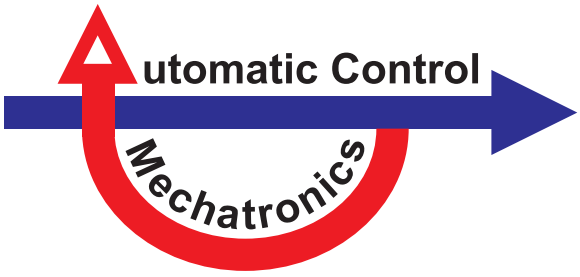

Tobias Münker

Machine Learning with Finite Impulse Response Models

Dissertation

Schriftenreihe der Arbeitsgruppe
Mess- und Regelungstechnik – Mechatronik
Department Maschinenbau

Herausgeber Oliver Nelles

UNIVERSITÄT SIEGEN

Band 5

Impressum
Prof. Dr.-Ing. Oliver Nelles
Arbeitsgruppe Mess- und Regelungstechnik
Department Maschinenbau
Universität Siegen
57068 Siegen
ISSN 2193-0538
URN urn:nbn:de:hbz:467-21182
Zugl.: Siegen, Univ., Diss., 2021

Machine Learning with Finite Impulse Response Models

DISSERTATION
zur Erlangung des Grades eines Doktors
der Ingenieurwissenschaften

vorgelegt von
Tobias Münker, M.Sc.
aus Olpe

genehmigt durch die Naturwissenschaftlich-Technische Fakultät
der Universität Siegen
Siegen 2021

Gedruckt auf alterungsbeständigem holz- und säurefreiem Papier.

Betreuer und erster Gutachter
Prof. Dr.-Ing. Oliver Nelles
Universität Siegen

Zweiter Gutachter
Prof. Dr.-Ing. Peter Kraemer
Universität Siegen

Tag der mündlichen Prüfung
09. März 2021

Acknowledgments

This dissertation was written during my time at the Institute of Mechanics and Control Engineering - Mechatronics at the University of Siegen. On this occasion, I would like to thank everyone who contributed to the success of this work.

First of all, I want to express my sincere gratitude to Prof. Dr.-Ing. Oliver Nelles for his continuous support and supervision of the thesis. Both the freedom to do independent research and his outstanding ability to make mathematically complex results accessible were essential for this research work.

I further want to thank Prof. Dr.-Ing. Peter Kraemer for his interest in my work and the supervision of the thesis as second examiner.

I highly appreciated the cooperation with Technip FMC, SMS Group, and Robert Bosch GmbH. The exchange and the possibility to use system identification on industry examples was very exciting.

Thanks to all colleagues and friends for creating a great working culture, for the many fruitful discussions, and for the good times outside working hours. In particular, I thank Julian Belz, Tim Decker, Tim Oliver Heinz, Henning Jung, Geritt Kampmann, Diana Klein, Timm Julian Peter, and Max Schüssler.

I want to express my thank to my parents Antje and Wolfram, for their love and for showing me how to become curious about the world, and my siblings Ines and Sven for sharing this curiosity with me. Finally, I want to thank my wife Nina-Carolin and my son Noah for their love and unconditional support in all my decisions.

Olpe, March 9th 2021
Tobias Münker

Abstract

An accurate and reliable model is the foundation for analysis, design, and control of a modern automation system. It is often too complicated, too expensive, or too inaccurate to develop these models based on first principles. In recent years, machine learning has made tremendous progress through the usage of data to generate models. Nevertheless, a key drawback is that for the identified models there is no stability guarantee. Thus, within this thesis, methods for identifying both linear and nonlinear systems based on finite impulse response (FIR) models are investigated. These avoid feedback and thus ensure stability.

Linear FIR models offer a compelling advantage due to the interpretability of the impulse response and inherent stability. Recently, novel methods for regularization based on a specifically designed Tikhonov regularization have been proposed. In this contribution these approaches are extended to allow for a better incorporation of existing prior knowledge. The developed method can be employed for order selection and gray-box identification. Its feasibility is demonstrated on numerical benchmarks and on a laboratory example.

Local model networks allow for the extension of linear identification methods to nonlinear systems. Here, the regularization mechanism applied to linear systems is employed to identify local FIR models. Especially for identification problems with a low signal to noise ratio, the method performs significantly better than local model networks with feedback of the output.

Finally, machine learning with convolutional neural networks is investigated. The relation between these and block-oriented nonlinear systems is analyzed. Then, a specific structure of a deep neural network containing FIR models as building blocks is proposed. This structure is equipped with a regularization scheme adopted from linear FIR models for the impulse responses comprised in the neural network. It is shown that the bias-variance trade-off is influenced positively on a benchmark example and achieves state-of-the-art results while additionally the internal impulse responses are smoothed and the stability of the system is guaranteed.

Zusammenfassung

Ein genaues und zuverlässiges Modell ist die Grundlage für Analyse, Auslegung und Regelung von modernen Automatisierungssystemen. Es ist häufig zu kompliziert, zu teuer oder zu ungenau, diese Modelle basierend auf physikalischen Gleichungen zu entwickeln. Innerhalb der letzten Jahre hat maschinelles Lernen deutliche Fortschritte bei der Verwendung von Daten zur Erstellung dieser Modelle gemacht. Ein wesentlicher Nachteil dieser Methoden ist, dass Stabilität der identifizierten Modelle nicht garantiert ist. Daher werden in dieser Arbeit Methoden zur Identifikation basierend auf FIR Modellen untersucht. Diese enthalten keine Rückkopplung und garantieren deshalb strukturbedingt Stabilität.

Lineare FIR Systeme bieten signifikante Vorteile aufgrund der einfachen Interpretierbarkeit ihrer Impulsantwort sowie ihrer inherenten Stabilität. In der jüngeren Literatur wurden neue Verfahren zur Vermeidung hoher Parametervarianz durch Wahl einer speziellen Tikhonov-Regularisierung vorgestellt. In dieser Arbeit werden diese Ansätze erweitert, um ein gezieltes Einbringen von Vorwissen zu ermöglichen. Die entwickelte Methode ist sowohl für Gray-Box-Identifikationsverfahren als auch für Ordnungsselektionsverfahren nutzbar. Die Anwendbarkeit wird anhand eines Laborsystems demonstriert.

Lokale Modellnetze ermöglichen die Erweiterung linearer Identifikationsverfahren auf nichtlineare Systeme. Hierbei wird der Regularisierungsmechanismus, der im Rahmen der linearen Verfahren angewendet wird, zur Identifikation der lokalen FIR Modelle genutzt. Besonders für Identifikationsprobleme mit schlechtem Signal-Rausch-Verhältnis zeigt die Methode deutlich bessere Ergebnisse als lokale Modelle mit Rückkopplung des Ausgangs.

Schließlich wird das maschinelle Lernen mit convolutional neural networks untersucht. Das Verhältnis zwischen diesen und blockorientierten nichtlinearen Systemen wird analysiert. Es wird eine spezifische Struktur von tiefen neuronalen Netzen basierend auf FIR Modellen als Netzbestandteilen vorgeschlagen. Bei der Identifikation des neuronalen Netzes wird der gleiche Regularisierungsansatz wie zur Identifikation linearer FIR Modelle verwendet. Es wird gezeigt, dass der Bias-Varianz-Tradeoff dadurch vorteilhaft beeinflust wird und der Ansatz auf einem aktuellen Benchmarkproblem mit den Ergebnissen von Methoden, die dem aktuellen Stand der Forschung entsprechen, verglichen. Hierbei werden vergleichbare Ergebnisse erzielt und zusätzlich glattere interne Impulsantworten und die Stabilität des Systems garantiert.

Contents

Nomenclature

a_i	denominator coefficient of a transfer function
a_i^{RBF}	linear coefficients of an RBF network
$\underline{b}^{\mathrm{NN}}$	linear parameters for the output of a NN
b_i	numerator coefficient of a transfer function
$\underline{b}_i$	FIR coefficients of the local model of the RFIR LMN
$b_{jl}^{(i)}(k)$	k-th filter coefficient from j-th input to l-th output at unit i
$\underline{b}_{\mathrm{ini}}$	initial weights of a NN layer
$b_{jl}^{(i)}(k)$	k-th filter coefficient for the filter from the j-th input to the l-th hidden output for the i-th unit
$\underline{b}(\underline{\theta}_p)$	input vector of a gray box state space system
$\underline{c}^{\mathrm{NN}}$	offset parameters for the output of a NN
c_i	offset of the i-th local model
c_i^{RBF}	nonlinear coefficients of an RBF network
c_l	class label
$c_l^{(i)}(k)$	value of the l-th convolutional neuron of unit i at time step k
c_s	spring constant
$\underline{c}(\underline{\theta}_p)$	output vector of a gray box state space system
d	damping constant
d_i	coefficients the Hilbert space representation of a function f
df_i	effective number of parameters for the i-th local model
$\underline{d}^{(i,j)}$	offset parameters of j-th dense layer in unit i
$d(\underline{\theta}_p)$	direct coupling between input and output of a gray box state space system
d_i	coefficients the Hilbert space representation of a function g
$\underline{e}_b$	error on mini-batch

$f(\cdot)$ some usually unknown function

$g(k)$ impulse response

$g_i^{(\mathrm{OBF})}(k)$ orthonormal basis functions for a penalty matrix

$\underline{h}(k)$ signal of a hidden layer of a NN

$h_l^{(i)}(k)$ hidden output neuron l in unit i at time step k

k discrete time step

$k(\cdot,\cdot)$ kernel function

l_G lengthscale of the squared exponential kernel

$\hat{m}(\underline{u}^*)$ posterior mean of a GP

m input space dimension

m_c mass of the car

n dynamic model order

$n(k)$ noise signal

n_{VC} VC dimension

n_e number of the epoch

n_n number of neurons of the deep RFIR NN

n_u number of units of the deep RFIR NN

n_x number of features

o order of the system for the construction of the IRP matrix

$p(\cdot)$ probability density function (pdf)

p_l linear parameters for the output layer of the deep RFIR NN

q offset parameter of the dense output layer of the deep RFIR NN

r_P rank of $\underline{P}$

r_R rank of $\underline{R}$

$\underline{u}$ input of a regression problem with dimension m

$u(k)$ input signal

$u_\delta(k)$ injected fuel mass

$\underline{u}^*$ test point for the computation of the posterior of a GP

$u_{\mathrm{VGT}}(k)$ position of the variable geometry turbocharger

$\underline{w}$ parameters of the NN which are not filter coefficients of convolutional layers

$\underline{x}(k)$	features at time step k
$\underline{x}_s(k)$	internal state of a state space system
$\underline{y}_{k_a:k_b}$	vector of signal values from k_a to k_b
$y(k)$	output signal
$y_u(k)$	undisturbed output
$\underline{y}_b$	batched output vector
$\underline{\tilde{y}}_i$	weighted offset-corrected output of i-th local model
$\underline{y}_s$	sampled values a GP output based on the prior distribution
$\underline{y}_t$	sampled training data
$\underline{y}_v$	sampled validation data
$\underline{z}$	data point
$z(i)$	i-th sample of input and output data
$A(q)$	denominator of a transfer function
$\underline{A}(\underline{\theta}_p)$	state-update matrix of a gray box state space system
$\underline{A}^{(i,j)}$	parameters of j-th dense layer in unit i
$B(q)$	numerator of a transfer function
$\underline{D}^{\mathrm{NN}}$	parameters for the inputs of the second layer of a NN
$\underline{E}^{\mathrm{NN}}$	parameters for the inputs of the input layer of a NN
$\underline{F}$	filter matrix for the FIR regularization
$\underline{F}_n$	IRP matrix for the FIR order n
$G(q)$	deterministic path of the general linear model
$H(q)$	noise path of the general linear model
$\underline{I}_F$	Fisher information matrix
J	loss function
J_{AIC}	loss term of the AIC
J_G	generalized cross-validation error
J_L	leave-one out cross-validation error
J_b	loss function for a mini-batch
J_p	moment of inertia of the pendulum
$\underline{K}$	kernel matrix

$\mathrm{KL}(p_1(\underline{z}), p_2(\underline{z}))$ KL divergence from $p_1(\underline{z})$ to $p_2(\underline{z})$

L Cholesky decomposition of the kernel matrix

$L_i(\underline{x})$ local models of an LMN

M number of neurons

N number of samples

$N(z)$ z-transformed noise

N_b number of samples within one mini-batch

$\underline{\mathrm{P}}$ unscaled version of the covariance matrix of FIR coefficients

$\underline{P_\theta}$ prior covariance matrix of the parameters

$\underline{Q}_X$ Q-part of QR factorization of the matrix $\underline{X}$

$R(\hat{y}(\underline{u}, \underline{\theta}))$ risk of the regression function $\hat{y}(\underline{u}, \underline{\theta})$.

