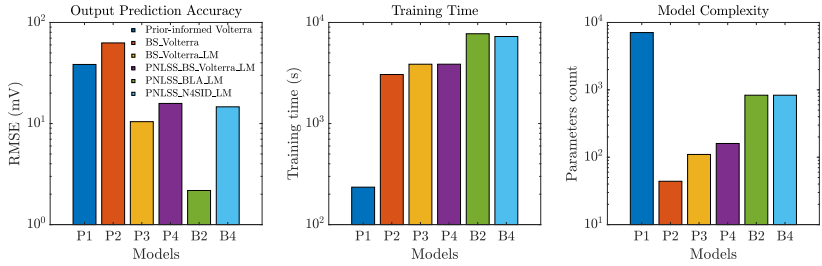


Figure 7.16: RMSE of output prediction in mV. The Volterra-based PNLSS model performs comparably to the PNLSS model initialised from N4SID, but requires fewer parameters and shorter training time.

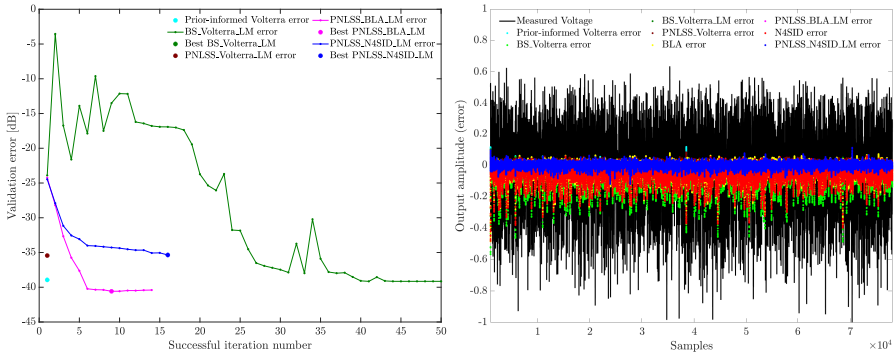


Figure 7.17: Left: Validation errors over LM optimisation iterations for the Benchmark system. Right: Output prediction error on test set.

7.6 Conclusion

This chapter presented a kernel-based identification framework for WH systems using *prior-informed* Volterra models. By constructing structured WH kernels and enforcing shape constraints, the dimensionality and complexity of the kernel were significantly reduced, enabling efficient model recovery. The optimised kernel was then mapped into a PNLSS representation through separate identification of the two LTI blocks and reconstruction of the static nonlinearity. Numerical simulations, experimental validation, and benchmark tests demonstrated that the proposed approach achieves accurate output prediction and parameter recovery while maintaining reduced model complexity and training time compared to conventional methods.

Chapter 8

Valorisation Plan

This chapter discusses the valorisation potential of the research presented in this dissertation. While the primary focus of the work was methodological, developing efficient and scalable techniques for nonlinear system identification, several outcomes of the project contribute to broader scientific and academic communities.

Specifically, two potential avenues of valorisation are highlighted. First, a publicly released measurement dataset supporting open research and benchmarking efforts in the field of system identification. Second, the methods developed during this research can form the basis of a software library to support practitioners and researchers working on data-driven modelling of nonlinear dynamical systems.

Although the research has not yet been commercialised or applied in an industrial setting, the outcomes are readily extensible to applied domains such as mechatronics, biomedical systems, or embedded control, where nonlinear dynamics play a key role. The methods developed offer the potential to reduce model complexity and improve identification accuracy in industrial workflows, thereby opening opportunities for future collaboration or technology transfer.

8.1 Sharing Research Tools and Data with the Community

This doctoral research resulted in three concrete outputs that can support the broader research community:

1. The implementation of subspace-based linear system identification of multiple poles in exponential polynomial-type signals;
2. The development of algorithmic tools and scripts for nonlinear system identification using Volterra series and low-rank approximation of tensor network structures;
3. The public release of a real-life measurement dataset suitable for benchmarking nonlinear modelling techniques.

Although these outputs have not yet been formally packaged into a stand-alone software library or used in commercial settings, they represent clear steps toward academic valorisation through open science, knowledge dissemination, and community reusability.

An important dissemination channel for this research has been the *Nonlinear Benchmark* platform (Wiener-Hammerstein Benchmark Results, 2025), which hosts a collection of community-driven benchmark systems designed to facilitate reproducible research in nonlinear system identification. A growing international academic network supports this platform and regularly organises workshops and seminars where researchers are invited to test their identification methods on standardised benchmark problems, submit their results using unified performance metrics, and make their code and findings publicly available.

As part of this PhD, the **Wiener-Hammerstein benchmark system** offered on the platform was used to evaluate the proposed identification modelling framework. A part of these results, published in Vijayan et al. (2025a), were presented at the Nonlinear System Identification Benchmark Workshop 2024 and subsequently published on the benchmark website, alongside a link to the GitHub repository containing the code used to generate the results (Wiener-Hammerstein Benchmark Results, 2025). While other researchers have not yet reused these contributions, they remain publicly available for comparison and reproduction, supporting the broader goals of transparency and method validation.

In addition, a **real-life measurement dataset** collected during this PhD has been released as open data. While the measurement setup itself existed before the dataset, it was carefully curated, structured, and documented for academic reuse. The dataset comprises input-output time series measurements from a nonlinear system and is suitable for evaluating system identification algorithms under realistic conditions. It is freely accessible via the IEEE DataPort[1] (Vijayan et al., 2025b). The data do not contain any personal or sensitive information and comply with ethical and legal standards, including General Data Protection Regulation (GDPR).

[1]Dataset: `https://ieee-dataport.org/documents/block-oriented-systems`, doi: 10.21227/v9j3-0z06

Together, these outputs represent an open and reproducible approach to methodological research. By sharing both algorithms and data, this work contributes to community-building within the system identification field and provides a foundation for future academic or applied extensions.

8.2 Towards a Reusable Software Pipeline

The methods developed during this PhD research have been implemented in a collection of modular scripts for 1. multiple pole identification, 2. nonlinear system identification, including routines for (*prior-informed*, regularised) Volterra kernel estimation, low-rank tensor approximations, shape-enforced block-structured model extraction, and nonlinear state-space realisation.

While not yet released as a formal software library, these tools are structured in a way that enables future packaging into an open-source toolbox (in MATLAB). The code used for the numerical simulations in Vijayan et al. (2023) is available via GitHub[2]. The code used for benchmark experiments, as presented in (Vijayan et al., 2025a), is already publicly available via GitHub and is also linked through the Nonlinear Benchmark website (Wiener-Hammerstein Benchmark Results, 2025). It can be accessed at GitHub repository[3].

The intended user base includes academic researchers, graduate students, and engineers working on system identification and data-driven modelling. A future library could significantly lower the entry barrier for applying advanced nonlinear modelling techniques in both academic and applied settings. The codebase was developed in MATLAB, using original implementations and built-in MATLAB toolboxes. While this limits direct open-source licensing under standard frameworks, the scripts can still be shared for academic use under a permissive research license.

8.3 Conclusion

Although the focus of this PhD was primarily methodological, its outcomes contribute to the broader scientific community by promoting reproducibility, openness, and reusability. The released dataset and shared benchmark results support comparative research, while the developed modelling tools have potential relevance in domains where accurate and interpretable nonlinear system models are needed, such as biomedical

[2]GitHub repository: `https://github.com/PrabhuVijayan/MultiplePolesEstimation.git`

[3]GitHub repository: `https://github.com/PrabhuVijayan/Low-rank-approximation-of-Prior-Informed-Volterra-Kernel`

devices, robotics, or embedded control. These contributions lay the foundation for further academic collaboration and possible industrial uptake in the future.

Chapter 9

Conclusions

This chapter summarises the main contributions and findings of the thesis, with a particular focus on two interconnected lines of research. Chapter 3 addressed the problem of parameter estimation in systems with repeated poles, proposing a robust method based on subspace techniques to handle multiplicities effectively. Chapters 5, 6, and 7 developed and validated a structured Volterra-based modelling framework for identifying Hammerstein, Wiener, and WH systems, respectively, ultimately enabling the estimation of PNLSS models.

Although these two strands may appear distinct, they are linked by their shared reliance on low-rank matrix approximation, subspace-based methodologies, and system-theoretic insights. The subspace framework provided the foundation for accurate linear dynamics estimation in both the multiple pole estimation case and the PNLSS realisation phase of the proposed nonlinear modelling framework. This chapter revisits the original research objectives and evaluates how effectively they were achieved through the contributions in the respective chapters. The chapter concludes with a reflection on the overall strengths and limitations of the work and identifies directions for future research.

9.1 Revisiting the Research Objectives

Having provided a broad summary of the thesis, we now reflect on the four specific research objectives introduced in Chapter 1. Each objective is examined in terms of how it was addressed, the degree of success achieved, and any limitations encountered. This retrospective view also highlights the methodological contributions and sets the stage for potential future extensions.

1. Estimation of Systems with Repeated Poles

The first research objective of this thesis was to develop a robust and interpretable method for estimating the parameters of systems with repeated poles, an important yet often overlooked challenge in system identification and harmonic retrieval. This was addressed in Chapter 3 by proposing a generalised model for damped exponential signals with multiple poles and extending the subspace-based estimation framework to accommodate pole multiplicities directly (Vijayan et al., 2023).

The method leverages SVD applied to the data matrix and a confluent Vandermonde structure to obtain stable estimates of the poles and associated amplitudes. To identify the correct multiplicity structure, a dual-strategy approach was proposed: sorting poles based on their magnitude and clustering them in the complex plane using the DBSCAN algorithm. This allowed for flexible and data-driven grouping of repeated poles, particularly in cases where phase variation made magnitude-based sorting insufficient. The final multiplicity structure was selected by minimising the mean squared error between the reconstructed and measured signal, ensuring that the chosen configuration best explained the observed dynamics.

The approach was shown to be highly effective in simulation studies, outperforming classical linear prediction techniques in terms of accuracy and robustness under moderate noise conditions. The subspace method's numerical stability, stemming from its reliance on SVD, contributed significantly to this performance. Real-data case studies further validated the method's practicality, especially in scenarios involving repeated complex-conjugate poles.

Despite its strengths, the method does exhibit some limitations. Accurate detection of multiplicity relies on the effectiveness of clustering in the presence of noise, and the DBSCAN algorithm requires careful tuning of its radius parameter. Additionally, while the proposed selection criterion based on reconstruction error is effective empirically, a theoretical analysis, such as deriving the Cramér–Rao bounds for the model, could offer deeper insight into estimation limits and confidence. Nevertheless, this contribution provides a valuable preprocessing step that improves the structural alignment and parameter accuracy in broader system identification tasks.

2. Structured Identification of Hammerstein Systems

The second major research objective of this thesis was to develop a structured, efficient, and interpretable modelling framework for Hammerstein systems using a Volterra-based approach (refer to Figure 1.6). This objective was addressed in Chapter 5, which laid out a stepwise methodology involving *prior-informed* Volterra kernel estimation, structured parameter extraction (Vijayan et al., 2025a), shape-constrained refinement, and final PNLSS realisation. Table 9.1 provides a symbolic comparison of each modelling stage against the baseline models initialised using linear approximation techniques, indicating where performance is better, worse, or approximately equal across key criteria such as

Table 9.1: Performance Summary of the Hammerstein System for Simulation (Sim.) and Measurement (Meas.) Datasets

Methodology Details	Output prediction accuracy		Parameter estimation accuracy		Training time		Parameter count	
	Sim.	Meas.	Sim.	Meas.	Sim.	Meas.	Sim.	Meas.
Prior-informed Volterra	≈	✓	✓	✓	✓	✓	✗	✗
BS_Volterra	≈	✓	✓	✓	✓	✓	≈	✗
BS_Volterra_LM	✓	✓	✓	✓	✓	✗	✗	✗
PNLSS_BS_Volterra_LM	≈	≈	–	–	✓	✗	≈	≈

Baseline: PNLSS models B2 & B4 initialised using BLA & N4SID, respectively.
Legend (compared to baseline): ✓ better; ≈ approximately equal; ✗ worse; – not applicable.

output prediction, parameter estimation, training time, and model complexity.

a. Volterra Kernel Estimation with Structural Priors: The first step of the framework successfully exploited prior knowledge about the Hammerstein architecture to constrain the Volterra kernel estimation problem. By incorporating the known structure of the system into the regression formulation, the dimensionality of the Volterra kernel was drastically reduced from exponential complexity I^D to a linear product form $D \cdot I$, where D is the degree and I corresponds to the memory length. This reduced complexity enabled fast estimation, with simulation runtimes consistently below one second, even for moderate sample sizes.

Despite the compact formulation, the *prior-informed* regularised Volterra kernel maintained prediction accuracy comparable to the baseline in simulation studies, and it significantly outperformed the baseline in real-world measurement datasets. An important added benefit was the interpretability: the estimated linear kernel component directly reflected the true dynamics of the underlying linear subsystem, offering insight that is often lost in black-box models.

b. Block-oriented Model Parameter Extraction and Refinement: The second step, extracting a Hammerstein block-oriented model from the Volterra kernel, was achieved by exploiting its low-rank structure. This decomposition resulted in a further reduction in parameter space, allowing for efficient estimation of the static nonlinearity and dynamic filter parameters. The block-oriented model achieved prediction and parameter estimation accuracy on par with the baseline in simulations, and better performance in measurements. Moreover, the estimated dynamic parameters remained physically interpretable.

To improve robustness and further enhance model accuracy, the extracted block-oriented model was refined through a nonlinear optimisation routine. Shape-based regularisation penalties were imposed during this optimisation to ensure stable, smooth, and decaying impulse responses. This refinement yielded models that outperformed the baseline in both simulation and experimental settings, particularly in output prediction accuracy and fidelity of the estimated dynamics.

c. PNLSS Model Realisation: The final step was the realisation of a PNLSS model from the refined Hammerstein block-oriented model. This transformation preserved the accuracy of the block-oriented model while converting it into a state-space form suitable for system-level analysis and control. While the output prediction accuracy of the resulting PNLSS model was comparable to that of a classically initialised PNLSS baseline, the identification procedure was significantly faster, often converging in under two seconds in simulation studies. The resulting PNLSS parameters were also more physically meaningful due to the informed initialisation, avoiding the instability and unpredictability that often arise from purely data-driven initial guesses.

3. Structured Identification of Wiener Systems

The third research goal of this thesis was to extend the Volterra-based modelling framework (Figure 1.6) to Wiener systems, which pose a distinct challenge due to their non-diagonal Volterra kernel structure. Unlike the Hammerstein case, the Wiener system lacks an explicit structural prior (zero-valued kernel elements) that could be embedded in the kernel during estimation. This lack of prior information significantly affects both computational efficiency and model compactness (Vijayan et al., 2025a). Nonetheless, Chapter 6 demonstrated that even in the absence of structural priors, a combination of tensor decomposition techniques and shape-regularised optimisation could lead to interpretable and competitive models. Table 9.2 summarises the comparative performance of different Wiener modelling stages.

Table 9.2: Performance Summary of the Wiener System for Simulation (Sim.), Measurement (Meas.), and Benchmark (Ben.) Datasets

Methodology Details	Output prediction accuracy			Parameter estimation accuracy			Training time			Parameter count		
	Sim.	Meas.	Ben.	Sim.	Meas.	Ben.	Sim.	Meas.	Ben.	Sim.	Meas.	Ben.
Volterra	✓	≈	✗	✓	✓	–	✗	✗	✓	✗	✗	✗
BS_Volterra	✗	✗	✗	✓	✓	–	✗	✗	≈	✓	≈	✓
BS_Volterra_LM	✓	≈	✓	✓	✓	–	✗	✗	≈	≈	✗	✓
PNLSS_BS_Volterra_LM	✓	✗	≈	–	–	–	✗	✗	≈	≈	≈	≈

Baseline: PNLSS models B2 & B4 initialised using BLA & N4SID, respectively.
Legend (compared to baseline): ✓ better; ≈ approximately equal; ✗ worse; – not applicable.

a. Volterra Kernel Estimation via Tensor Networks: The first step, Volterra kernel estimation, was achieved using regularised least squares along with tensor train decomposition. This representation allowed the high-dimensional Wiener kernel to be estimated with a manageable computational effort, despite the full kernel lacking structural sparsity. However, the dense nature of the Wiener kernel led to a significantly larger parameter count compared to the Hammerstein case, resulting in increased training time and reduced output prediction accuracy in simulation studies.

Importantly, the linear kernel obtained from the Volterra model still retained the ability to approximate the system's dynamic behaviour, which is critical for downstream model extraction. However, the lack of prior structural information made the estimated kernel less compact, and performance was slightly inferior to the baseline in both simulation and measurement data.

b. Block-oriented Model Extraction and Refinement: The second step, extracting and refining a Wiener block-oriented model from the Volterra kernel, was more successful. Using low-rank decomposition and shape-penalty-based optimisation, the dense kernel was reduced to a compact and interpretable block-oriented model. Shape constraints on the impulse response ensured stable and physically meaningful dynamics, while the nonlinear static component was estimated with reasonable accuracy.

This stage led to better output prediction and parameter estimation accuracy compared to the initial Volterra model, particularly in simulation experiments. However, due to the computationally expensive initial kernel estimation, this approach remained less efficient overall than the equivalent framework for Hammerstein systems.

c. PNLSS Model Realisation: The third step, realising a PNLSS model from the refined Wiener model, was achieved using subspace identification techniques. The resulting PNLSS models demonstrated comparable accuracy and complexity to those obtained via traditional initialisation methods. However, training time was notably longer, owing to the time-intensive low-rank decomposition and kernel estimation required earlier in the modelling stages.

Thus, while the final PNLSS performance was on par with the baseline, the practical advantage in efficiency and robustness seen in the Hammerstein case was diminished. Still, the interpretability and structure-aware nature of the model offer a meaningful benefit over black-box approaches.

d. Applicability to General Benchmark Systems: Interestingly, the proposed framework also showed promising results when applied to general nonlinear benchmark systems from the process industry, which do not conform to standard block-oriented

architectures. Despite the absence of any known internal structure, the framework still performed competitively, offering reasonable prediction accuracy and stability. This highlights the adaptability of the approach even beyond its intended structural assumptions.

4. Structured Identification of WH Systems

The fourth and final research objective was to extend the proposed modelling framework to WH systems, arguably the most complex among the considered block-oriented models due to their sandwich structure with two linear dynamic subsystems. Chapter 7 addressed this by introducing a *prior-informed* Volterra kernel formulation tailored to WH systems, followed by structured extraction, refinement (Vijayan et al., 2025a), and eventual PNLSS realisation. Table 9.3 summarises the performance and trade-offs across the WH identification modelling framework.

Table 9.3: Performance Summary of the WH System for Simulation (Sim.), Measurement (Meas.), and Benchmark (Ben.) Datasets

Methodology Details	Output prediction accuracy			Parameter estimation accuracy			Training time			Parameter count		
	Sim.	Meas.	Ben.	Sim.	Meas.	Ben.	Sim.	Meas.	Ben.	Sim.	Meas.	Ben.
Prior-informed Volterra	✗	✗	✗	–	–	–	≈	✓	✓	✗	✗	✗
BS_Volterra	✗	✗	✗	✓	✓	✓	≈	✗	✓	✓	✓	✓
BS_Volterra_LM	✓	≈	≈	✓	✓	✓	≈	✗	✓	✓	✓	✓
PNLSS_BS_Volterra_LM	≈	✗	≈	–	–	–	≈	✗	✓	✓	✓	✓

Baseline: PNLSS models B2 & B4 initialised using BLA & N4SID, respectively.
Legend (compared to baseline): ✓ better; ≈ approximately equal; ✗ worse; – not applicable.

a. Volterra Kernel Estimation with WH Structural Priors: The WH system's Volterra kernel exhibits a specific banded structure due to the cascade of two LTI blocks. This prior knowledge was embedded during kernel estimation, resulting in low-rank matrices that are shifted and overlaid to form the overall banded kernel. The identification exploited this shifting-lag pattern to efficiently assemble the structured Volterra kernel, capturing the system behaviour with significantly fewer parameters. As a result, the training time was substantially reduced compared to estimating a full, dense Volterra kernel, while still preserving the essential characteristics of the WH system dynamics.

However, unlike the Hammerstein case, the output prediction accuracy of the *prior-informed* WH-Volterra model did not surpass the baseline in simulation, though it showed comparable or improved performance in measurements and benchmark experiments. A key limitation is that the linear component of the kernel represents a convolution of the two LTI blocks, making it impossible to directly recover each LTI

subsystem individually. This is in contrast to Hammerstein or Wiener systems, where the single LTI block could be recovered directly. Thus, while the kernel captures the overall dynamic behaviour, it does not offer full subsystem identifiability at this stage.

b. Block-oriented Model Extraction and Refinement Despite the above limitation, the second step, the extraction of a structured block-oriented model, was effectively realised. The WH block-oriented model was derived from the *prior-informed* Volterra kernel, reducing the parameter space further while maintaining a modular architecture. The nonlinearity and both LTI blocks were estimated using the banded structure, followed by nonlinear optimisation with shape-penalty constraints on both LTI impulse responses to enforce stability and physical realism.