$\underline{R}$ penalty matrix

$\underline{R}_X$ R-part of QR factorization of the matrix $\underline{X}$

$R_S(\underline{\theta})$ empirical risk of $\underline{\theta}$

$\underline{R}_i$ penalty matrix for i-th local model

$\underline{S}$ smoothing matrix

$\underline{S}_X$ diagonal matrix of singular values of a matrix $\underline{X}$

$T_e(k)$ engine torque

T_s sampling time

$U(z)$ z-transformed input

$\underline{U}_{P1}$ rows of the unitary matrix of the SVD of $\underline{P}$ which correspond to non-zero singular values

$\underline{U}_{P2}$ rows of the unitary matrix of the SVD of $\underline{P}$ which correspond to zero singular values

$\underline{U}_{R1}$ columns of the unitary matrix of the SVD of the penalty matrix corresponding to non-zero singular values

$\underline{U}_{R2}$ columns of the unitary matrix of the SVD of the penalty matrix corresponding to zero singular values

$\underline{U}_X$ unitary matrix of singular value decomposition of a symmetric matrix $\underline{X}$

$\underline{W}$ weighting matrix for the IRP matrices

$\underline{W}_i$ weighting matrix for i-th local model

$\underline{X}$	regressor
$\underline{X}_b$	regressor for the calculation of an FIR mini-batch
$\underline{\tilde{X}}_i$	weighted regressor of i-th local model
$Y(z)$	z-transformed output
$\underline{Z}$	matrix of input and output samples
α	decay factor for kernels and penalty matrices
$\alpha(k)$	signal of the pendulum angle
β	regularization strength of the squared exponential kernel
δ	certainty for a generalization bound
χ	penalty term for VC dimension based generalization bounds
$\chi(n, N)$	penalty term for an information criterion
η	learning rate
γ	concatenated hyperparameters for a regularized identification problem
κ	decay factor of the learning rate
λ	scaling factor of the penalty matrix
μ	mean of a one dimensional pdf
$\mu_i(\underline{z})$	membership functions of an LMN
$\underline{\mu}$	mean vector of a multivariate pdf
$\phi(\underline{u})$	feature mapping which maps each point in the input space to a function
ρ	scaling factor for the prior covariance of the parameters
σ^2	variance of the output noise
$\underline{\theta}_0$	true parameters
$\underline{\theta}$	model parameters
$\underline{\theta}_B$	Bayesian part of the transformed parameters
$\underline{\theta}_F$	Frequentist part of the transformed parameters
$\underline{\theta}_i^L$	part of the parameters on which the i-th local model depends linearly
$\hat{\underline{\theta}}_N$	ML estimate with N data points
$\underline{\theta}_T$	transformed FIR parameters
$\underline{\theta}_p$	physical parameters for a gray box model
$\underline{\theta}^*$	parameter which minimizes the KL divergence to the true pdf

ζ blending factor between transfer functions

$\hat{\sigma}_p(\underline{u}^*)$ posterior covariance of a GP

Δ_{ij} width of the i-th local model in the j-th direction

$\hat{\underline{\Sigma}}_{\underline{\theta}}$ covariance of the posterior distribution of the regularized FIR parameters

$\underline{\Sigma}$ covariance matrix of a multivariate pdf

$\underline{\Sigma}_y$ covariance matrix of the output for the marginal likelihood

$\Phi_i(\underline{z})$ validity functions of an LMN

$\mathbb{E}_{x\sim p(x)}$ expected value with respect to the pdf $p(x)$

$\mathcal{H}$ Hilbert space

$\mathcal{L}_2$ Hilbert space of square integrable function

$\mathcal{N}(\underline{\mu}, \underline{\Sigma})$ multivariate Gaussian distribution with mean μ and variance Σ

$\mathcal{U}(a, b)$ Uniform distribution from a to b

$\hat{\cdot}$ estimated quantity

$< f, g >_{\mathcal{H}}$ scalar product of the functions f and g in the Hilbert space $\mathcal{H}$

Abbreviations

Abbreviation	Explanation
Adam	Adaptive moment
AIC	Akaike's information criterion
APRBS	Amplitude modulated pseudo random binary signal
ARX	Autoregressive with exogenous input
BIC	Bayesian information criterion
EGR	Exhaust gas recirculation
BIC	Bayesian information criterion
FIR	Finite impulse response
FIRP	Finite impulse response preserving
GP	Gaussian process
i.i.d.	Independent identically distributed
IIRP	Infinite impulse response preserving
IRP	Impulse response preserving
LMN	Local model network
LOLIMOT	Local linear model tree
MAP	Maximum a posteriori
ML	Maximum likelihood
MPC	Model predictive control
MSE	Mean squared error
NMSE	Normalized mean squared error
NRMSE	Normalized root mean squared error
NARX	Nonlinear ARX
NN	Neural network
NOE	Nonlinear output error
OBF	Orthonormal basis fuction
OE	Output error
pdf	Probability density function

Abbreviation	Explanation
PCA	Principle component analysis
PRBS	Pseudo random binary signal
RBF	Radial basis function
ReLU	Rectified linear unit
RKHS	Reproducing kernel Hilbert space
SGD	Stochastic gradient descent
SNR	Signal-to-noise ratio
SVD	Singular value decomposition
VC	Vapnik-Chervonenkis
VGT	Variable geometry turbocharger

1 Introduction

In recent years, machine learning has seen significant success. Especially, the ability of a method to achieve superhuman performance in the game Go using deep neural networks [117] is considered a significant achievement. Besides, the performance of algorithms on image classification tasks has grown tremendously [62, 132]. These scientific successes have received significant resonance by media. There is, actually, a growing hype about *artificial intelligence*. The ever increasing computational power leads to the prognosis that the cognitive ability of algorithms can become superior to humans [107]. To understand the scientific progress in this field, a reasonable view on this topic is essential. In [58], it is argued that the term *machine learning* is more appropriate to describe the techniques used for recent successes. The success of learning technology is, as explained there, the result of a combination of statistics and computer science. System identification [90, 67] is a field of research which includes beside machine learning a perspective of system and control. Here, temporal dependencies are included in the machine learning problem. This has the consequence that other issues, like the stability of the identified model, become more critical.

1.1 Models – the Key Success Factor for Modern Automation

The result of an identification problem is a valid model of a plant or a process. In this section, we analyze possible application areas of these models. Modern automation systems have a trend to act more autonomously than they did several years ago. It has been analyzed that in several worldwide production plants effects of flexible automation will lead to productivity gains and a tremendous opportunity of producers to refine their market position [34].

From a technical perspective, models play a critical role here. Understanding a process can most often be related to the availability of some sort of model for analysis.

Furthermore, most modern control schemes like *model predictive control* (MPC) [20] or robust control [68] require an explicit model. Besides, models allow for prediction, simulation, and fault diagnosis of processes [90].

There exist several approaches to obtain a model. The most prominent strategy in engineering is to apply first principles for the derivation of a model [55]. For control systems, these models usually result in a system of coupled ordinary differential equations. In most cases, the parameters of the equations are derived by calculations (e.g., calculation of the mass of parts) or specially designed experiments (e.g., derivation of the stiffness of a spring element). Especially for dampings in mechanical models, often heuristic approaches are used. This first principle modeling approach is often termed *white box* approach since it is transparent and does not use any data at all [90]. When measured data is employed to improve the model, e.g., by estimation of the damping coefficients, the model is called *gray box*. If no physical model is assumed at all, the approach is called *black box*. But also for the black box approaches, several assumptions of the model structure are necessary. It has thus been argued that there are several shades of gray for the quantization of the amount of structural assumptions [69]. So, models derived from data become more important for future automation systems and are necessary to achieve productivity gains in production systems.

1.2 Comparison of FIR and IIR Models

To motivate the research done in this thesis, we will analyze the shortcomings of the most prominent examples of dynamic models that are used for system identification. These models are the autoregressive with exogenous input (ARX) model and the finite impulse response (FIR) model. The mathematical derivations of the identification procedures are provided in detail in Sect. 2.2.1. The example presented here is to motivate the research objective of the thesis.

The first order system

$$Y(z) = \frac{(a-1)z}{a-z}U(z) + N(z), \tag{1.1}$$

with $a = 0.9$, $Y(z)$ denoting the z-transformation of the output, $U(z)$ denoting the z-transformation of the input, and $N(z)$ describing the noise is considered. The noise is assumed to be white and Gaussian. If the equation is rewritten in discrete

time it holds that

$$y(k) = 0.1u(k) + 0.9y(k-1) + n(k) - 0.9n(k-1) \tag{1.2}$$

with k denoting the discrete time step. If an ARX model is employed, the model structure is assumed to be

$$\hat{y}(k) = a_1\hat{y}(k-1) + b_0u(k) + n(k). \tag{1.3}$$

Despite being completely the same in the deterministic part, the stochastic part (the part with the noise terms $n(k)$) differs. The consequences that result can be seen when the model is used for identification. Therefore, the system is excited by a pseudo-random binary signal. The input and output signal are shown for 1 000 samples in Fig. 1.1. For the identification, 10^5 samples have been used. This

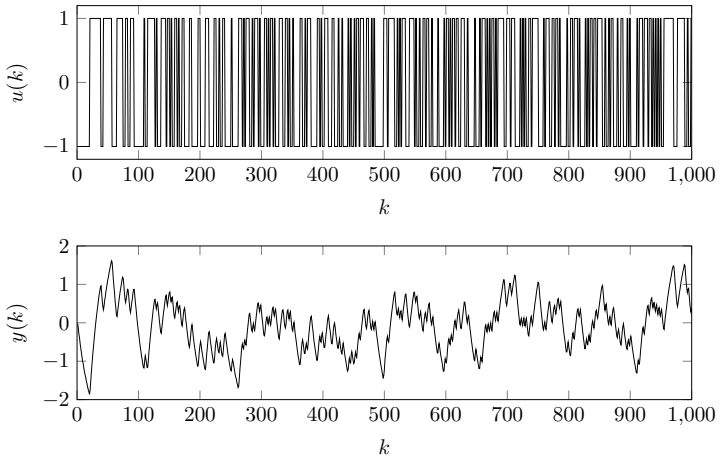

Figure 1.1: Input and output of the first order system

high number is chosen to make clear that not the number of samples is an issue, but the problem lies in the way the coefficients are estimated and the wrong noise assumption.