The refinement step successfully improved both prediction accuracy and parameter estimation, particularly in measurement and benchmark scenarios. Importantly, the model retained a smaller number of parameters than the baseline while achieving comparable or better performance. That said, the well-known scaling ambiguity between the two LTI blocks (a consequence of the serial structure) remains a theoretical limitation. However, for practical purposes, the model approximated the system dynamics accurately.

c. PNLSS Model Realisation: The third step, translating the refined WH block-oriented model into a PNLSS representation, was accomplished using subspace identification of the two dynamic parts and polynomial estimation of the nonlinear terms. The resulting PNLSS models maintained comparable output prediction accuracy to the baseline model across all test cases, but with a smaller parameter count. Moreover, the training time remained competitive due to the prior model structure and refinement guiding the initialisation.

9.2 Scope for Future Research

While this work focuses on canonical block-oriented SISO structures, the proposed methodology creates opportunities for future extensions and methodological development.

1. Consideration of repeated poles in the LTI blocks of block-oriented models: The methodology proposed in Chapter 3 for estimating repeated poles in LTI systems using subspace-based identification can be extended to the nonlinear identification frameworks developed in Chapters 5–7. In particular, incorporating pole multiplicity information into the estimation of Volterra kernels and subsequent

block-oriented model parametrisation (Hammerstein, Wiener, and WH) could lead to more accurate and physically meaningful representations of systems whose underlying linear dynamics exhibit near or exact pole repetitions. Such an extension would enable the identification of block-oriented models that better capture resonant or weakly damped behaviours often observed in real-world systems. Moreover, accounting for pole multiplicity in the regularisation or kernel reconstruction steps could improve numerical stability and reduce model bias in the presence of ill-conditioned system dynamics.

2. Selection of appropriate model structure and hyperparameters: In this thesis, for Hammerstein, Wiener, and WH block-oriented models, it has been assumed that the model structure (i.e. which block type), the model order of the linear subsystem, and the memory (dynamics length) are known a priori. Additionally, the nonlinearity in the system is modelled with a polynomial of prescribed degree. However, in practice, these must often be identified from data. Future work could develop systematic methods to automatically select which block-oriented structure is most appropriate (Hammerstein vs. Wiener vs. WH vs. possibly parallel branch or feedback), the order/complexity (model order of linear blocks, memory length in Volterra kernels) of the linear dynamics, the degree (or basis) and form of the static nonlinearity, regularisation/hyperparameter settings in kernel estimation or block-oriented model fitting that balance bias vs variance, and also enforce interpretability.

Several strategies have been proposed in the literature to aid this process. For instance, Schoukens et al. (2015) introduced a method based on analysing variations in the BLA across operating conditions to distinguish between Hammerstein, Wiener, and WH structures. Similarly, the work Blanco-Murillo and Yagüe-Jiménez (2018) proposed a data-driven strategy for selecting Volterra kernel memory and polynomial degree using designed excitation signals.

Integrating such structure-discrimination and parameter-selection strategies with the Volterra kernel-based identification framework developed in this thesis could enable its application to scenarios where the underlying system structure is not known in advance. Other possible avenues include using information criteria (Akaike, Bayesian), cross-validation, or Bayesian/hierarchical methods that put priors over structure or hyperparameters. Incorporation of physics (if known) to constrain possible structures (e.g. knowing if actuator nonlinearity is input-side or output-side) is also promising. This would make the proposed methodology more autonomous, robust, and broadly applicable in real-world nonlinear system identification tasks.

3. Extension to Parallel and MIMO Block-oriented Systems: The proposed framework can be extended to identify parallel and multi-input, multi-output (MIMO) systems. In many real-world applications, nonlinear systems exhibit multiple parallel

pathways or interact through multiple inputs and outputs. Extending the current methodology to such systems will require structured Volterra representations that capture cross-channel interactions, efficient tensor decompositions to manage the rapidly growing parameter space, and the design of priors that encode the interdependencies between subsystems.

The existing tensor-based and block-structured formulations offer a promising starting point for this generalisation. However, such extensions introduce new challenges. In the case of parallel block-oriented systems, one key complication is that the linear dynamics of individual branches combine additively in the overall Volterra kernel. As a result, the superposition of impulse responses obscures the identity of the individual subsystems, making it difficult to disentangle or uniquely recover the dynamics of each parallel path without additional structural or parametric priors. This limitation becomes particularly significant when the goal is not only accurate input–output prediction but also true subsystem parameter recovery.

For MIMO systems, the complexity increases further. The Volterra kernels for MIMO systems are high-dimensional tensors that capture interactions across multiple inputs and outputs, often with intricate cross-channel dependencies. Identifying such systems requires careful modelling of both within-channel dynamics and inter-channel couplings, and the parameter space grows rapidly with system size. Designing low-rank and structured representations that scale well with the number of inputs and outputs, while remaining identifiable and computationally tractable, remains an open research challenge.

4. Unified Kernel Estimation with Integrated Priors: In this work, the Volterra kernel estimation process incorporates structural prior knowledge (for Hammerstein and WH), regularisation, and low-rank structure enforcement sequentially. A natural direction for future work is to develop a unified estimation algorithm that jointly incorporates structural priors derived from assumed block-oriented model forms, shape-enforcing penalties to encourage smooth and decaying dynamics, and low-rank constraints to reduce parameter complexity and improve robustness. Such a joint formulation might involve structured nuclear norm regularisation or constrained convex optimisation, though further investigation is needed to evaluate feasibility and tractability.

5. Extension to Block-oriented Systems with Feedback: The present work has focused on standard block-oriented models without feedback, where the underlying nonlinearity and dynamics can be effectively represented using the Volterra series framework. A promising yet largely unexplored research direction is to extend this methodology to systems that include feedback loops, such as simple feedback, WH feedback, or linear fractional representation (LFR) structures. These architectures

appear frequently in practice, for instance, in servo and motion control systems with friction or backlash (Wu et al., 2014), power amplifiers employing nonlinear feedback for distortion compensation (Tikhonov and Abdipour, 2023), biochemical and physiological control systems with nonlinear regulation (Ponsiglione et al., 2023), and mechanical structures with hysteretic feedback (Ruderman and Bertram, 2015). In such cases, the Volterra representation becomes inadequate due to its inherent open-loop assumption Paduart (2008), and alternative representations such as (polynomial) NARX or, more generally, Nonlinear Autoregressive Moving Average model with exogenous inputs (NARMAX) could be employed to capture the nonlinear feedback dynamics more accurately (Ali et al., 2021; Boussaada et al., 2018). Unlike Volterra models, NARX models incorporate past outputs as regressors, allowing them to capture internal feedback effects and achieve a more compact representation (Zhu and Lang, 2020; Xiao et al., 2014). This could lead to improved identifiability and scalability when extending the current framework to complex nonlinear dynamical systems.

However, several open questions remain. The structure and properties of NARX/NARMAX kernels are not yet fully understood, particularly regarding their potential low-rank characteristics or their correspondence with interpretable block-oriented structures containing feedback. Moreover, it is unclear how an equivalent PNLSS parametrisation could be derived from such kernels, as the theoretical equivalence between Volterra, block-oriented, and PNLSS representations has so far been established only for standard Hammerstein, Wiener, and WH models. Investigating these relationships could therefore provide a pathway to generalising the proposed Volterra-based identification framework to a broader class of nonlinear dynamical systems with feedback.

A practical starting point could be to leverage the Volterra-based kernel estimation and regularisation framework developed in this thesis to initialise NARX-type models, treating the feedback terms as additional regressors and extending the tensor-based representations to include delayed outputs. The structural insights gained from the equivalence between Volterra, block-oriented, and PNLSS models could further guide the formulation of low-rank or regularised NARX kernels, enabling a systematic exploration of feedback nonlinearities while maintaining interpretability and computational tractability.

6. Generalisation to Non-Polynomial Nonlinearities: The current framework is limited to systems where the nonlinearity can be described or approximated using polynomials. While this includes a broad class of systems, many practical systems involve more complex nonlinearities (e.g., saturation, dead zone). Future work could involve incorporating non-parametric representations such as splines or kernel functions, and using neural networks or deep learning architectures to model static nonlinearities, while retaining the interpretability and modularity of the current approach.

7. Comparison with Neural Network-based Modelling Approaches: An alternative to the Volterra and PNLSS models developed in this thesis is the use of neural network architectures for nonlinear system identification. Feedforward and, by extension, cascaded forward neural networks can serve as universal function approximators (Hornik et al., 1989; Csáji, 2001; Scarselli and Chung Tsoi, 1998), analogous to the Volterra expansion in representing static or weakly dynamic nonlinearities. Recurrent architectures, such as RNNs and LSTMs, explicitly capture system dynamics through internal states, in a manner conceptually similar to the state evolution in PNLSS models (Ljung et al., 2020).

Neural networks often achieve high prediction accuracy and can flexibly model complex nonlinear behaviours; however, they typically require large training datasets, extensive tuning of hyperparameters, and regularisation strategies to ensure stability and generalisation. More importantly, their parameters usually lack physical interpretability, making it difficult to relate model parameters to system characteristics such as poles, gains, or saturation effects. In contrast, the Volterra- and PNLSS-based framework developed in this thesis provides structured, interpretable models that directly link to system physics, allowing for explicit control over stability, linear dynamics, and nonlinearity.

Future work could explore hybrid modelling strategies that combine the interpretability and structural insights of Volterra and PNLSS models with the flexibility of data-driven neural networks, for instance, by using neural architectures to learn residual dynamics or approximate unmodelled nonlinearities.

8. Autoencoder perspective on the proposed modelling framework: A promising future direction lies in formalising and exploiting the structural equivalence between the proposed modelling framework and an autoencoder (AE) architecture. In this analogy, the encoder maps the high-dimensional Volterra kernel representation to a compact latent space corresponding to a block-oriented model. At the same time, the decoder reconstructs a flexible PNLSS model from this latent representation. Future work could explore the theoretical conditions under which this mapping is bijective, and implement data-driven AE variants that learn such latent representations from input-output trajectories. Incorporating physical priors and low-rank constraints within the AE could further ensure interpretability and preserve correspondence with the underlying system dynamics. This line of research may enable automated discovery of interpretable nonlinear model structures and bridge the gap between physics-based identification and deep representation learning.

A natural starting point would be to use the Volterra-to-block-to-PNLSS mappings developed in this thesis as a supervised initialisation for training the encoder and decoder networks, providing a physics-informed baseline that constrains the AE to produce interpretable latent representations.

9. Extension Toward Control Applications: Another interesting future direction is to extend the proposed system identification framework toward nonlinear control design. The identified Volterra- and PNLSS-based models provide accurate, low-order representations that can be directly exploited for model-based predictive control (MPC), feedback linearization, or gain-scheduling strategies. By leveraging the block-oriented structure (Hammerstein, Wiener, or WH), one can design controllers that explicitly compensate for static nonlinearities while maintaining tractable linear dynamics for stability and performance analysis. Moreover, incorporating regularised or data-driven model inversion could enable feedforward control or nonlinear disturbance rejection. Future research may also explore adaptive control schemes using the identified models for systems with slowly varying nonlinearities, or employ tensor-based control synthesis exploiting the compact TT representation to handle high-dimensional nonlinear plants efficiently.

These extensions can be initiated by using the identified PNLSS models as surrogate plants in simulation-based controller design, where their compact and block-structured representations allow efficient computation of gradients, model inversion, and predictive control laws.

Appendix A

Tensor Preliminaries

A.1 Introduction to Tensors

A tensor is a multidimensional array of numbers. More formally, a tensor $\mathcal{H} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_D}$ is said to be of order D, where each entry is indexed by D integers $(i_1, i_2, \ldots, i_D)$. Familiar examples of tensors in system identification include: scalars ($D = 0$), vectors ($D = 1$), matrices ($D = 2$). A D-dimensional or D-way tensor $\mathcal{H}$ is called symmetric, if all its dimensions are equal, i.e., $I_1 = I_2 = \cdots = I_D$, and if its entries $\mathcal{H}(i_1, i_2, \ldots, i_D)$ remain invariant under any permutation of indices $\pi(i_1, i_2, \ldots, i_D)$, where the operator $\pi(.)$ denotes permutation.

Following MATLAB's convention, indices are 1-based (starting from 1). Lowercase letters denote indices (e.g., i_1, i_2), while uppercase letters denote the corresponding dimension sizes (e.g., I_1, I_2). For instance, the entry of a matrix $\boldsymbol{H} \in \mathbb{R}^{M \times N}$ at row m and column n is written as $\boldsymbol{H}(m, n)$.

A.1.1 Tensor Unfolding (Matricisation)

Definition A.1.1 (Mode-k Unfolding)**.** Given a tensor $\mathcal{H} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_k \times \cdots \times I_D}$, the mode-$k$ unfolding or matricisation of $\mathcal{H}$, denoted by $\mathcal{H}_{(k)}$, is a matrix in $\mathbb{R}^{I_k \times (I_1 \ldots I_{k-1} I_{k+1} \ldots I_D)}$. This operation rearranges mode-k fibres (obtained by fixing all indices except the kth) into the columns of a matrix. The column ordering follows a column-major order (i.e., varying the first index fastest).

Example 21 (Unfolding a third-order tensor). Let $\mathcal{H} \in \mathbb{R}^{4\times3\times2}$ be defined by its frontal slices:

$$\mathcal{H}(:,:,1) = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} \quad \mathcal{H}(:,:,2) = \begin{bmatrix} 13 & 14 & 15 \\ 16 & 17 & 18 \\ 19 & 20 & 21 \\ 22 & 23 & 24 \end{bmatrix}.$$

Mode-1 unfolding $\mathcal{H}_{(1)} \in \mathbb{R}^{4\times6}$ arranges mode-1 fibres (i.e., vectors obtained by fixing all indices except the first) as columns:

$$\mathcal{H}_{(1)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix}. \tag{A.1}$$

Mode-2 unfolding $\mathcal{H}_{(2)} \in \mathbb{R}^{3\times8}$ arranges mode-2 fibres as columns:

$$\mathcal{H}_{(2)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix}. \tag{A.2}$$

Mode-3 unfolding $\mathcal{H}_{(3)} \in \mathbb{R}^{2\times12}$ arranges mode-3 fibres as columns:

$$\mathcal{H}_{(3)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 2 & 5 & 8 & 11 & 3 & 6 & 9 & 12 \\ 13 & 16 & 19 & 22 & 14 & 17 & 20 & 23 & 15 & 18 & 21 & 24 \end{bmatrix}. \tag{A.3}$$

A.1.2 Tensor Reshaping and Vectorisation

Definition A.1.2 (Reshaping and Matricisation)**.** Reshaping changes the dimensions of a tensor while preserving its data ordering (usually in column-major form). A common case is matricisation, which reshapes a tensor into a matrix.

In MATLAB, the command $\mathrm{reshape}(\mathcal{H}, [I_1, I_2, \ldots, I_D])$ reshapes the D-way tensor $\mathcal{H}$ into the new dimensions, assuming the total number of elements is preserved:

$$\prod_{d=1}^{D} I_d = \mathrm{numel}(\mathcal{H}),$$

where the operator $\mathrm{numel}(\cdot)$ returns the number of elements in the tensor.

Vectorisation is a special case of reshaping where a tensor is flattened into a vector using multi-indexing:

$$i = [i_1 i_2 \dots i_D] = i_1 + \sum_{d=2}^{D} (i_d - 1) \prod_{l=1}^{d-1} I_l. \tag{A.4}$$

This defines a one-to-one mapping from $(i_1, i_2, \dots, i_D)$ to a single linear index $i \in 1, \dots, I$, where $I = I_1 I_2 \dots I_D$. The vectorised form $\mathrm{vec}(\mathcal{H})$ arranges these elements linearly. The same indexing formula in (A.4) can also be used in reverse to recover the original multi-index from a single index, which is useful when reshaping a tensor back to its original dimensions.

Example 22. For the tensor in Example 21, we have:

$$\mathrm{vec}(\mathcal{H}) = \mathrm{reshape}(\mathcal{H}, [24, 1]) = \begin{bmatrix} 1 & 2 & \dots & 24 \end{bmatrix}^\top.$$

A.1.3 Tensor-Matrix Products, Vector Contractions, and the Kronecker Product

Definition A.1.3 (Mode-k Tensor-Matrix product)**.** Consider $\mathcal{H} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_k \times \dots \times I_D}$ a D-way tensor and $\boldsymbol{U} \in \mathbb{R}^{J \times I_k}$ be a matrix. The mode-k product of $\mathcal{H}$ with $\boldsymbol{U}$ is a tensor $\mathcal{H}' \in \mathbb{R}^{I_1 \times \dots \times I_{k-1} \times J \times I_{k+1} \times \dots \times I_D}$, defined as:

$$(\mathcal{H} \bullet_k \boldsymbol{U})_{i_1, \dots, i_{k-1}, j, i_{k+1}, \dots, i_D} = \sum_{k=1}^{I_k} u_{j, i_k} h_{i_1, \dots, i_k, \dots, i_D}. \tag{A.5}$$

Definition A.1.4 (Multilinear Contraction with a vector)**.** Let $\mathcal{H}$ be a symmetric D-way tensor of dimensions $I \times I \times \dots I$ (D times) and let $\boldsymbol{x} \in \mathbb{R}^I$ be a vector. The contraction of $\mathcal{H}$ with $\boldsymbol{x}$ in all modes is:

$$\mathcal{H} \underbrace{(\boldsymbol{x}, \dots, \boldsymbol{x})}_{D \text{ times}} = \mathcal{H} \bullet_1 \boldsymbol{x}^\top \bullet_2 \boldsymbol{x}^\top \bullet_3 \dots \bullet_D \boldsymbol{x}^\top. \tag{A.6}$$

This results in a scalar, which can also be interpreted as a homogeneous polynomial of degree D in the entries of $\boldsymbol{x}$.

A fundamental identity involving the vectorisation operator is the following:

$$\mathrm{vec}(\mathcal{H} \bullet_1 \boldsymbol{U}_1 \bullet_2 \cdots \bullet_D \boldsymbol{U}_D) = (\boldsymbol{U}_D \otimes \cdots \otimes \boldsymbol{U}_1).\mathrm{vec}(\mathcal{H}), \tag{A.7}$$

where the Kronecker product reverses the order of the mode-wise multiplications. This identity is particularly useful in simplifying computations and rewriting Definition A.1.4 as:

$$\mathcal{H}\,\underbrace{(\boldsymbol{x}, \ldots, \boldsymbol{x})}_{D \text{ times}} = \mathrm{vec}(\mathcal{H})^\top \underbrace{(\boldsymbol{x} \otimes \cdots \otimes \boldsymbol{x})}_{D \text{ times}}, \tag{A.8}$$

which expresses the contraction of a tensor $\mathcal{H}$ with a rank-1 tensor (outer product of D vectors) as an inner product between two vectors.

Bibliography

M. N. Alghafir and J. F. Dunne. A NARX Damper Model for Virtual Tuning of Automotive Suspension Systems with High-Frequency Loading. *Vehicle System Dynamics*, 50(2):167–197, 2012. doi: 10.1080/00423114.2011.575946.

W. Ali, W. U. Khan, M. A. Z. Raja, Y. He, and Y. Li. Design of Nonlinear Autoregressive Exogenous Model Based Intelligence Computing for Efficient State Estimation of Underwater Passive Target. *Entropy*, 23(5):550, 2021. doi: 10.3390/e23050550.

B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Dover Publications, 2012.

A. Antoniou. *Digital Signal Processing: Signals, Systems, and Filters*. McGraw-Hill, New York, 2006.

P. Antsaklis and A. Michel. *Linear Systems*. Birkhäuser Boston, 2005.

E.-W. Bai. Frequency Domain Identification of Wiener Models. *Automatica*, 39(9): 1521–1530, 2003. doi: 10.1016/S0005-1098(03)00149-3.

E.-W. Bai. *Frequency Domain Identification of Hammerstein Models*, pages 161–180. Springer London, 2010a. doi: 10.1007/978-1-84996-513-2_11.

E.-W. Bai. *Decoupling the Linear and Nonlinear Parts in Hammerstein Model Identification*, pages 335–345. Springer London, 2010b. doi: 10.1007/978-1-84996-513-2_20.

E.-W. Bai, Z. Cai, S. Dudley-Javorosk, and R. K. Shields. Identification of a Modified Wiener–Hammerstein System and its Application in Electrically Stimulated Paralyzed Skeletal Muscle Modeling. *Automatica*, 45(3):736–743, 2009. doi: 10.1016/j.automatica.2008.09.023.

N. N. Balaji, S. Lian, M. Scheel, M. R. W. Brake, P. Tiso, J.-P. Noël, and M. Krack. Numerical Assessment of Polynomial Nonlinear State-Space and Nonlinear-Mode Models for Near-Resonant Vibrations. *Vibration*, 3(3):320–342, 2020. doi: 10.3390/vibration3030022.

H. Barkhuijsen, R. D. Beer, and D. V. Ormondt. Improved Algorithm for Noniterative Time-domain Model Fitting to Exponentially Damped Magnetic Resonance Signals. *Journal of Magnetic Resonance*, 73(3):553–557, 1987. doi: 10.1016/0022-2364(87)90020-4.

K. Batselier. Enforcing Symmetry in Tensor Network MIMO Volterra Identification. *IFAC-PapersOnLine*, 54(7):469–474, 2021. doi: 10.1016/j.ifacol.2021.08.404.

K. Batselier. A Khatri-Rao Product Tensor Network for Efficient Symmetric MIMO Volterra Identification. *IFAC-PapersOnLine*, 56(2):7282–7287, 2023. doi: 10.1016/j.ifacol.2023.10.339.