A first order ARX model is estimated for different noise levels. The identified impulse responses $g(k)$ are shown in Fig. 1.2. It can be seen that for increasing noise variance, the impulse response is represented wrongly, although the model structure is able to represent the deterministic part exactly. This also shows that for system identification, a high number of available data (which is often referred to as big data) is not necessarily sufficient to reach a good model. A significant amount of care has to be taken when the noise of the system is modeled. An inaccurate noise model can deteriorate the performance. If an estimator does not reach the true parameters

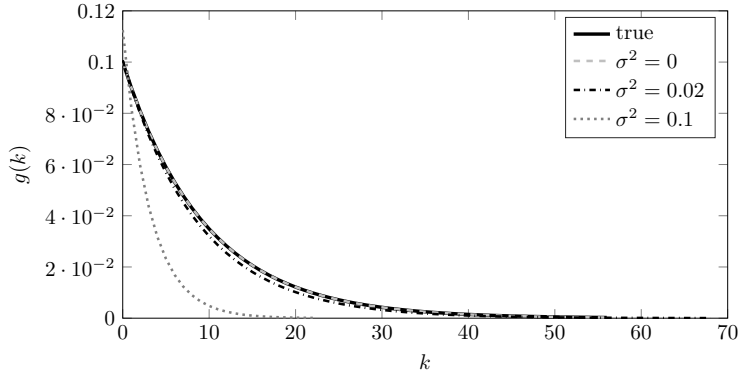

Figure 1.2: Comparison between true impulse response and first order ARX estimates at different variances of the measurement noise.

of the system for $N \to \infty$ with probability 1, then it is inconsistent. So this simple example shows that ARX models are inconsistent if noise enters a system at the output.

A model structure which does not suffer from this inconsistency is the FIR model. For this structure, no feedback is included, and thus the biasing effect of this feedback is mitigated. A significant issue, though, for FIR models is the variance error in the parameter estimate. To demonstrate this, an FIR model is identified with 1 000 samples and different variances of the applied noise. The results are shown in Fig. 1.3. Here, it can be seen that especially the last coefficients of the impulse response are prone to errors and that, as expected, the error grows with the noise level.

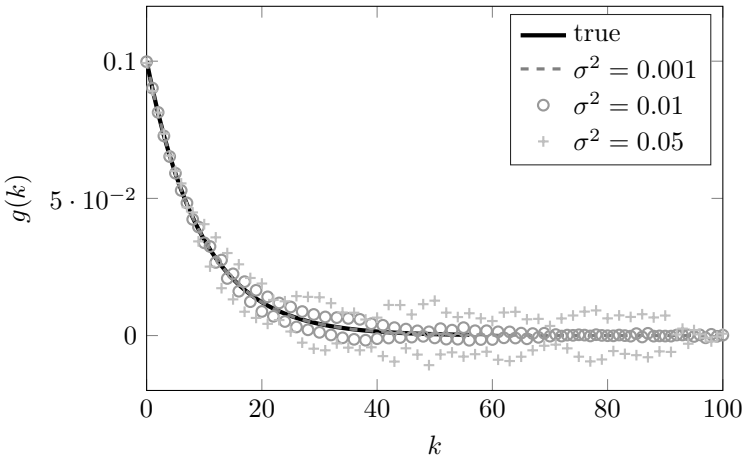

Figure 1.3: Identified models of a first order system using FIR models of order $n = 80$.

The problem can be increased by consideration of a low-pass filtered input signal, which does not excite high frequencies [30]. The identification result is shown in Fig. 1.4. From numerical point of view, the reason for this dilemma is a bad condi-

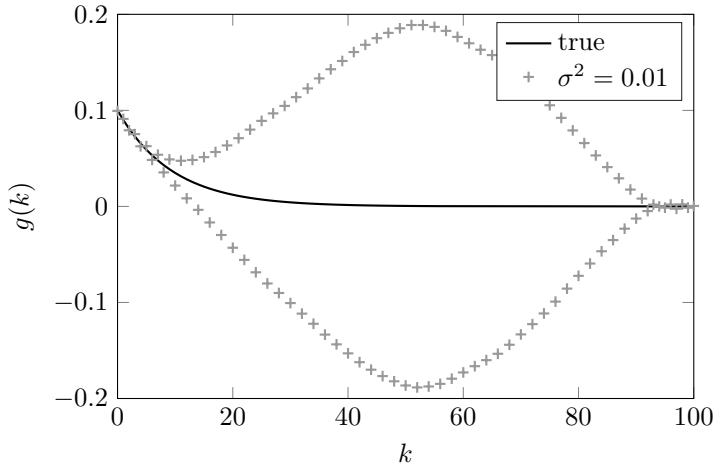

Figure 1.4: Identified impulse response of an FIR model estimated with a low-pass filtered input.

tioning of the pseudo-inverse of the regression matrix.

To summarize the results, the most prominent problem with ARX models is their unrealistic noise model and their resulting inconsistency. The most severe issue for FIR models is their high variance error.

1.3 Structure and Contribution of the Thesis

As demonstrated in the previous section FIR and ARX models suffer from different problems. The goal of this thesis is to use novel regularization techniques to enable the reduction of variance error for FIR systems for both linear and nonlinear problems. The different methods developed for this purpose are shown in Fig. 1.5. For linear systems, there is a significant body of recent work for appropriate regularization of the impulse response [100]. These methods are used to develop three different methods, which are ordered by their interpretability and complexity in the figure. The Impulse Response Preserving (IRP) method from Chap. 3 is a linear method and thus the one with the lowest complexity and best interpretability. The local linear model tree with regularized local FIR models (LOLIMOT RFIR) extends this approach to the nonlinear world in Chap. 4. Here, the coefficients of the local models can be interpreted as impulse responses. Thus, interpretability is worse than in the linear case but better than for the most complex method, the deep regularized FIR neural network (deep RFIR NN), which is described in Chap. 5. In detail, the different chapters of the thesis describe the following:

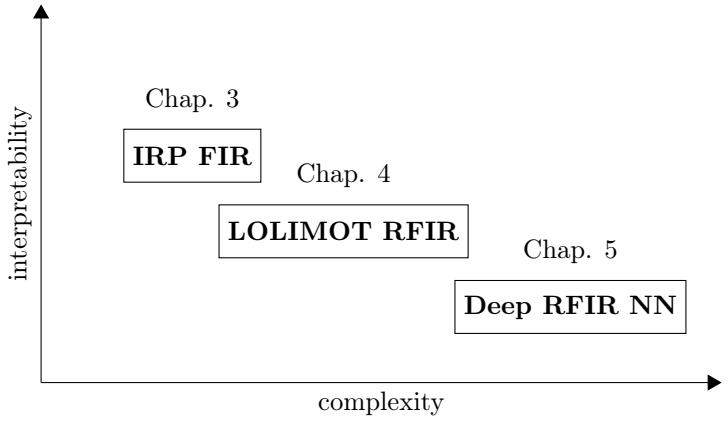

Figure 1.5: Introduced methods in the thesis. IRP FIR: Impulse response preserving FIR, LOLIMOT RFIR: LOLIMOT with regularized local FIR models, Deep RFIR NN: Deep regularized FIR neural network

Foundations of system identification In this chapter, the links between system identification methods and statistics are described. System identification lies at the intersection between computer science, statistics, and control. Therefore, for a holistic understanding of the field, all three disciplines have to be considered. Here, mainly the foundations for the results in the following chapters are laid out. Some results have not been presented in this form. This includes the interpretation of the result of the LOLIMOT algorithm with local estimation as a Gaussian mixture model and the application of the VC bounds to FIR models.

Impulse response preserving identification This chapter proposes a novel method to solve the order selection problem for linear systems. Therefore, it introduces the concept of impulse response preserving filter (IRP) matrices for system identification. Classical methods in system identification like ARX or OE identification impose the order of a model as a rigid concept. The complexity of the model identified is solely described by the order n for these approaches. In contrast, IRP matrices allow for incorporation of prior knowledge of system dynamics for linear systems by a penalty term whose strength can be optimized based on data by generalized cross-validation (GCV). It is shown that incorporation of impulse response preserving filter matrices improves state-of-the-art system identification methods on examples where incorporation of prior knowledge is appropriate.

Regularized Local Model Networks Here, the concept of regularized FIR system identification is transferred to the nonlinear world. Usually, nonlinear FIR (NFIR) systems tend to be very prone to overfitting, due to the high number of parameters required. To circumvent this issue, two measures are adopted. First, local model

networks are used, which allow the variables of the validity functions to be different from the variables of the local models. This allows for a low dimensional nonlinear space. Second, the regularized approach is used for a weighted local regression. The number of parameters required for the local FIR models is then significantly lower compared to unregularized local FIR models. The approach is used with several theoretical examples for both the MISO and MIMO case. Also, a simulated Diesel engine process is investigated and identified.

Deep FIR Neural Networks Recently, deep convolutional neural networks have shown tremendous success in the classification of images. In this chapter, the regularized identification approach is extended to the nonlinear world and combined with a novel regularization approach for the filter coefficients of the deep neural networks. The coefficients of the convolutional filters of the neural network are interpreted as impulse response coefficients. A kernel-based regularization scheme regularizes these. Several options for the design of such networks are investigated on a complicated nonlinear benchmark example. It is shown that competitive state-of-the-art results on this benchmark can be achieved without recurrence in a neural network structure.

2 Foundations of System Identification

In this chapter, a short overview of system identification techniques is given. It starts by introducing concepts from statistical learning, including maximum likelihood (ML) and Bayesian approaches, which form the basis for data based problems. Then, methods for the identification of dynamic systems are presented. Here, special emphasis is given to methods allowing incorporation of prior information for linear systems and their relation to Gaussian processes. Finally, methods for structure selection are revised, since these are applied for the nonlinear method developed in Sect. 4 for structure selection and allow for an understanding of the properties of learning methods. Often system identification is decomposed in static and dynamic systems [90]. So we follow this approach and start with static systems in Sect. 2.1 and continue with dynamical systems in Sect. 2.2. A recent overview of algorithms for regression can also be found in [120].

2.1 Statistical Learning

In this thesis, the term *Statistical Learning* refers to the process of building a model from data. This term is mostly used by statisticians and is part of the title of well known standard books [43, 127]. The usage of this term in statistics is quite common, but in system identification, it is often avoided. It seems that *learning* here is considered somehow exclusive to humans. To emphasize the link between statistics and system identification, the term is employed anyhow. For the description of the learning problem, a data generating distribution $p(\underline{z})$ is considered with $\underline{z}$ denoting the data point. This distribution is the true distribution that generates the data. This concept is very general and contains classification, density estimation and regression as special cases [127]. For the description of a regression problem, the variable $\underline{z}$ contains both, the input $\underline{u}$ and the output y. So, for regression problems, $\underline{z}^T = [\underline{u}^T, y]$ with $\underline{u} \in \mathbb{R}^m$, m denoting the input space dimension and $y \in \mathbb{R}$. The

variable m instead of the common n is used for the input space dimension to mitigate confusion with the order n of an FIR model, see Sect. 2.2.1.

The same concept can also be utilized to describe a classification problem. Here, $\underline{u}$ is again the location in the input space and $c_l \in \{0, \ldots, n_c\}$ is the class label assigned to one of the n_c classes. These are collected in the random variable $\underline{z}^T = [\underline{u}^T, c_l]$ then. In system identification, regression tasks dominate and are thus considered in this short overview. For more information on learning techniques for classification problems, the reader is referred to [51, 43]. For regression, a closer look at the problem statement is important. There are several ways to approach the problem from a theoretical perspective, and a sound understanding of the results obtained by each viewpoint makes it necessary to distinguish some assumptions. As stated above, the regression problem is characterized by a joint probability distribution $p(\underline{u}, y)$. According to the definition of conditioned probability this distribution can be decomposed as $p(\underline{u}, y) = p(y|\underline{u})p(\underline{u})$. For regression, usually one is interested only in the first part $p(y|\underline{u})$, and in practice, interest is often limited even more and only the conditioned expected value

$$\mathbb{E}_{y \sim p(y|\underline{u})} y = \int y \, p(y|\underline{u}) dy \tag{2.1}$$

is considered.