K. Batselier, Z. Chen, and N. Wong. Tensor Network Alternating Linear Scheme for MIMO Volterra System Identification. *Automatica*, 84:26–35, 2017. doi: 10.1016/j.automatica.2017.06.033.

K. Batselier, C.-Y. Ko, and N. Wong. Tensor Network Subspace Identification of Polynomial State Space Models. *Automatica*, 95:187–196, 2018. doi: 10.1016/j.automatica.2018.05.015.

Z. Ben Ahmed, G. Favier, and N. Derbel. Tensor-based Methods for Wiener-Hammerstein System Identification. In *10th International Multi-Conferences on Systems, Signals & Devices*, pages 1–6, 2013. doi: 10.1109/SSD.2013.6564100.

N. Bershad, P. Celka, and S. Mc Laughlin. Analysis of Stochastic Gradient Identification of Wiener-Hammerstein Systems for Nonlinearities with Hermite Polynomial Expansions. *IEEE Transactions on Signal Processing*, 49(5):1060–1072, 2001. doi: 10.1109/78.917809.

S. A. Billings. Identification of Nonlinear Systems – A Survey. *IEE Proceedings D (Control Theory and Applications)*, 127:272–285, 1980. doi: 10.1049/ip-d.1980.0047.

S. A. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency and Spatio-Temporal Domains*. Wiley, 2013.

S. A. Billings and S. Y. Fakhouri. Identification of a Class of Nonlinear Systems using Correlation Analysis. *Proceedings of the Institution of Electrical Engineers*, 125:691–697, 1978. doi: 10.1049/piee.1978.0161.

G. Birpoutsoukis, A. Marconato, J. Lataire, and J. Schoukens. Regularized Nonparametric Volterra Kernel Estimation. *Automatica*, 82:324–327, 2017. doi: 10.1016/j.automatica.2017.04.014.

G. Birpoutsoukis, P. Z. Csurcsia, and J. Schoukens. Efficient Multidimensional Regularization for Volterra Series Estimation. *Mechanical Systems and Signal Processing*, 104:896–914, 2018. doi: 10.1016/j.ymssp.2017.10.007.

S. Bittanti and L. Piroddi. Nonlinear Identification and Control of a Heat Exchanger: A Neural Network Approach. *Journal of the Franklin Institute*, 334(1):135–153, 1997. doi: 10.1016/S0016-0032(96)00059-2.

J. L. Blanco-Murillo and V. Yagüe-Jiménez. A Method for Informed Selection of Memory-length and Nonlinearity-order Parameters in Volterra-Wiener Systems from Exponential Sweep Excitations. *Multidimensional Systems and Signal Processing*, 29:1861–1893, 2018. doi: 10.1007/s11045-017-0535-3.

Z. Boussaada, O. Curea, A. Remaci, H. Camblong, and N. Mrabet Bellaaj. A Nonlinear Autoregressive Exogenous (NARX) Neural Network Model for the Prediction of the Daily Direct Solar Radiation. *Energies*, 11(3):620, 2018. doi: 10.3390/en11030620.

S. Boyd and L. Chua. Fading Memory and the Problem of Approximating Nonlinear Operators with Volterra Series. *IEEE Transactions on Circuits and Systems*, 32(11): 1150–1161, 1985. doi: 10.1109/TCS.1985.1085649.

S. L. Brunton, J. L. Proctor, and J. N. Kutz. Sparse Identification of Nonlinear Dynamics with Control (SINDYc). *IFAC-PapersOnLine*, 49(18):710–715, 2016. doi: 10.1016/j.ifacol.2016.10.249.

M. Champneys, G. I. Beintema, R. Tóth, M. Schoukens, and T. J. Rogers. Baseline Results for Selected Nonlinear System Identification Benchmarks. *IFAC-PapersOnLine*, 58(15):474–479, 2024. doi: 10.1016/j.ifacol.2024.08.574.

C.-T. Chen. *Linear System Theory and Design*. HRW Series in Electrical and Computer Engineering. Holt, Rinehart, and Winston, 1984.

C. J. Clark, G. Chrisikos, M. S. Muha, A. A. Moulthrop, and C. P. Silva. Time-domain Envelope Measurement Technique with Application to Wideband Power Amplifier Modeling. *IEEE Transactions on Microwave Theory and Techniques*, 46 (12):2531–2540, 1998. doi: 10.1109/22.739245.

P. Crama and J. Schoukens. Initial Estimates of Wiener and Hammerstein Systems using Multisine Excitation. *IEEE Transactions on Instrumentation and Measurement*, 50(6):1791–1795, 2001. doi: 10.1109/19.982982.

P. Crama and J. Schoukens. Computing an Initial Estimate of a Wiener-Hammerstein System with a Random Phase Multisine Excitation. *IEEE Transactions on Instrumentation and Measurement*, 54(1):117–122, 2005. doi: 10.1109/TIM.2004.838130.

B. C. Csáji. *Approximation with Artificial Neural Networks*. PhD thesis, Eötvös Loránd University, Faculty of Sciences, 2001.

B. De Moor. DaISy: Database for the Identification of Systems. Department of Electrical Engineering, ESAT/STADIUS, KU Leuven, 2000. `homes.esat.kuleuven.be/~smc/daisy/` (Used datasets: 96-014, 97-002, 98-002).

C. De Persis and P. Tesi. Formulas for Data-Driven Control: Stabilization, Optimality, and Robustness. *IEEE Transactions on Automatic Control*, 65(3):909–924, 2020. doi: 10.1109/TAC.2019.2959924.

O. Debals, L. De Lathauwer, and M. Van Barel. *Tensorization and Applications in Blind Source Separation*. PhD thesis, KU Leuven, 2017.

J. Decuyper, K. Tiels, and J. Schoukens. PNLSS Toolbox 1.0. *IFAC-PapersOnLine*, 54 (7):374–378, 2021. doi: 10.1016/j.ifacol.2021.08.388.

T. D'haene, R. Pintelon, J. Schoukens, and E. Van Gheem. Variance Analysis of Frequency Response Function Measurements using Periodic Excitations. *IEEE Transactions on Instrumentation and Measurement*, 54(4):1452–1456, 2005. doi: 10.1109/TIM.2005.851075.

L. Di Natale, M. Zakwan, B. Svetozarevic, P. Heer, G. Ferrari-Trecate, and C. N. Jones. Stable Linear Subspace Identification: A Machine Learning Approach. In *2024 European Control Conference (ECC)*, pages 3539–3544, 2024. doi: 10.23919/ECC64448.2024.10590843.

P. Dymarski. *Hidden Markov Models*. IntechOpen, Rijeka, 2011. doi: 10.5772/601.

M. Enqvist. *Linear Models of Nonlinear Systems*. PhD thesis, Linköping University, 2005.

M. Enqvist and L. Ljung. Linear Approximations of Nonlinear FIR Systems for Separable Input Processes. *Automatica*, 41(3):459–473, 2005. doi: 10.1016/j.automatica.2004.11.016.

E. Eskinat, S. H. Johnson, and W. L. Luyben. Use of Hammerstein Models in Identification of Nonlinear Systems. *AIChE Journal*, 37(2):255–268, 1991. doi: 10.1002/aic.690370211.

J. Espinosa, J. Vandewalle, and V. Wertz. *Fuzzy Logic, Identification and Predictive Control (Advances in Industrial Control)*. Springer-Verlag, 2004.

G. Favier and A. Kibangou. Tensor-Based Approaches for Nonlinear and Multilinear Systems Modeling and Identification. *Algorithms*, 16(9), 2023. doi: 10.3390/a16090443.

D. Fey and E. Bullinger. Limiting the Parameter Search Space for Dynamic Models with Rational Kinetics using Semi-definite Programming. *IFAC Proceedings Volumes*, 43 (6):150–155, 2010. doi: 10.3182/20100707-3-BE-2012.0088.

M. Fira, H. N. Costin, and L. Goraş. On the Classification of ECG and EEG Signals with Various Degrees of Dimensionality Reduction. *Biosensors*, 11(5), 2021. doi: 10.3390/bios11050161.

K. P. Fruzzetti, A. Palazoğlu, and K. A. McDonald. Nolinear Model Predictive Control using Hammerstein Models. *Journal of Process Control*, 7(1):31–41, 1997. doi: 10.1016/S0959-1524(97)80001-B.

H. O. Garcés, J. M. Palma, A. J. Rojas, and V. Valdebenito. Model Analysis of Heat Transfer by Hammerstein Systems and Optical Instrumentation. *IFAC-PapersOnLine*, 52(1):442–447, 2019. doi: 10.1016/j.ifacol.2019.06.102.

F. Giri and E.-W. Bai. *Block Oriented Nonlinear System Identification*. Springer, 2010. doi: 10.1007/978-1-84996-513-2.

F. Giri, Y. Rochdi, and F. Z. Chaoui. An Analytic Geometry Approach to Wiener System Frequency Identification. *IEEE Transactions on Automatic Control*, 54(4): 683–696, 2009. doi: 10.1109/tac.2009.2014915.

W. Greblicki. Nonparametric Identification of Wiener Systems. *IEEE Transactions on Information Theory*, 38(5):1487–1493, 1992. doi: 10.1109/18.149500.

W. Greblicki. Nonlinearity Estimation in Hammerstein Systems Based on Ordered Observations. *IEEE Transactions on Signal Processing*, 44(5):1224–1233, 1996. doi: 10.1109/78.502334.

J. C. Gómez and A. Jutan. Identification and Model Predictive Control of a pH Neutralization Process Based on Linear and Wiener Models. *IFAC Proceedings Volumes*, 36(16):1507–1512, 2003. doi: 10.1016/S1474-6670(17)34973-X.

A. Hammerstein. Nichtlineare Integralgleichungen nebst Anwendungen. *Acta Mathematica*, 54(1):117–176, 1930. doi: 10.1007/BF02547519.

I. Hamtil, J. Farlík, and M. Krátký. The Air Defence Missile System Effective Coverage Determination using Computer Simulation. In *International Conference on Military Technologies (ICMT) 2015*, pages 1–5, 2015. doi: 10.1109/MILTECHS.2015.7153719.

Z. Han, B. Cheng, C. Wang, and W. Yang. Identification of CSTR using Extreme Learning Machine based Hammerstein-Wiener Model. In *2017 3rd IEEE International Conference on Control Science and Systems Engineering (ICCSSE)*, pages 733–736, 2017. doi: 10.1109/CCSSE.2017.8088031.

Z. Hasiewicz and G. Mzyk. Hammerstein System Identification by Non-parametric Instrumental Variables. *International Journal of Control*, 82(3):440–455, 2009. doi: 10.1080/00207170802225930.