The first and most common assumption is that for the conditional probability of $p(y|\underline{u})$ a parametric model depending on some parameter $\underline{\theta}$ is available. To denote the model, the hat symbol "$\hat{\cdot}$" will be used. Now, the model $\hat{p}(y|\underline{u}, \underline{\theta})$ is given and it is assumed that it matches the true conditional probability distribution $p(y|\underline{u}) = \hat{p}(y|\underline{u}, \underline{\theta}_0)$ for the true value of the parameter $\underline{\theta}_0$. For the real world, this assumption is, in many cases, fairly optimistic. However, the existence of some true $\underline{\theta}_0$ makes things easier from a theoretic perspective. In fact, the development of the Maximum Likelihood (ML) method [39, 40, 41], described in detail in Sect. 2.1.1, is based on this assumption. Another important consequence is the behavior of the regression estimate if the data distribution $p(\underline{u})$ changes. Since model and true distribution coincide, if the estimated parameters $\hat{\underline{\theta}}$ equal the true parameters $\underline{\theta}_0$, the input space distribution can vary arbitrarily without affecting the validity of the model.

Extrapolation is also not a problem since by definition for every $\underline{u}$, the probability $p(y|\underline{u}, \hat{\theta})$ is correct. In reality, these changes often matter significantly. Thus, results obtained by making this assumption require special care in reality.

The second approach deals with this problem differently. Also here a parametric model $\hat{p}(y|\underline{u}, \underline{\theta})$ is given. The goal is now to measure and optimize the risk [127]

$$R(\hat{y}(\underline{u}, \underline{\theta})) = \int (y(\underline{u}) - \hat{y}(\underline{u}, \underline{\theta}))^2 p(\underline{u}, y)\, dy\, d\underline{u} \tag{2.2}$$
$$= \mathbb{E}_{\underline{u},y\sim p(\underline{u},y)} \left(y(\underline{u}) - \hat{y}(\underline{u}, \underline{\theta})\right)^2 .$$

Instead of estimating a whole probability distribution, here, only the so-called regression function $\hat{y}(\underline{u}, \underline{\theta})$ is estimated. This risk is the integral of the squared deviation of the model from the true function weighted by the probability distribution of the data. It is important to notice that in this definition of risk, the probability distribution $p(\underline{u})$ of the input space is included and is crucial for an interpretation of error values. If the assumption of a true model is dropped, the input distribution $p(\underline{u})$ plays a crucial role in the determination of an error. In reality, neglection of this fact can lead to a wrong assessment of the error on novel data, especially if dynamic models are considered. The obtained error is only valid for the distribution of input data which has been used during training. If this distribution is changed, the obtained risk is not valid anymore.

2.1.1 Maximum Likelihood

In reality one has no access to the true underlying probability distribution $p(\underline{u}, y)$. There are usually independent identically distributed (i.i.d.) samples from the distribution available. These samples are denoted by $\underline{z}(i) = \left[\underline{u}(i)^T, y(i)\right]^T$ and are stacked in the matrix $\underline{Z} = [\underline{z}(1), \ldots, \underline{z}(N)]^T$ with N denoting the number of data points. The central question is now how to obtain the parameter $\underline{\theta}$ from $\underline{Z}$. In statistics every function, which assigns a value of $\hat{\underline{\theta}}$ to observed $\underline{Z}$ is called an estimator [130]. One method to obtain such an estimator has been developed by R. A. Fisher in a series of papers [39, 40, 41] and has been called the maximum likelihood (ML) method. Since the mathematics of probability has been under development at this time, the arguments given in these papers do not entirely coincide with nowadays notation, for an overview of the historical development see [118]. Now, a parametric model of the probability distribution $\hat{p}(\underline{z}|\underline{\theta})$ is considered which depends on some parameters $\underline{\theta}$. Maximum likelihood works as follows: The parameters of the parametric probability distribution $\hat{p}(\underline{z}|\underline{\theta})$ are chosen such that the probability of the observed data is maximized. Usually, it is assumed that the samples are identical independently distributed (i.i.d). Thus, e.g., it holds that $p(\underline{z}(1), \underline{z}(2)) = p(\underline{z}(1))p(\underline{z}(2))$. For N

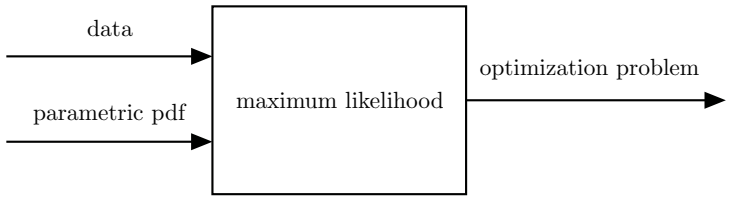

Figure 2.1: Schematic illustration of the maximum likelihood method. Data and a parametric probability density function are transformed to an optimization problem for parameter estimation.

data points, the optimization problem

$$\underset{\underline{\theta}}{\text{maximize}} \quad \prod_{i=1}^{N} \hat{p}(\underline{z}(i)|\underline{\theta}) \tag{2.3}$$

has to be solved. The logarithm is a monotonic function. Thus, taking the logarithm of a pdf does not change the optimal parameter values. Furthermore, the product can be rewritten as a sum such that the following optimization problem of the so-called log-likelihood results

$$\underset{\underline{\theta}}{\text{minimize}} \quad -\sum_{i=1}^{N} \log p(\underline{z}(i)|\underline{\theta}). \tag{2.4}$$

By these steps, the problem of statistical estimation has been transformed to an optimization problem. The general idea is illustrated in Fig. 2.1. Samples of a probability distribution (data) and a parametric pdf are transformed into an optimization problem by the ML method. It has already been apparent to Gauss [46] that this type of estimator is reasonable. Assuming that there is an exact $\underline{\theta}_0$ such that the data has been sampled from the probability distribution $\hat{p}(\underline{z}|\underline{\theta}_0)$ and several regularity conditions hold, Maximum Likelihood enjoys several favorable properties, which will be reviewed briefly. These properties are also illustrated in Fig. 2.2.

Consistency The first property is consistency, see [39, 130]. It means that if the number of data tends to infinity, the true value $\underline{\theta}_0$ for $\underline{\theta}$ is recovered. More formally this property is defined as

$$\lim_{N\to\infty} \|\hat{\underline{\theta}}_N - \underline{\theta}_0\|_\infty \xrightarrow[N\to\infty]{P} 0. \tag{2.5}$$

Here, $\hat{\underline{\theta}}_N$ denotes the ML estimate with N data points and $\underline{\theta}_0$ the true parameter. The equation means that the probability of estimates deviating more than a small ϵ from the true value tends to zero as the number of data points tends to infinity. This explains some of the success big data approaches had recently. For many of these

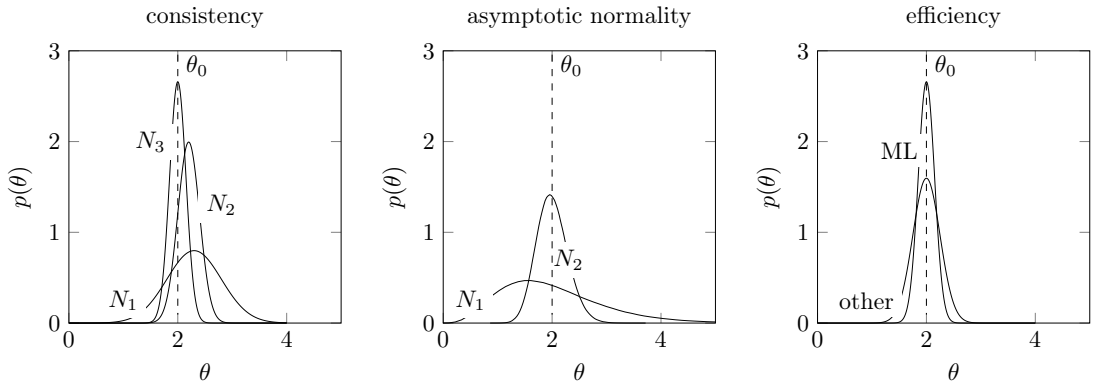

Figure 2.2: Properties of the ML method. The consistency of the ML estimator for the pdf of a single parameter $\hat{\theta}$ is illustrated for the number of parameters $N_1 < N_2 < N_3$. The asymptotic normality (middle) is demonstrated for the number of parameters $N_1 < N_2$. The shape of the pdf converges to a normal distribution. The ML method is asymptotically efficient (right). The pdf of any other estimator has a higher variance than the ML estimator.

approaches, the case $N \to \infty$ can be achieved approximatively. Big data systems include applications like image recognition, where large labeled datasets are available, e.g. [62]. In mechanical engineering or production, this, however, is often not the case. Available data is limited, and often, for some products, only a few samples are available. In this case, other techniques, e.g. Bayesian approaches described in Sect. 2.1.5 can be more advantageous.

Asymptotic Normality Another fact which can be analyzed for maximum likelihood estimators is what happens to the distribution of the estimated parameters if the number of observations tends to infinity. This analysis has been firstly conducted in [40, 32]. For $N \to \infty$ the distribution of the parameters is given according to

$$\lim_{N\to\infty} p(\hat{\underline{\theta}}_N) = \lim_{N\to\infty} \mathcal{N}\left(\hat{\underline{\theta}}_N | \underline{\theta}_0, \frac{1}{\sqrt{N}} \underline{I}_F^{-1}(\hat{\underline{\theta}}_N)\right). \tag{2.6}$$

The m-dimensional multivariate Gaussian distribution with mean vector $\underline{\mu}$ and covariance matrix $\underline{\Sigma}$ is described by

$$\mathcal{N}(\underline{x}|\underline{\mu}, \underline{\Sigma}) = \frac{1}{\sqrt{(2\pi)^m \det(\underline{\Sigma})}} \exp\left(-\frac{1}{2}(\underline{x}-\underline{\mu})^T \underline{\Sigma}^{-1}(\underline{x}-\underline{\mu})\right). \tag{2.7}$$

So, asymptotically the estimate $\hat{\underline{\theta}}_N$ has the mean of the true parameters $\underline{\theta}_0$. This property is called *asymptotic unbiasedness*. In general the ML estimator is not unbi-

ased for finite N [90]. For the calculation of the covariance, the Fisher information matrix $\underline{I}_F(\underline{\theta})$ is defined as

$$\underline{I}_F(\underline{\theta}) = -\int \frac{\partial^2 \log \hat{p}(\underline{Z}|\underline{\theta})}{\partial \underline{\theta}^2} p(\underline{Z}) d\underline{Z}. \tag{2.8}$$

The result (2.6) provides more information than consistency only. It also provides an understanding of the variance of the solution for $\hat{\theta}_N$. A reasonable estimate for the information matrix can be found by replacing the expectation by its empirical counterpart

$$\hat{\underline{I}}_F(\hat{\underline{\theta}}_N) = \sum_{i=1}^{N} \frac{\partial^2 \log \hat{p}(\underline{z}(i)|\underline{\theta})}{\partial \underline{\theta}^2}\bigg|_{\underline{\theta}=\hat{\underline{\theta}}_N}. \tag{2.9}$$

In contrast to (2.8) the expression (2.9) can be calculated based on the available data. To do so, the second derivatives of the log-likelihood function have to be calculated. These are then evaluated at the locations of the data points.

Asymptotic Efficiency A natural question that arises is if there could be another rule to construct an estimate, which has lower variance than the maximum likelihood estimate. This property has been analyzed in [32]. It can, however, be shown that the maximum likelihood estimator is the most efficient asymptotically unbiased estimator [130]. Practically this means that if the number of data available is big, then there is no better way to estimate the parameter than the usage of ML estimation.

Relation to Information Theory An exciting and practically highly relevant question is what happens if the true probability distribution $p(\underline{z})$ cannot be reached by the modeled probability distribution $\hat{p}(\underline{z}|\underline{\theta})$. This could, e.g., occur if the true pdf $p(\underline{z})$ is a uniform distribution and $\hat{p}(\underline{z}|\underline{\theta})$ is modeled as a Gaussian distribution. An answer to this question can be found from an information theoretic viewpoint to ML [5]. The goal, in this case, cannot be to recover $\underline{\theta}_0$ as described in the previous paragraph but to find parameters $\underline{\theta}^*$ for the modeled probability distribution which make the modeled pdf, in some sense, *most similar* to the true probability distribution. This notion of closeness has to be made more explicit mathematically.

One way to describe how good one probability distribution is described by another is the so-called Kullback-Leibler (KL) divergence [51]

$$\mathrm{KL}\left(p(\underline{z})||\hat{p}(\underline{z}|\underline{\theta})\right) = \int \log \frac{p(\underline{z})}{\hat{p}(\underline{z}|\underline{\theta})} p(\underline{z}) d\underline{z}. \tag{2.10}$$

To illustrate the behavior of the KL divergence, a simple example is considered. Let

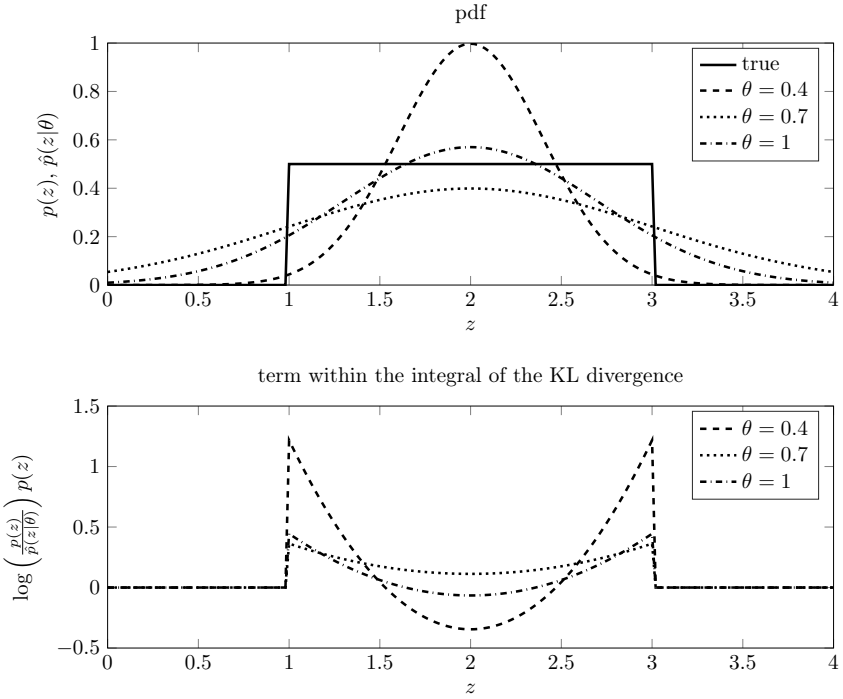

Figure 2.3: Term within the integral of the KL divergence between a uniform distribution and Gaussian distributions with correct mean and varying standard deviations θ.

the true pdf be given by a uniform distribution between 1 and 3, so $p(z) = \mathcal{U}(z|1,3)$, and let the model be given by a Gaussian distribution with fixed mean 2 and the standard deviation as a free parameter, so $\hat{p}(z|\theta) = \mathcal{N}(z|2,\theta^2)$. There is no θ so that $\hat{p}(z|\theta)$ can be equal to the true pdf $p(z)$. The densities for different values of θ are shown in the upper part of Fig. 2.3. The lower part shows the term $\log \frac{p(z)}{\hat{p}(z)} p(z)$ which is the argument of the integral. For the uniform distribution the value of $p(z)$ is zero outside the interval $[1,3]$. Since $\lim_{x\to\infty} \log(x)x = 0$, the term within the integral is plotted as zero there.

Two things can be seen in the lower plot of the figure. First, the KL divergence assigns high values to regions where the pdf of the true distribution is higher than the modeled pdf. Second, for regions where the pdf of the true distribution is lower than the modeled one, a negative value can occur ($\theta = 0.4$ at $z = 2$ in the figure). Thus, the KL divergence makes sure that everywhere, where the true distribution is high, a high probability of the model is assigned. The parameter θ can now be optimized to minimize the KL divergence between $\hat{p}(z,\theta)$ and $p(z)$. This results in $\theta = 0.764$.

For ML estimation it can now be shown that exactly this θ will be discovered asymp-

totically. To see this, the definition of the KL divergence is considered and written as an expected value

$$\mathrm{KL}\left(p(\underline{z}), \hat{p}(\underline{z}|\underline{\theta})\right) = \mathbb{E}_{\underline{z}\sim p(\underline{z})}\left(\log \frac{p(\underline{z})}{\hat{p}(\underline{z}|\underline{\theta})}\right). \tag{2.11}$$

This expected value can also be written as

$$\mathrm{KL}\left(p(\underline{z}), \hat{p}(\underline{z}|\underline{\theta})\right) = -\mathbb{E}_{\underline{z}\sim p(\underline{z})}\left\{\log \hat{p}(\underline{z}|\underline{\theta})\right\} + \mathbb{E}_{\underline{z}\sim p(\underline{z})}\left\{\log p(\underline{z})\right) \tag{2.12}$$

by rewriting the logarithm. Since the second term is independent of the parameters $\underline{\theta}$, the minimization of the KL divergence is equivalent to the maximization of $\mathbb{E}_{\underline{z}\sim p(\underline{z})}\left\{\log \hat{p}(\underline{z}|\underline{\theta})\right\}$. This can be written as the optimization problem

$$\underset{\underline{\theta}}{\text{maximize}} \int \log \hat{p}(\underline{z}|\underline{\theta}) p(\underline{z}) d\underline{z}. \tag{2.13}$$

When it is possible to draw samples from the true distribution $p(\underline{z})$ then with an empirical estimate of (2.13) the optimization problem can be rewritten as

$$\underset{\underline{\theta}}{\text{minimize}} -\sum_{i=1}^{N} \log p(\underline{z}(i)|\underline{\theta}). \tag{2.14}$$

This expression is the same as (2.4). Thus ML estimation corresponds to the minimization of the KL divergence between the estimated and the true distribution. This means that also in the case the true and the estimated distribution do not coincide, the obtained estimate will for $N \to \infty$ be near (in the sense of a small KL divergence) to the true solution.

Example To exemplify the usage of the method, the mean of a Gaussian distribution is estimated by the maximum likelihood method [108]. The one-dimensional Gaussian distribution is described by

$$\hat{p}(z, \mu) = \frac{1}{2\pi} \exp\left(-\frac{1}{2}(z-\mu)^2\right) \tag{2.15}$$

with unit variance and unknown mean value μ. The logarithm of this distribution results in

$$\log \hat{p}(z, \mu) = -\log(2\pi) - \frac{1}{2}(z-\mu)^2. \tag{2.16}$$

This is the log-likelihood of one sample. For many samples concatenated in a vector $\underline{z}$, this results in

$$\log \hat{p}(\underline{z}, \mu) = \sum_{i=1}^{N} \left(-\log(2\pi) - \frac{1}{2} \left(z(i) - \mu \right)^2 \right) = -N \log(2\pi) - \frac{1}{2} \sum_{i=1}^{N} \left(z(i) - \mu \right)^2 . \tag{2.17}$$

To maximize the likelihood, the derivative with respect to μ is calculated according to

$$\frac{\partial \log \hat{p}(\underline{z}, \mu)}{\partial \mu} = \sum_{i=1}^{N} \left(\mu - z(i) \right) . \tag{2.18}$$

Setting this derivative to zero results in the maximum likelihood estimator

$$\mu = \frac{1}{N} \sum_{i=1}^{N} z(i). \tag{2.19}$$

This estimator is unbiased. In general, this only holds in the asymptotic case for ML estimators ($N \to \infty$).

Discussion of the Assumptions The results of this section are very strong. In fact, they say that you cannot do better than using an ML estimator, if there is a parameter for the modeled pdf which describes the true pdf exactly, and if there are very many data points. Unfortunately, in reality, this does not hold and several problems occur. The first is that in case of a misspecified distribution, no statement about efficiency can be made. Furthermore, it is possible that convergence to an appropriate value of the parameter happens only very slowly. Another, maybe practically more severe effect, are potential local optima of the likelihood function. The likelihood function is not necessarily convex or unimodal. We will discuss Least-Squares in Sect. 2.1.2, which is the most important case for the maximum likelihood of having an analytic unimodal solution. In general, this does not hold. For linear output error identification introduced in Sect. 2.2.1 multiple local maxima can exist for the likelihood, and the same is true for the maximum likelihood identification of multilayer neural networks. So, if the global optimum is not found, the guarantees found in this section may not hold at all. Another problem is that all the results only hold asymptotically. In reality, the number of samples will always be limited. A class of methods which often performs well in the so-called low data regime are Bayesian methods described in Sect. 2.1.5. But, even if an assumption is violated for ML it does not necessarily mean that it fails completely. Often the resulting models show reliable behavior in reality.

2.1.2 Linear Regression

The term *linear regression* refers to problems which are linear in their parameters. Mathematically an n_x-dimensional feature $\underline{x}$ is mapped to an output $\hat{y}$. Since this mapping is linear, it can be represented by a scalar product with the parameter vector $\underline{\theta}$. Thus, the model can be written as

$$\hat{y}(k) = \underline{x}(k)^T \underline{\theta}. \tag{2.20}$$

There are various possible choices for the entries of $\underline{x}(k)$, which are also called *features* [43]. It can be the input itself, contain a function of an element of the input of the system, e.g. $\log(u_1(k))$, or it can be an interaction between two inputs $u_1(k) \cdot u_2(k)$. To identify the parameters $\underline{\theta}$ from N measured output samples $y(k)$, an identification criterion is necessary. A widespread identification criterion is the mean squared error (MSE)

$$J = \frac{1}{N} \sum_{k=1}^{N} \left(y(k) - \hat{y}(k) \right)^2 . \tag{2.21}$$

Beside often used, it is not directly clear that this error criterion is the best choice. In [67], Ljung describes that this error is useful for its own right. This problem has already been in the mind of Gauss when discovering this method [46]. In his argumentation, two models are compared. The first model makes a mistake at one observation, which is as big as the sum of errors made by another model at two different points. The absolute sum of errors made by these models is now equal. So, if the sum of absolute values is employed as an error measure, the performance of both models is the same. If, in contrast, the squared error is chosen as a loss function, the model with two smaller errors is significantly better. In his argumentation, which of the two models is to be preferred is not a matter of science but a matter of the purpose of the model. It is interesting to note that this observation made at the beginning of the nineteenth century is very much in accordance with modern machine learning literature, arguing that the intended application of the learned model should determine the loss function. [51, 3]. Another reason to choose the squared loss is that the resulting optimization problem can be solved very efficiently, as shown in the next paragraph. Therefore, the features are stacked in a *regression* matrix

$$\underline{X} = \begin{bmatrix} \underline{x}(1) & \underline{x}(2) & \dots & \underline{x}(N) \end{bmatrix}^T \tag{2.22}$$

and the obtained outputs are stacked in the output vector

$$\underline{y} = \begin{bmatrix} y(1) & y(2) & \dots & y(N) \end{bmatrix}^T . \tag{2.23}$$