M. Hong and S. Cheng. Hammerstein-Wiener Model Predictive Control of Continuous Stirred Tank Reactor. In *Electronics and Signal Processing*, pages 235–242. Springer, 2011. doi: 10.1007/978-3-642-21697-8_30.

K. Hornik, M. Stinchcombe, and H. White. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2(5):359–366, 1989. doi: 10.1016/0893-6080(89)90020-8.

X.-L. Hu and H.-F. Chen. Strong Consistence of Recursive Identification for Wiener Systems. *Automatica*, 41(11):1905–1916, 2005. doi: 10.1016/j.automatica.2005.06.006.

Y. Hua and T. K. Sarkar. Matrix Pencil Method for Estimating Parameters of Exponentially Damped/Undamped Sinusoids in Noise. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(5):814–824, 1990. doi: 10.1109/29.56027.

I. W. Hunter and M. J. Korenberg. The Identification of Nonlinear Biological Systems: Wiener and Hammerstein Cascade Models. *Biological Cybernetics*, 55(2-3):135–144, 1986. doi: 10.1007/BF00341929.

A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

J. Jencia, D. Hepsiba, and L. D. Vijay Anand. Identification of Heat Exchanger Process through Hammerstein and Polynomial Models. *Physica Scripta*, 99(7):075215, 2024. doi: 10.1088/1402-4896/ad514e.

J.-N. Juang and R. S. Pappa. An Eigensystem Realization Algorithm for Modal Parameter Identification and Model Reduction. *Journal of Guidance, Control, and Dynamics*, 8(5):620–627, 1985. doi: 10.2514/3.20031.

F. Jurado. Hammerstein-model based Predictive Control of Micro-turbines. *International Journal of Energy Research*, 30(7):511–521, 2006. doi: 10.1002/er.1166.

H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

A. Y. Kibangou and G. Favier. Wiener-Hammerstein Systems Modeling Using Diagonal Volterra Kernels Coefficients. *IEEE Signal Processing Letters*, 13(6):381–384, 2006. doi: 10.1109/LSP.2006.871705.

D. Kreider, R. Kuller, D. Ostberg, and F. Perkins. *An Introduction to Linear Analysis*. Addison-Wesley Publishing Company Inc, 1966.

S. Kung, K. Arun, and D. Bhaskar Rao. State-Space and Singular-Value Decomposition-based Approximation Methods for the Harmonic Retrieval Problem. *Journal of the Optical Society of America*, 73(12):1799–1811, 1983. doi: 10.1364/JOSA.73.001799.

H. Kuttruff. *Room Acoustics*. Taylor & Francis, London, 5th edition, 2009.

J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Society for Industrial and Applied Mathematics, 2016.

S. Lacy and D. Bernstein. Subspace Identification with Guaranteed Stability using Constrained Optimization. *IEEE Transactions on Automatic Control*, 48(7):1259–1263, 2003. doi: 10.1109/TAC.2003.814273.

W. E. Larimore. Canonical Variate Analysis in Identification, Filtering, and Adaptive Control. In *29th IEEE Conference on Decision and Control*, volume 2, pages 596–604, 1990. doi: 10.1109/CDC.1990.203665.

L. Lauwers. *Some Practical Applications of the Best Linear Approximation in Nonlinear Block-Oriented Modelling*. PhD thesis, Vrije Universiteit Brussel, Brussels, Belgium, 2011.

L. Lauwers, J. Schoukens, and R. Pintelon. Initial Estimates for Wiener-Hammerstein Systems Using the Best Linear Approximation. In *I^2MTC, Proceedings of the IEEE International Instrumentation and Measurement Technology Conference*, pages 928–932, 2008. doi: 10.1109/IMTC.2008.4547169.

L. Ljung. *System Identification: Theory for the User*. Prentice Hall PTR, 1999.

L. Ljung, C. Andersson, K. Tiels, and T. B. Schön. Deep Learning and System Identification. *IFAC-PapersOnLine*, 53(2):1175–1181, 2020. doi: 10.1016/j.ifacol.2020.12.1329.

I. R. Manchester, M. Revay, and R. Wang. Contraction-Based Methods for Stable Identification and Robust Machine Learning: A Tutorial. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2955–2962, 2021. doi: 10.1109/CDC45484.2021.9683128.

A. Marconato, J. Sjöberg, and J. Schoukens. Initialization of Nonlinear State-Space Models Applied to the Wiener–Hammerstein Benchmark. *Control Engineering Practice*, 20(11):1126–1132, 2012. doi: 10.1016/j.conengprac.2012.07.004.

V. Z. Marmarelis. Identification of Nonlinear Biological Systems Using Laguerre Expansions of Kernels. *Annals of Biomedical Engineering*, 21:573–589, 1993. doi: 10.1007/BF02368639.

A. Mauroy, I. Mezić, and Y. Susuki. *The Koopman Operator in Systems and Control: Concepts, Methodologies, and Applications*. Springer, 2020.

T. McKelvey, H. Akçay, and L. Ljung. Subspace-based Multivariable System Identification from Frequency Response Data. *IEEE Transactions on Automatic Control*, 41(7):960–979, 1996. doi: 10.1109/9.508900.

L. Meirovitch. *Analytical Methods in Vibrations*. Macmillan, 1967.

E. Memmel, C. Menzen, and K. Batselier. Bayesian Framework for a MIMO Volterra Tensor Network. *IFAC-PapersOnLine*, 56(2):7294–7299, 2023. doi: 10.1016/j.ifacol.2023.10.341.

J. D. Morningred, B. E. Paden, D. E. Seborg, and D. A. Mellichamp. An Adaptive Nonlinear Predictive Controller. *Chemical Engineering Science*, 47(4):755–762, 1992. doi: 10.1016/0009-2509(92)80266-F.

G. Morren, P. Lemmerling, and S. Van Huffel. Decimative Subspace-based Parameter Estimation Techniques Applied to Magnetic Resonance Spectroscopy Signals. In *2001 Conference Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 3, pages 2319–2322, 2001. doi: 10.1109/IEMBS.2001.1017240.

G. Mzyk. Combined Parametric-Nonparametric Identification of Block-Oriented Systems. In *Lecture Notes in Control and Information Sciences*, volume 454. Springer, 2014.

S. N. Naikwad and S. V. Dudul. Identification of a Typical CSTR Using Optimal Focused Time Lagged Recurrent Neural Network Model with Gamma Memory Filter. *Applied Computational Intelligence and Soft Computing*, 2009:385757, 2009. doi: 10.1155/2009/385757.

K. Narendra and P. Gallman. An Iterative Method for the Identification of Nonlinear Systems using a Hammerstein Model. *IEEE Transactions on Automatic Control*, 11 (3):546–550, 1966. doi: 10.1109/TAC.1966.1098387.

S.-I. Niculescu, I. Boussaada, X.-G. Li, G. Mazanti, and C.-F. Méndez-Barrios. Stability, Delays and Multiple Characteristic Roots in Dynamical Systems: A Guided Tour. *IFAC-PapersOnLine*, 54(18):222–239, 2021. doi: 10.1016/j.ifacol.2021.11.143.

A. Nikolakopoulou and R. D. Braatz. Polynomial NARX-based Nonlinear Model Predictive Control of Modular Chemical Systems. *Computers & Chemical Engineering*, 177:108272, 2023. doi: 10.1016/j.compchemeng.2023.108272.

S. J. Norquay, A. Palazoglu, and J. A. Romagnoli. Application of Wiener Model Predictive Control (WMPC) to an Industrial C2-Splitter. *Journal of Process Control*, 9(6):461–473, 1999a. doi: 10.1016/S0959-1524(99)00036-0.

S. J. Norquay, A. Palazoglu, and J. A. Romagnoli. Application of Wiener Model Predictive Control (WMPC) to a pH Neutralization Experiment. *IEEE Transactions on Control Systems Technology*, 7(4):437–445, 1999b. doi: 10.1109/87.772159.

K. Ogata. *Modern Control Engineering*. Instrumentation and Controls Series. Prentice Hall, 2010.

B. A. Ojokoh, O. A. Sarumi, K. V. Salako, A. J. Gabriel, A. E. Taiwo, O. V. Johnson, I. P. Adegun, and O. T. Babalola. Modeling and Predicting the Spread of COVID-19: A Continental Analysis. In *Data Science for COVID-19*, pages 299–317. Elsevier, 2022. doi: 10.1016/B978-0-323-90769-9.00039-6.

A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall Press, 3rd edition, 2009.

I. Oseledets. Approximation of $2^d \times 2^d$ Matrices Using Tensor Decomposition. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2130–2145, 2010. doi: 10.1137/090757861.

I. Oseledets. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33 (5):2295–2317, 2011. doi: 10.1137/090752286.

I. Oseledets and E. Tyrtyshnikov. TT-cross Approximation for Multidimensional Arrays. *Linear Algebra and its Applications*, 432(1):70–88, 2010. doi: 10.1016/j.laa.2009.07.024.

E. G. Ovy and Q. Sun. Wind Turbine Dynamics Modelling by a Bond Graph Approach. *International Journal of Dynamics and Control*, 6:1523–1542, 2018. doi: 10.1007/s40435-017-0390-y.

J. Paduart. *Identification of Nonlinear Systems Using Polynomial Nonlinear State Space Models*. PhD thesis, Vrije Universiteit Brussel, Brussels, Belgium, 2008.

J. Paduart, J. Schoukens, and L. Gommé. On the Equivalence between some Block Oriented Nonlinear Models and Nonlinear Polynomial State Space Models. *IEEE Instrumentation & Measurement Technology Conference*, pages 1–6, 2007. doi: 10.1109/IMTC.2007.379029.

J. Paduart, L. Lauwers, J. Swevers, K. Smolders, J. Schoukens, and R. Pintelon. Identification of Nonlinear Systems Using Polynomial Nonlinear State Space Models. *Automatica*, 46(4):647–656, 2010. doi: 10.1016/j.automatica.2010.01.001.

G. Pajunen. Adaptive Control of Wiener Type Nonlinear Systems. *Automatica*, 28(4): 781–785, 1992. doi: 10.1016/0005-1098(92)90037-G.

I. Papp. Missile Mathematical Model and System Design. *AARMS – Academic and Applied Research in Military and Public Management Science*, 16(1):29–35, 2017. doi: 10.32565/aarms.2017.1.3.

J.-M. Papy, L. De Lathauwer, and S. Van Huffel. Exponential Data Fitting using Multilinear Algebra: The Decimative Case. *Journal of Chemometrics*, 23(7–8): 341–351, 2009. doi: 10.1002/cem.1212.

M. Pawlak. On the Series Expansion Approach to the Identification of Hammerstein Systems. *IEEE Transactions on Automatic Control*, 36(6):763–767, 1991. doi: 10.1109/9.86954.