A model structure which assumes a linear relationship between $\underline{X}$ and $\underline{y}$

$$\underline{\hat{y}} = \underline{X}\,\underline{\theta} \tag{2.24}$$

with the parameters $\underline{\theta}$ is called a linear model. This model is linear in the parameters but not necessarily linear in the inputs since it is easy to introduce a nonlinearity in the input by using e.g. $x_1^2(k)$ as a new feature. The well-known solution to the linear regression problem is [90]

$$\underline{\hat{\theta}} = \left(\underline{X}^T \underline{X}\right)^{-1} \underline{X}^T \underline{y} \tag{2.25}$$

which can be derived by simply setting the derivative of the loss function to zero. From an algorithmic perspective, this equation is solved using the QR factorization of $\underline{X}$

$$\underline{X} = \underline{Q}_X \, \underline{R}_X \tag{2.26}$$

with the orthonormal matrix $\underline{Q}_X$ and the upper tridiagonal matrix $\underline{R}_X$. Substituting results in

$$\underline{\hat{\theta}} = \underline{R}_X^{-1} \underline{Q}_X^T \underline{y}. \tag{2.27}$$

The computational complexity can be separated into three parts [15]. The first is the computation of the QR factorization, which has the computational complexity $2Nn_x^2$. The second is the computation of $\underline{s} = \underline{Q}_X^T \underline{y}$, which is a simple matrix-vector multiplication, and thus of complexity $n_x N$. Finally, the third component is the computation of $\underline{R}^{-1}\underline{s}$ which is done by back-substitution and has the complexity n_x^2. So the overall complexity is $2Nn_x^2 + Nn_x + n_x^2$, whereby for high N, the first part is dominating. It is remarkable that the complexity of least-squares grows only linearly with the number of data points. This means that for moderately sized datasets, e.g. up to 1 Mio. data points, a closed-form estimate of $\underline{\hat{\theta}}$ can be obtained.

The quadratic error criterion can also be derived using the arguments of the ML method. Therefore, the density to be identified is the probability of the output given the features $p(y|\underline{x}, \underline{\theta})$. For this density the parametric form

$$p(y(k)|\underline{x}(k), \underline{\theta}) = \mathcal{N}(y(k)|\underline{x}(k)^T \underline{\theta}, \sigma^2) \tag{2.28}$$

is assumed. Taking the logarithm results in the log-likelihood function

$$\log p(y(k)|\underline{x}(k),\underline{\theta}) = -\frac{1}{2\sigma^2}\left(y(k) - \underline{x}(k)^T\underline{\theta}\right)^2 - \frac{1}{2}\log(\sigma^2) - \frac{1}{2}\log 2\pi \tag{2.29}$$

Since only the first term depends on the parameter $\underline{\theta}$, the maximum of the likelihood of N i.i.d. samples can be found by

$$\hat{\underline{\theta}} = \arg\underset{\underline{\theta}}{\text{minimize}} \sum_{k=1}^{N}\left(y(k) - \underline{x}(k)^T\underline{\theta}\right)^2 \tag{2.30}$$

which is the least squares solution.

2.1.3 Neural Networks

Initially, the term *neural network* has been chosen to refer to the structure of the human brain [108]. Nowadays, the term neural network is understood in a very general way as a representation of an artificial structure consisting of similar units (neurons), which learn a mapping from an input to an output. A commonly made distinction is between wide and deep networks. A wide network consists of multiple units connected in parallel. An radial basis function (RBF) network with a Gaussian kernel [90]

$$\hat{y} = \sum_{i=1}^{M} a_i^{\mathrm{RBF}} \exp\left(-\frac{1}{2\sigma^2}\left\|\underline{u} - \underline{c}_i^{\mathrm{RBF}}\right\|_2^2\right) \tag{2.31}$$

is a wide network with M neurons connected in parallel. The parameters $\underline{a}^{\mathrm{RBF}}$ are the heights of the basis functions, σ^2 is the variance which determines the width of the basis function, and $\underline{c}_i^{\mathrm{RBF}}$ are the centers. These networks have the advantage that a large number of parameters (in this case, $\underline{a}^{\mathrm{RBF}}$ are linear). Thus, the least-squares estimate can be calculated analytically by linear regression. For a detailed solution procedure, especially the decomposition of the problem into linear and nonlinear parameters, of this optimization problem the reader is referred to [90]. Deep neural networks [108, 51, 63] look different. Here multiple layers are stacked and the output is a concatenation of nonlinear functions with unknown parameters. A two layer rectified linear unit (ReLU) network is described by

$$\hat{y}(k) = \underline{b}^{\mathrm{NN}}\underline{\sigma}_{\mathrm{ReLU}}(\underline{D}^{\mathrm{NN}}\,\underline{h}(k)) + c^{\mathrm{NN}} \tag{2.32}$$

$$\underline{h}(k) = \underline{\sigma}_{\mathrm{ReLU}}(\underline{E}^{\mathrm{NN}}\,\underline{u}(k)) \tag{2.33}$$

These structure has a hidden unit described by $\underline{h}(k)$ and the function $\sigma_{\mathrm{ReLU}}(x) = \max(x, 0)$ is the so-called ReLU nonlinearity. The variables $\underline{b}^{\mathrm{NN}}$, $\underline{D}^{\mathrm{NN}}$, c^{NN} and $\underline{E}^{\mathrm{NN}}$ are learnable. Modern NN consist of more than two layers normally. This works by stacking layers described by (2.33) one after another. Compared to wide neural networks which utilize the analytical least-squares solution for linear regression extensively, the parameters of these structures are usually learned with first-order nonlinear optimization methods, such as stochastic gradient descend (SGD) [51].

2.1.4 Local Model Networks

Deep neural networks, which consist of multiple stacked layers of nonlinearities, are hardly interpretable. One option to increase the interpretability is to calculate the output of the neural network by a weighted combination of several local models. Such a structure is called a local model network (LMN) [86]. It can be described by

$$\hat{y}(\underline{x}, \underline{z}) = \sum_{i=1}^{M} L_i(\underline{x}) \Phi_i(\underline{z}) \tag{2.34}$$

with $L_i(\underline{x})$ denoting the local models, $\Phi_i(\underline{z})$ denoting the validity functions, and M denoting the number of local models. The validity functions depend on the argument $\underline{z}$ and the local models depend on $\underline{x}$. One possibility is that the input spaces for the local models and the validity functions are identical ($\underline{z} = \underline{x}$). Often, it can be advantageous to use different input spaces for the linear models and the nonlinear validity functions [90]. This is applied for local FIR models in Chap. 4. The validity functions have the property to sum to 1. This can be ensured by introducing local membership functions $\mu_i(\underline{z})$ and to compute the validity functions by

$$\Phi_i(\underline{z}) = \frac{\mu_i(\underline{z})}{\sum_{j=1}^{M} \mu_j(\underline{z})}. \tag{2.35}$$

One common approach for choosing the membership function, also adopted by the local linear model tree (LOLIMOT) algorithm [89], are Gaussian functions of the form

$$\mu_i(\underline{z}) = \exp\left(-\frac{1}{2} \underline{z}^T \underline{\Sigma}_i \, \underline{z}\right). \tag{2.36}$$

The orientation of the membership function is usually chosen orthogonally. This results in a diagonal covariance matrix with entries

$$\Sigma_i = k_\sigma \mathrm{diag}(\Delta_{ij}) \tag{2.37}$$

with $k_\sigma = \frac{1}{3}$ (recommended value) and Δ_{ij} denoting the width of the i-th local model in the j-th direction [90]. This idea has been proposed under several names. The names validity and membership functions stem from the fuzzy interpretation of these systems [123]. Thus, this type of system is often named after its inventors as Takagi Sugeno (TS) fuzzy system. It has been shown that this structure type is a universal approximator [133], e.g. it can approximate any nonlinear function arbitrarily well. Another interpretation is the mixture of experts formulation [56]. For dynamic systems, the term local model network has been proposed by Murray-Smith [87] is often employed. The local models are usually chosen, such that its parameters are linear. Thus estimation of the linear parameters of the local models can be done analytically.

For the estimation of the linear parameters two ways are possible. One is the so-called global estimation approach. The optimal parameters are determined by jointly optimizing all linear parameters. The second is the so-called local optimization approach, which computes the linear parameters by a local weighted linear regression.

Global Estimation The global estimation approach optimizes all linear parameters jointly [90]. This results in the following optimization problem for identification of the linear parameters

$$\underset{\underline{\theta}_1^L,\ldots,\underline{\theta}_M^L}{\text{minimize}} \quad \sum_{k=1}^{N} \left(y(k) - \sum_{i=1}^{M} L_i(\underline{x}(k), \underline{\theta}_i^L) \Phi_i(\underline{z}(k)) \right)^2 . \tag{2.38}$$

Here, $\underline{\theta}_i^L$ denotes the part of the parameters on which the i-th local model depends linearly. For local linear model this will be the slopes and the offset. If it is assumed that the system generating the data is disturbed by Gaussian noise with constant variance σ^2, the probability distribution of the output is given by

$$p(y(k)|\underline{x}(k), \underline{z}(k)) = \mathcal{N}\left(y(k) \middle| \sum_{i=1}^{M} L_i(\underline{x}(k), \underline{\theta}_i^L) \Phi_i(\underline{z}(k)), \sigma^2 \right) . \tag{2.39}$$

Then the ML estimator of the parameters corresponds to (2.38).

Local Estimation Local estimation of LMNs works differently. Instead of minimization of the whole loss for all models, the local loss for each model is computed. The global estimation approach optimizes all linear parameters jointly. This results

in the following optimization problem for identification of the linear parameters

$$\underset{\underline{\theta}_1^L,\ldots,\underline{\theta}_M^L}{\text{minimize}} \quad \sum_{k=1}^{N}\sum_{i=1}^{M}\Big(y(k) - L_i(\underline{x}(k),\underline{\theta}_i^L)\Phi_i(\underline{z}(k))\Big)^2. \tag{2.40}$$

Also here, a maximum likelihood interpretation can be found. Therefore, the following probability distribution is considered

$$p(y(k)|\underline{x}(k),\underline{z}(k)) = \sum_{i=1}^{M}\mathcal{N}(y(k)|L_i(\underline{x}(k)),\sigma^2)\Phi_i(\underline{z}(k)) \tag{2.41}$$

To exemplify the difference, a one dimensional example is chosen. The two local models are $L_1(x) = 0.8x + 0.2$, $L_2(x) = -0.5x + 0.5$, the variance $\sigma^2 = 0.1$ and the membership functions $\mu_1(x) = \exp\left(-\frac{1}{25}(x-0.25)^2\right)$ and $\mu_2(x) = \exp\left(-\frac{1}{25}(x-0.75)^2\right)$. Here, for the z-input space it holds that $x = z$. For the local estimation case the probability distribution is given by

$$p(y|x) = \mathcal{N}(y|L_1(x),\sigma^2)\Phi_1(x) + \mathcal{N}(y|L_2(x),\sigma^2)\Phi_2(x) \tag{2.42}$$

and for the global estimation case

$$p(y|x) = \mathcal{N}(y|L_1(x)\Phi_1(x) + L_2(x)\Phi_2(x),\sigma^2). \tag{2.43}$$

The probability distribution for both cases are illustrated in Fig. 2.4. It can be seen that in the area where the local models are blended the behavior is different. While the global approach has a Gaussian shape around the mean value of the function, for the local estimation approach the function becomes multimodal. Interestingly, both functions have the same expected values but differ significantly in their pdfs.