V. Peterka. Bayesian System Identification. *Automatica*, 17(1):41–53, 1981. doi: 10.1016/0005-1098(81)90083-2.

G. Pillonetto, T. Chen, A. Chiuso, G. De Nicolao, and L. Ljung. *Regularization for Nonlinear System Identification*, pages 313–342. Springer International Publishing, 2022. doi: 10.1007/978-3-030-95860-2_8.

R. Pintelon and J. Schoukens. *System Identification: A Frequency Domain Approach*. Wiley / IEEE Press, 2012.

V. F. Pisarenko. The Retrieval of Harmonics from a Covariance Function. *Geophysical Journal International*, 33(3):347–366, 1973. doi: 10.1111/j.1365-246X.1973.tb03424.x.

A. M. Ponsiglione, F. Montefusco, L. Donisi, A. Tedesco, C. Cosentino, A. Merola, M. Romano, and F. Amato. A General Approach for the Modelling of Negative Feedback Physiological Control Systems. *Bioengineering*, 10(7), 2023. doi: 10.3390/bioengineering10070835.

G. R. D. Prony. Essai expérimental et analytique: sur les lois de la dilatabilité de fluides élastiques et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alcool à différentes températures. *Journal de l'école polytechnique*, 1(22):24–76, 1795.

L. Qi. Eigenvalues of a Real Supersymmetric Tensor. *Journal of Symbolic Computation*, 40(6):1302–1324, 2005. doi: 10.1016/j.jsc.2005.05.007.

S. J. Qin. An Overview of Subspace Identification. *Computers & Chemical Engineering*, 30(10):1502–1513, 2006. doi: 10.1016/j.compchemeng.2006.05.045.

M. Reed and B. Simon. *Methods of Modern Mathematical Physics: Functional Analysis*. Functional Analysis. Academic Press, 1972.

R. Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag, 1996.

B. Roland, D. Bertrand, and R. Gaël. High-resolution Spectral Analysis of Mixtures of Complex Exponentials Modulated by Polynomials. *IEEE Transactions on Signal Processing*, 54(4), Apr. 2006. doi: 10.1109/TSP.2006.870556.

A. Romanenko, L. O. Santos, and P. A. Afonso. Unscented Kalman Filtering of a Simulated pH System. *Industrial & Engineering Chemistry Research*, 43(23):7531–7538, 2004. doi: 10.1021/ie049655v.

M. Ruderman and T. Bertram. Structural Hysteresis Model of Transmitting Mechanical Systems. In *Journal of Physics: Conference Series*, volume 585, 2015. doi: 10.1088/1742-6596/585/1/012009.

W. J. Rugh. *Nonlinear System Theory: The Volterra/Wiener Approach*. Johns Hopkins University Press, 1981.

F. Scarselli and A. Chung Tsoi. Universal Approximation Using Feedforward Neural Networks: A Survey of Some Existing Methods, and Some New Results. *Neural Networks*, 11(1):15–37, 1998. doi: 10.1016/S0893-6080(97)00097-X.

M. Schetzen. *The Volterra and Wiener Theories of Nonlinear Systems*. Wiley, 1980.

J. Schoukens, R. Pintelon, T. Dobrowiecki, and Y. Rolain. Identification of Linear Systems with Nonlinear Distortions. *Automatica*, 41:491–504, 2005. doi: 10.1016/j.automatica.2004.10.004.

J. Schoukens, Y. Rolain, and R. Pintelon. Analysis of Windowing/Leakage Effects in Frequency Response Function Measurements. *Automatica*, 42:27–38, 2006. doi: 10.1016/j.automatica.2005.08.004.

J. Schoukens, W. D. Widanage, K. R. Godfrey, and R. Pintelon. Initial Estimates for the Dynamics of a Hammerstein System. *Automatica*, 43(7):1296–1301, 2007. doi: 10.1016/j.automatica.2006.12.003.

J. Schoukens, J. Lataire, R. Pintelon, G. Vandersteen, and T. Dobrowiecki. Robustness Issues of the Best Linear Approximation of a Nonlinear System. *IEEE Transactions on Instrumentation and Measurement*, 58(5):1737–1745, 2009a. doi: 10.1109/TIM.2009.2012948.

J. Schoukens, J. Suykens, and L. Ljung. Wiener-Hammerstein Benchmark. In *15th IFAC Symposium on System Identification (SYSID 2009)*, 2009b.

J. Schoukens, R. Pintelon, and Y. Rolain. *Mastering System Identification in 100 Exercises*. Wiley / IEEE Press, 2012.

J. Schoukens, K. Tiels, and M. Schoukens. Generating Initial Estimates for Wiener-Hammerstein Systems Using Phase Coupled Multisines. *IFAC Proceedings Volumes*, 47(3):499–504, 2014a. doi: 10.3182/20140824-6-ZA-1003.00037.

J. Schoukens, R. Pintelon, Y. Rolain, M. Schoukens, K. Tiels, L. Vanbeylen, A. Van Mulders, and G. Vandersteen. Structure Discrimination in Block-oriented Models using Linear Approximations: A Theoretic Framework. *Automatica*, 53:225–234, 2015. doi: 10.1016/j.automatica.2014.12.045.

M. Schoukens and K. Tiels. Identification of Block-oriented Nonlinear Systems starting from Linear Approximations: A Survey. *Automatica*, 85:270–292, 2017. doi: 10.1016/j.automatica.2017.06.044.

M. Schoukens, R. Pintelon, and Y. Rolain. Identification of Wiener–Hammerstein Systems by a Nonparametric Separation of the Best Linear Approximation. *Automatica*, 50(2):628–634, 2014b. doi: 10.1016/j.automatica.2013.12.027.

T. B. Schön, A. Wills, and B. Ninness. System Identification of Nonlinear State-Space Models. *Automatica*, 47(1):39–49, 2011. doi: 10.1016/j.automatica.2010.10.013.

M. Schüssler and O. Nelles. Extrapolation Behavior Comparison of Nonlinear State Space Models. *IFAC-PapersOnLine*, 54(7):487–492, 2021. doi: 10.1016/j.ifacol.2021.08.407.

A. P. Seyranian, E. Lund, and N. Olhoff. Multiple Eigenvalues in Structural Optimization Problems. *Structural Optimization*, 8:207–227, 1994. doi: 10.1007/BF01742705.

K. Shea. *Measuring the Impact of Monitoring: How We Know Transparent Near-Real-Time Data Can Help Save the Forests*, pages 263–273. Springer International Publishing, 2022. doi: 10.1007/978-3-030-78853-7_18.

K. Shi, X. Ma, and G. T. Zhou. Acoustic Echo Cancellation Using a Pseudocoherence Function in the Presence of Memoryless Nonlinearity. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 55(9):2639–2649, 2008. doi: 10.1109/tcsi.2008.920114.

G. R. Shinde, A. B. Kalamkar, P. N. Mahalle, N. Dey, J. Chaki, and A. E. Hassanien. Forecasting Models for Coronavirus Disease (COVID-19): A Survey of the State-of-the-Art. *SN Computer Science*, 1(4):197, 2020. doi: 10.1007/s42979-020-00209-9.

D. Sidorov, A. Tynda, V. Muratov, and E. Yanitsky. Volterra Black-Box Models Identification Methods: Direct Collocation vs. Least Squares. *Mathematics*, 12(2):227, 2024. doi: 10.3390/math12020227.

P. Sircar. *Mathematical Aspects of Signal Processing*. Cambridge University Press, 2016.

J. Sjöberg and J. Schoukens. Initializing Wiener–Hammerstein Models based on Partitioning of the Best Linear Approximation. *Automatica*, 48(2):353–359, 2012. doi: 10.1016/j.automatica.2011.07.007.

J. Sjöberg, L. Lauwers, and J. Schoukens. Identification of Wiener–Hammerstein models: Two algorithms based on the Best Split of a Linear Model applied to the SYSID'09 Benchmark Problem. *Control Engineering Practice*, 20(11):1119–1125, 2012. doi: 10.1016/j.conengprac.2012.07.001.

J. Smith. *Introduction to Digital Filters: With Audio Applications*. Music Signal Processing Series. W3K, 2007.

T. Söderström and P. Stoica. *System Identification*. Prentice-Hall Software Series. Prentice Hall, 1989.

M. Soheili, M. Shafieirad, T. Morovati, and E. K. Ronaghi. Identification of Nonlinear Systems in the Frequency Domain using Two-dimensional Volterra Series. In *2024 10th International Conference on Control, Instrumentation and Automation (ICCIA)*, pages 1–5, 2024. doi: 10.1109/ICCIA65044.2024.10768151.

P. Stoica and K. C. Sharmen. Maximum Likelihood Methods for Direction-of-arrival Estimation. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38(7): 1132–1143, 1990. doi: 10.1109/29.57542.

J. Suykens, B. De Moor, and J. Vandewalle. Nonlinear System Identification using Neural State Space Models, applicable to Robust Control Design. *International Journal of Control*, 62(1), 1995. doi: 10.1080/00207179508921536.

A. H. Tan, H. K. Wong, and K. Godfrey. Identification of a Wiener–Hammerstein System using an Incremental Nonlinear Optimisation Technique. *Control Engineering Practice*, 20(11):1140–1148, 2012. doi: 10.1016/j.conengprac.2012.04.007.

K. Tiels, M. Schoukens, and J. Schoukens. Generation of Initial Estimates for Wiener-Hammerstein Models via Basis Function Expansions. *IFAC Proceedings Volumes*, 47(3):481–486, 2014. doi: 10.3182/20140824-6-ZA-1003.02292.

V. Y. Tikhonov and A. R. Abdipour. Comparison of Nonlinear Distortion Compensation Methods in Power Amplifiers. *Synchroinfo Journal*, 9(5):2–8, 2023. doi: 10.36724/2664-066X-2023-9-5-2-8.

J. Umenberger and I. R. Manchester. Specialized Algorithm for Identification of Stable Linear Systems using Lagrangian Relaxation. In *2016 American Control Conference (ACC)*, pages 930–935, 2016. doi: 10.1109/ACC.2016.7525034.

S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*, volume 9 of *Classics in Applied Mathematics*. SIAM, 2004. doi: 10.1137/1.9780898719697.

P. Van Overschee and B. De Moor. N4SID: Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems. *Automatica*, 30(1):75–93, 1994. doi: 10.1016/0005-1098(94)90230-5.

L. Vanbeylen. A Fractional Approach to Identify Wiener–Hammerstein Systems. *Automatica*, 50(3):903–909, 2014. doi: 10.1016/j.automatica.2013.12.013.

G. Vandersteen and J. Schoukens. Measurement and Identification of Nonlinear Systems Consisting of Linear Dynamic Blocks and One Static Nonlinearity. *IEEE Transactions on Automatic Control*, 44(6):1266–1271, 1999. doi: 10.1109/9.769388.