Construction of Validity Functions For the construction of the validity functions, several options are possible. One approach is to apply the Gustaf-Kesselson clustering algorithm [2] on the available data. Also, algorithms generating splits based on clustering in the parameter space [81] are possible. Alternative algorithms which are more robust are tree-based splitting techniques [89, 88, 91, 52]. These algorithms are inspired by tree construction methods, e.g. by CART [17]. A tree-based construction algorithm for local model networks is the local linear model tree (LOLIMOT) algorithm [89, 88]. The working principle is depicted in Fig. 2.5. The idea is to start with one local model and to conduct a stepwise splitting of the input space. In the figure, the input space has two dimensions x_1 and x_2. This input space is then

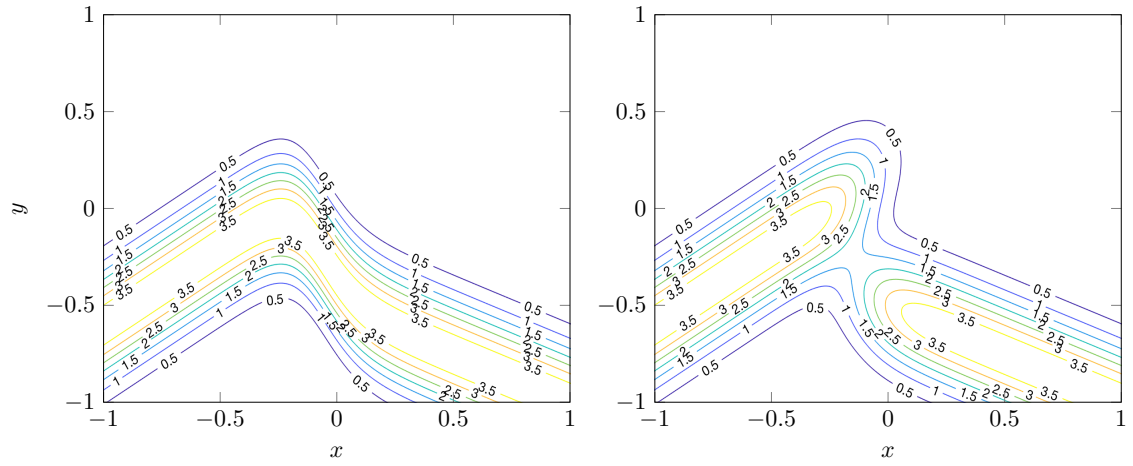

Figure 2.4: Probability distribution of a globally (left) and locally estimated (right) LMN.

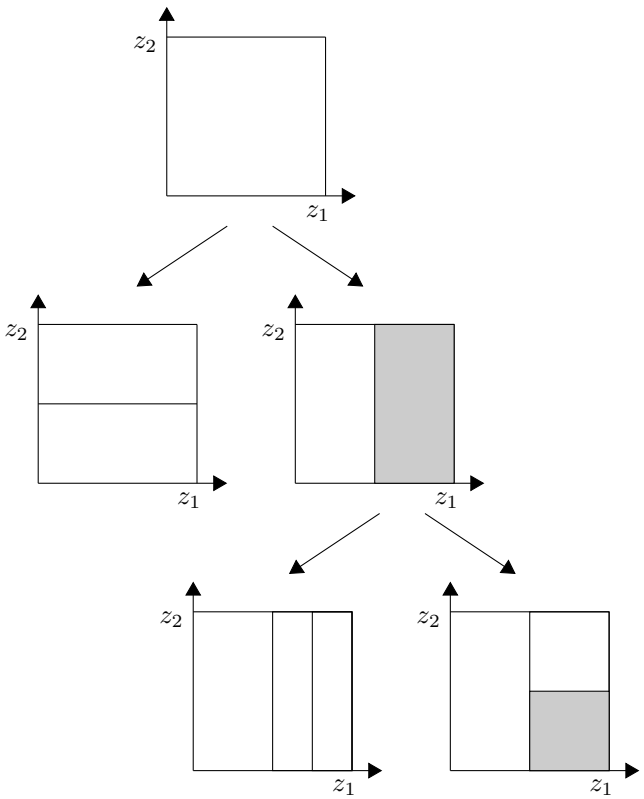

Figure 2.5: Principle of the LOLIMOT algorithm. The worst local model is subsequently split orthogonally [90].

split orthogonally in each dimension. Therefore, two membership functions μ_1 and μ_2 are constructed, which are centered in the middle of the two new local models. The parameters of the new local models are estimated in the next step. Afterwards, the local loss of each local model is computed. Based on this local loss, it is decided which model is to be split next. This worst model is split again. This algorithm is continued until a termination criterion is met. The error on training data decreases monotonously with an increasing number of splits. To stop the training process, a complexity selection criterion, which is described in detail in Sect. 2.3, is used. Usually, the corrected Akaike's information criterion ($\mathrm{AIC_c}$) works well [52].

2.1.5 Bayesian Approach

Methods which assign single values to parameters as an estimate are also known as frequentist methods [43]. In contrast, Bayesian methods compute a probability distribution of the parameters using Bayes rule and a prior probability. Therefore, we consider that a distribution of the parameters $p(\underline{\theta})$ is given. Furthermore, a probabilistic model $p(\underline{y}|\theta)$ is available. The goal of Bayesian identification (or inference) is to estimate the posterior probability

$$p(\underline{\theta}|\underline{y}) = \frac{p(\underline{y}|\underline{\theta})p(\underline{\theta})}{p(\underline{y})}. \tag{2.44}$$

In the Bayesian statistics literature, $p(\underline{\theta})$ is called prior, $p(\underline{y}|\underline{\theta})$ is the likelihood and $p(\underline{y})$ is the evidence [85].

These methods have the advantages that instead of a single estimate of the parameters, the whole probability density function for the parameters is available. The disadvantage is that the prior probability of the parameter has to be chosen somehow. The maximum of the posterior distribution can be found according to

$$\begin{aligned} \underset{\underline{\theta}}{\text{maximize}}\; p(\underline{\theta}|\underline{y}) &= \underset{\underline{\theta}}{\text{maximize}}\; \log\Big(p(\underline{y}|\underline{\theta})p(\underline{\theta})\Big) \\ &= \underset{\underline{\theta}}{\text{maximize}}\; \Big(\log p(\underline{y}|\underline{\theta}) + \log p(\underline{\theta})\Big), \end{aligned} \tag{2.45}$$

considering that $p(\underline{y})$ is a constant. The solution to this problem is referred to as the maximum a posteriori (MAP) estimate. In practice, several choices for prior probabilities of the parameters are possible. The most common ones are Gaussian distributions.

Gaussian Bayesian Models To exemplify the methodology, a linear model is considered. In the Gaussian case, the parameters are usually distributed around zero (if nothing else is known a priori) with a given covariance matrix $\underline{P}_\theta$. Thus, the probability density of the parameters is given as

$$p(\underline{\theta}) = \mathcal{N}(\underline{\theta}|\underline{0}, \underline{P}_\theta). \tag{2.46}$$

If the output is disturbed by white noise, the pdf for the output of the model is

$$p(\underline{y}|\underline{\theta}) = \mathcal{N}(\underline{y}|\underline{X}\,\underline{\theta}, \sigma^2\underline{I}). \tag{2.47}$$

The maximum of the pdf is found as for the prior (2.46) and the conditional probability of the output (2.47) as a solution to the optimization problem

$$\underset{\underline{\theta}}{\text{minimize}} \frac{1}{\sigma^2}\|\underline{y} - \underline{X}\,\underline{\theta}\|_2^2 + \underline{\theta}^T\,\underline{P}_\theta^{-1}\underline{\theta}. \tag{2.48}$$

For $\underline{P}_\theta = \rho\underline{I}$, this reduces to the well known Ridge regularization [16]. The parameter ρ is a hyperparameter to control the variance of the prior distribution.

2.1.6 Gaussian Process Models

Gaussian process models [103] extend the idea of Bayesian models to functions. If it is possible to sample a function, then a Bayesian learning approach for the whole function seems to be possible. Functions are mathematically more complicated than parametric models. This is due to the fact that functions are mappings from a real input space to a real output space. Functions are thus infinite dimensional objects since, for each of the infinitely possible function inputs, a unique output value exists. A parametric representation would thus require infinitely many parameters, one for each possible input value. Handling of this problem is made possible by the introduction of a *kernel* or *covariance* function $k(\cdot,\cdot)$. It is assumed that two output values of the unknown function $f(\underline{u}(1))$ and $f(\underline{u}(2))$ are jointly Gaussian distributed with the covariance $k(\underline{u}(1), \underline{u}(2))$. This avoids an explicit representation of each input and each output as a random variable. It is now possible to obtain a joint probability distribution for the output, given some query output points. The kernel

matrix for query points has to be a valid covariance matrix

$$\underline{K} = \begin{bmatrix} k(\underline{u}(1),\underline{u}(1)) & k(\underline{u}(1),\underline{u}(2)) & \dots & k(\underline{u}(1),\underline{u}(N)) \\ k(\underline{u}(2),\underline{u}(1)) & k(\underline{u}(2),\underline{u}(2)) & \dots & k(\underline{u}(2),\underline{u}(N)) \\ \vdots & \vdots & \ddots & \vdots \\ k(\underline{u}(N),\underline{u}(1)) & k(\underline{u}(N),\underline{u}(2)) & \dots & k(\underline{u}(N),\underline{u}(N)) \end{bmatrix} \tag{2.49}$$

for both an arbitrary number and arbitrary values of input locations $\underline{u}(1),\dots,\underline{u}(N)$. Since $\underline{K}$ should represent a valid covariance matrix, the kernel function has to fulfill several properties. First, it has to be symmetric. Furthermore, every possible matrix $\underline{K}$ that can be constructed has to be positive definite. If this is fulfilled the function $k(\cdot,\cdot)$ has the property to be positive definite. From a standard Gaussian process, values of a function can be sampled at N query locations $\underline{u}$. Therefore, the kernel matrix is built for these locations, and samples are drawn from a Gaussian distribution with exactly this covariance matrix. The numerical implementation of this sampling step works as follows: First, N i.i.d. distributed random numbers $\underline{z}$ are generated. Then, the Cholesky decomposition of $\underline{P} = \underline{K} = \underline{L}^T\underline{L}$ is computed. The sampled values of the output can be obtained as $\underline{y}_s = \underline{L}\,\underline{z}$. The most common kernel is the squared exponential kernel. It is given by the covariance function

$$k(\underline{u}(i),\underline{u}(j)) = \beta \exp\left(\frac{-\|\underline{u}(i)-\underline{u}(j)\|_2^2}{l_G^2}\right). \tag{2.50}$$

This kernel has two parameters the lengthscale l_G and the factor β which controls the regularization strength. Samples from the kernel are shown in Fig. 2.6. The lower the hyperparameter l_G is chosen, the more variation is allowed between neighboring values.