V. Verdult. *Nonlinear System Identification: A State-Space Approach*. Twente University Press (TUP), 2002.

M. Verhaegen and P. Dewilde. Subspace Model Identification Part 1. The Output-Error State-Space Model Identification Class of Algorithms. *International Journal of Control*, 56(5):1187–1210, 1992a. doi: 10.1080/00207179208934363.

M. Verhaegen and P. Dewilde. Subspace Model Identification Part 2. Analysis of the Elementary Output-Error State-Space Model Identification Algorithm. *International Journal of Control*, 56(5):1211–1241, 1992b. doi: 10.1080/00207179208934364.

M. Verhaegen and V. Verdult. *Filtering and System Identification: A Least Squares Approach*. Cambridge University Press, 2007.

P. Vijayan, P. Dreesen, I. Markovsky, and M. Ishteva. Parameter Estimation of Multiple Poles by Subspace-Based Method. In *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 1948–1953, 2023. doi: 10.1109/CoDIT58514.2023.10284047.

P. Vijayan, P. Dreesen, J. Lataire, and M. Ishteva. Enhancing Block-Oriented System Identification Using Prior-Informed Volterra Kernels. *IEEE Transactions on Instrumentation and Measurement*, 74:1–11, 2025a. doi: 10.1109/TIM.2025.3575968.

P. Vijayan, P. Dreesen, J. Lataire, and M. Ishteva. Block-Oriented Systems, 2025b. IEEE DataPort, Accessed 2025, doi: 10.21227/v9j3-0z06.

J. Vörös. Parameter Identification of Discontinuous Hammerstein Systems. *Automatica*, 33(6):1141–1146, 1997. doi: 10.1016/S0005-1098(97)00009-5.

J. Vörös. Parameter Identification of Wiener Systems with Discontinuous Nonlinearities. *Systems & Control Letters*, 44(5):363–372, 2001. doi: 10.1016/S0167-6911(01)00155-4.

W. Wang and R. Henriksen. Generalized Predictive Control of Nonlinear Systems of the Hammerstein Form. *Modeling, Identification and Control*, 15(4):253–262, 1994. doi: 10.4173/mic.1994.4.4.

B. Warsito, R. Santoso, Suparti, and H. Yasin. Cascade Forward Neural Network for Time Series Prediction. *Journal of Physics: Conference Series*, 1025(1), 2018. doi: 10.1088/1742-6596/1025/1/012097.

E. W. Weisstein. Multinomial series, 2005. Accessed: 2025, from MathWorld–A Wolfram Web Resource, `mathworld.wolfram.com/MultinomialSeries.html`.

D. T. Westwick and R. E. Kearney. *Identification of Nonlinear Physiological Systems*. Wiley-IEEE Press, 2003.

D. T. Westwick and J. Schoukens. Initial Estimates of the Linear Subsystems of Wiener–Hammerstein Models. *Automatica*, 48(11):2931–2936, 2012. doi: 10.1016/j.automatica.2012.06.091.

N. Wiener. Response of a Nonlinear Device to Noise. Technical Report 129, MIT Radiation Lab, 1942.

N. Wiener. *Nonlinear Problems in Random Theory*. M.I.T. paperback series. MIT Press, 1958.

Wiener-Hammerstein Benchmark Results. Nonlinear Benchmark. Online, 2025. Accessed: 2025, `www.nonlinearbenchmark.org/results-reporting`.

Y. Wu, Z. Wang, Y. Li, W. Chen, R. Du, and Q. Chen. Characteristic Modeling and Control of Servo Systems with Backlash and Friction. *Mathematical Problems in Engineering*, 2014(1), 2014. doi: 10.1155/2014/328450.

Z. Xiao, X. Jing, and L. Cheng. Estimation of Parametric Convergence Bounds for Volterra Series Expansion of Nonlinear Systems. *Mechanical Systems and Signal Processing*, 45(1):28–48, 2014. doi: 10.1016/j.ymssp.2013.11.012.

Y. Xu, B. Mu, and T. Chen. On Kernel Design for Regularized Volterra Series Identification of Wiener–Hammerstein Systems. *Automatica*, 179:112457, 2025. doi: 10.1016/j.automatica.2025.112457.

Z. Zeng. Computing Multiple Roots of Inexact Polynomials. *Mathematics of Computation*, 74:869–903, 2005. doi: 10.1090/S0025-5718-04-01692-8.

D. Zhang, H. Wang, X. Wang, and Z. Lü. Accuracy Assessment of the Global Forest Watch Tree Cover 2000 in China. *International Journal of Applied Earth Observation and Geoinformation*, 87:102033, 2020. doi: 10.1016/j.jag.2019.102033.

E. Zhang, J. Antoni, R. Pintelon, and J. Schoukens. Fast Detection of System Nonlinearity using Nonstationary Signals. *Mechanical Systems and Signal Processing*, 24(7): 2065–2075, 2010. doi: 10.1016/j.ymssp.2010.05.011.

Y. Zhu. Distillation Column Identification for Control using Wiener Model. In *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*, volume 5, pages 3462–3466, 1999. doi: 10.1109/ACC.1999.782408.

Y.-P. Zhu and Z. Lang. A New Convergence Analysis for the Volterra Series Representation of Nonlinear Systems. *Automatica*, 111, 2020. doi: 10.1016/j.automatica.2019.108599.

D. Zill and M. Cullen. *Differential Equations with Boundary-Value Problems*. Cengage Learning, 2008.

Z. Zou, M. Yu, Z. Wang, X. Liu, Y. Guo, F. Zhang, and N. Guo. Nonlinear Model Algorithmic Control of a pH Neutralization Process. *Chinese Journal of Chemical Engineering*, 21(4):395–400, 2013. doi: 10.1016/S1004-9541(13)60479-6.

Z. Y. Zou, G. P. Liu, and N. Guo. Predictive Control of Nonlinear Hammerstein Systems and Application to pH Processes. In *European Control Conference (ECC)*, pages 3657–3662, 2003. doi: 10.23919/ECC.2003.7086509.

Statement on the use of Generative AI

I did not use generative AI assistance tools during the research or writing process of my thesis, except for language correction. Specifically, Grammarly and ChatGPT were used for language assistance. The Dutch translation of the abstract and the popularised abstract was done with the help of ChatGPT, and my Promoters verified the translation. The text, code, and images in this thesis are my own (unless otherwise specified). Generative AI has only been used in accordance with the KU Leuven guidelines and appropriate references have been added. I have reviewed and edited the content as needed and I take full responsibility for the content of the thesis.

Curriculum Vitae

Prabhu Vijayan completed his Master's in Computational Engineering from Friedrich-Alexander Universität Erlangen-Nürnberg (FAU), Germany, in 2020. His master's thesis, titled "Implementation and validation of fatigue assessment algorithm based on advanced short crack fracture mechanics," was conducted in collaboration with IBGSL Lifetime Management, Framatome GmbH, and the Chair of Applied Mechanics at FAU Erlangen. In 2021, he began his PhD as part of the research project Structured Low-Rank Matrix/Tensor Approximation (SeLMA) – Numerical Optimisation-Based Algorithms and Applications, under the supervision of Prof. dr. Mariya Ishteva in the ADVISE-NUMA research group at KU Leuven, Campus Geel and Prof. dr. ir. John Lataire at the Department of Fundamental Electricity and Instrumentation (ELEC), Vrije Universiteit Brussel, Brussels, Belgium.

Prabhu's research focuses on system identification and nonlinear modelling, particularly using Volterra methods, tensor representations, and block-oriented structures. His work bridges theory and practice, with an emphasis on data-driven modelling using structured priors. The work presented in this thesis is a detailed account of the research carried out during his doctoral studies.

Publication List

Peer-reviewed Journal Article

Prabhu Vijayan, Philippe Dreesen, John Lataire, and Mariya Ishteva, "Enhancing Block-Oriented System Identification Using Prior-Informed Volterra Kernels," in *IEEE Transactions on Instrumentation and Measurement*, vol. 74, pp. 1–11, 2025, Article No. 6507311.
DOI: 10.1109/TIM.2025.3575968

Peer-reviewed Conference Proceedings

Prabhu Vijayan, Philippe Dreesen, Ivan Markovsky, and Mariya Ishteva, "Parameter Estimation of Multiple Poles by Subspace-Based Method," in *Proceedings of the 2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2023, pp. 1948–1953.
DOI: 10.1109/CoDIT58514.2023.10284047

Workshops

- **Prabhu Vijayan**, Philippe Dreesen, John Lataire, and Mariya Ishteva, "PNLSS modelling and estimation using prior-incorporated Volterra kernel: A Wiener-Hammerstein case study," Workshop on Nonlinear System Identification Benchmarks, Lugano, Apr. 2024.
- **Prabhu Vijayan**, Philippe Dreesen, John Lataire, and Mariya Ishteva, "Nonlinear state space modelling and estimation using prior-incorporated Volterra kernel for some block-oriented systems," 43rd Benelux Meeting on Systems and Control, Blankenberge, Mar. 2024.
- **Prabhu Vijayan**, Philippe Dreesen, John Lataire, and Mariya Ishteva, "Volterra kernel-based Nonlinear State Space Modelling and Estimation for some SISO Block-oriented systems," NUMA study day 2024, Leuven, Jan. 2024.
- **Prabhu Vijayan**, Philippe Dreesen, John Lataire, and Mariya Ishteva, "Demonstrating equivalence between PNLSS and Volterra models for some

SISO block-oriented models," ERNSI Workshop 2023, Stockholm, Sep. 2023.

- **Prabhu Vijayan**, Philippe Dreesen, Ivan Markovsky, and Mariya Ishteva, "Parameter Estimation of Multiple Poles by Subspace-based Methods," NUMA study day 2023, Gent, Jan. 2023.
- **Prabhu Vijayan**, Philippe Dreesen, Ivan Markovsky, and Mariya Ishteva, "Parameter Estimation of Multiple Poles by Subspace-based Methods," ERNSI Workshop 2022, Leuven, Sep. 2022.
- **Prabhu Vijayan**, Philippe Dreesen, Ivan Markovsky, and Mariya Ishteva, "Parameter estimation of multiple poles by subspace-based methods," Workshop on Low-Rank Models and Applications, LRMA 22, Mons, Sep. 2022.
- **Prabhu Vijayan**, Philippe Dreesen, Ivan Markovsky, and Mariya Ishteva, "Parameter estimation of multiple poles by subspace-based methods," 41st Benelux Meeting on Systems and Control, Brussels, Jul. 2022.

FACULTY OF ENGINEERING
DEPARTMENT ELECTRICITY (ELEC)
Pleinlaan 2
1050 Brussel
elec.paddlecms.net

FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE
Kleinhoefstraat 4
B-2440 Geel
www.numa.cs.kuleuven.be, iiw.kuleuven.be/onderzoek/advise