Computation of the Posterior As described in Sect. 2.1.5, Bayesian methods use Bayes formula to derive the posterior probability density function. For Gaussian processes this posterior can be computed for the whole function. It can be shown that the posterior at a query test point $\underline{u}^*$ in the input space is [103]

$$p(y(\underline{u}^*)|\underline{y},\underline{X}) = \mathcal{N}(\hat{m}(\underline{u}^*),\hat{\sigma}_p(\underline{u}^*)), \tag{2.51}$$

with $\underline{X}$ containing all samples of $\underline{u}(k)$,

$$\hat{m}(\underline{u}^*) = \underline{k}(\underline{u}^*,\underline{X})\left(\underline{K}+\sigma^2\underline{I}\right)^{-1}\underline{y}, \tag{2.52}$$

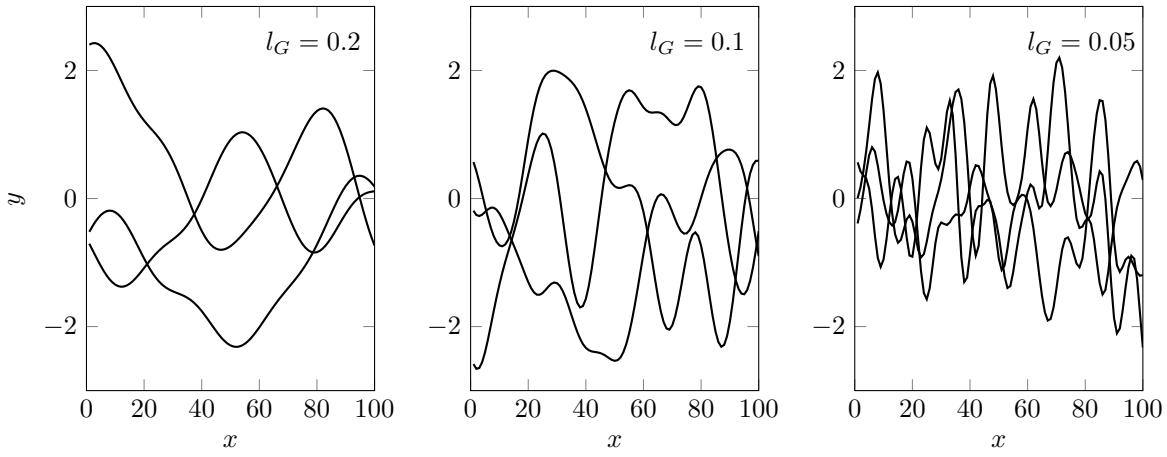

Figure 2.6: Samples from the squared exponential kernel with varying l_G and fixed $\beta = 1$. The left plot shows samples for $l_G = 0.2$, the middle for $l_G = 0.1$ and the right for $l_G = 0.05$.

and

$$\hat{\sigma}_p(\underline{u}^*) = k(\underline{u}^*, \underline{u}^*) - \underline{k}(\underline{u}^*, \underline{X}) \left(\underline{K} + \sigma^2 \underline{I}\right)^{-1} \underline{k}(\underline{X}, \underline{u}^*). \tag{2.53}$$

The prediction of the mean depends on the given output data $\underline{y}$. For the predicted variance $\hat{\sigma}_p(\underline{u}^*)$ this does not hold. The variance does only depend on the distribution of the input data.

Relation of GPs and Kernel Based Methods Another understanding for Gaussian process models can be obtained when kernel methods [109] are considered. This perspective also starts by the mathematical idea that functions are infinite-dimensional objects. Consider two input values, $u(1)$ and $u(2)$, lying in the space of the real numbers $u(1), u(2) \in \mathbb{R}$ and whose corresponding functions values $y(1) = f(u(1))$ and $y(2) = f(u(2))$ are given. A function that describes the behavior between $u(1)$ and $u(2)$ is obviously not unique. In fact, there are uncountable infinitely many possible function values for the infinite number of locations in between $u(1)$ and $u(2)$. To characterize the different functions, one possibility is to assign a number to each function describing its complexity. One possible mapping is called the norm $\|f\|_{\mathcal{H}}$ in the space $\mathcal{H}$. This idea is illustrated in Fig. 2.7. Here, three possible choices of functions are shown. It is not clear a priori which of these functions to choose. In fact the choice is motivated by the assumptions made. The assumption that a kernel-based method makes is that the functions having the minimal norm and fulfill (in the case of noise free measurements) the conditions $f(u(1)) = y(1)$ and $f(u(2)) = y(2)$. This

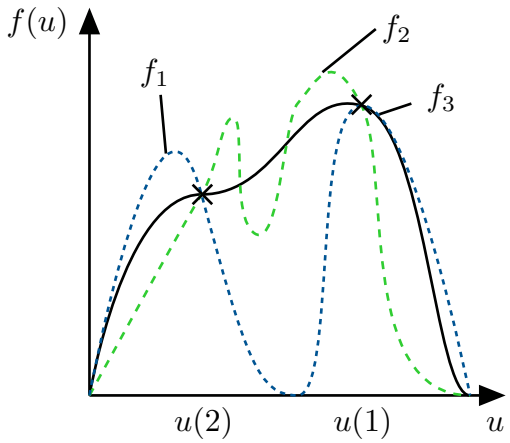

Figure 2.7: Three functions f_1, f_2 and f_3 which fit two points exactly. All functions are elements of the Hilbert space $\mathcal{L}_2$.

can be written as an optimization problem

$$\begin{aligned} &\underset{f\in\mathcal{H}}{\text{minimize}} && \|f\|_{\mathcal{H}} \\ &\text{subject to} && f(u(i)) = y(i),\ i = 1,\dots,N. \end{aligned} \tag{2.54}$$

If the data is allowed to deviate at the points with an error $e(i) = y(i) - f(u(i))$ this problem can be formulated as

$$\underset{f\in\mathcal{H}}{\text{minimize}} \quad \sum_{i=1}^{N} \left(y(i) - f(u(i))\right)^2 + \lambda \|f\|_{\mathcal{H}}^2 \tag{2.55}$$

where λ controls the trade-off between complexity of the function and the first fitting term. The norm of the function is squared to avoid the occurrence of a square root in the optimization problem. This idea seems appealing. Instead of minimizing over real numbers, now the problem is converted into a minimization over functions. One possibility to make this problem solvable is by choice of the space $\mathcal{H}$, from which the functions stem. It is useful to characterize $\mathcal{H}$ as a so-called reproducing Hilbert space. For this space, the points $\underline{u}$ in the input space are mapped to a function each, according to the feature map

$$\phi(\underline{u}) = k(\cdot, \underline{u}). \tag{2.56}$$

Here, to each point, a function is assigned. Usually, the space $\mathcal{H}$ in which these functions lie is called the feature space. This space $\mathcal{H}$ is constructed using the kernel

$k(\cdot,\cdot)$. Therefore, a function which is an element of $\mathcal{H}$ is constructed according to

$$f(\cdot) = \sum_{i=1}^{N} d_i k(\cdot, \underline{u}(i)) \tag{2.57}$$

for arbitrary N, d_i, and $u(i)$. Another function g can then be represented in the same form as

$$g(\cdot) = \sum_{i=1}^{N'} e_i k(\cdot, \underline{u}'(i)). \tag{2.58}$$

for arbitrary N', e_i, and $u(i)'$. In this space a scalar product can be defined according to [103] as

$$< f, g >_{\mathcal{H}} = \sum_{i=1}^{N} \sum_{j=1}^{N'} d_i e_j k(u(i), u'(j)). \tag{2.59}$$

If the scalar product is defined this way, the so-called reproducing property

$$< k(\cdot, \underline{u}''), f >_{\mathcal{H}} = \sum_{i=1}^{n} d_i k(\underline{u}(i), \underline{u}'') = f(\underline{u}'') \tag{2.60}$$

is a special case of (2.59) with $e_j = 1$, $u'(j) = u''$, and $N' = 1$. This is the reason why $\mathcal{H}$ is also called a *reproducing kernel Hilbert space* (RKHS). The norm of a function $\|f\|_{\mathcal{H}}$ can be calculated according to

$$\|f\|_{\mathcal{H}} = \sqrt{< f, f >} = \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N} d_i d_j k(\underline{u}(i), \underline{u}(j))}. \tag{2.61}$$

Now, if the coefficients d_i and the locations $\underline{u}(i)$ are specified, it is possible to calculate the norm of a function. However, the locations $\underline{u}(i)$ can be any point in the space and not necessarily coincide with the points where data has been measured. How is it possible to minimize $\|f\|_{\mathcal{H}}$? The answer to this problem is given by the so-called *representer theorem* [59], which states that the minimizer f^* of (2.54) and (2.55) can be represented as

$$f^*(\cdot) = \sum_{i=1}^{N} d_i k(\cdot, \underline{u}(i)) \tag{2.62}$$

where $\underline{u}(i)$ are the locations of the data points. With this result problem (2.55) can be rewritten as

$$\underset{\underline{d}}{\text{minimize}} \quad \sum_{i=1}^{N} \left(y(i) - \sum_{j=1}^{N} d_i u(j) \right)^2 + \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} d_i d_j k(\underline{u}(i), \underline{u}(j)) \quad . \tag{2.63}$$

Thus, kernel methods with squared error loss lead to the same optimization problem as Gaussian process models.

2.2 Dynamic Models

The main factor which distinguishes dynamic models from the classical problem of statistical learning is the dependence of the process on the time history of the input. In general, a deterministic dynamic system can be described by a nonlinear state space system

$$\underline{x}_s(k+1) = \underline{f}(\underline{x}_s(k), \underline{u}(k)) \tag{2.64}$$

$$y(k) = g(\underline{x}_s(k), \underline{u}(k)). \tag{2.65}$$

Here, the variable $\underline{x}_s(k)$ is the (hidden) state of the system, $\underline{u}(k)$ is the input of the system, the function $\underline{f}(\underline{x}_s(k), u(k))$ is the state update and $g(\underline{x}_s(k), u(k))$ is the output equation.

The class of models which can be described by this type of structure is quite large. Many classical mechatronic models, see [55] for an overview, can be described in this form. However, the model described by (2.65) is deterministic. If disturbances are considered the most general form of a stochastic nonlinear dynamic model is described by two pdfs

$$p(\underline{x}_s(k+1)|\underline{x}_s(k), \underline{u}(k)) \tag{2.66}$$

which is called state update density and

$$p(y(k)|\underline{x}_s(k), \underline{u}(k)) \tag{2.67}$$

called state output density. Now, the transition from one state to the next is stochastic and also the output of the system is affected by noise.

The goal of system identification is to identify models for these systems based on measured input and output data. As described in Sect. 2.1.1 a valid choice is the construction of a maximum likelihood estimator for the prediction of the outputs pdf. To simplify notation of the pdfs, consider a signal $y(k)$ for which the following vector is introduced.

$$\underline{y}_{k_a:k_b} = [y(k_a), y(k_a+1), \ldots, y(k_b)]^T \tag{2.68}$$

with discrete start time k_a and discrete end time k_b. For ML estimation, as for a static systems, the pdf of a model $\hat{p}(\underline{y}|\underline{u},\underline{\theta})$ is considered. The pdf of the output conditioned on the input can be decomposed according to

$$\hat{p}(\underline{y}_{0:N}|\underline{u}_{0:N},\underline{\theta}) = \hat{p}(y(0)|u(0),\underline{\theta})\hat{p}(y(1)|y(0),\underline{u}_{0:1},\underline{\theta})\ldots\hat{p}(y(N)|\underline{y}_{0:N-1},\underline{u}_{0:N},\underline{\theta}). \tag{2.69}$$

In an ML problem, this has to be minimized with respect to the unknown parameters $\underline{\theta}$. To make (2.69) hold, only the definition of conditional probability and the assumption that the system is causal is employed. This holds, since for a causal system it is true that $p(y(k)|\underline{y}_{0:k-1},\underline{u}_{0:N}) = p(y(k)|\underline{y}_{0:k-1},\underline{u}_{0:k})$. The probability distribution $p(y(k)|\underline{y}_{0:k-1},\underline{u}_{0:k})$ is called the one-step predictor, due to the reason that the pdf of the output is conditioned on all past inputs and outputs. Since maximizing the logarithm of (2.69) is the same as minimizing the negative sum of the logarithm of the probabilities of the one-step predictors

$$J = \sum_{k=0}^{N} \log \hat{p}(y(k)|\underline{y}_{0:k-1},\underline{u}_{0:k}). \tag{2.70}$$

methods relying on this decomposition are called prediction-error methods (PEM) [66].

To find the likelihood $p(\underline{y}_{0:N}|\underline{u}_{1:n},\underline{y}_{1:n-1})$ for a nonlinear state-space system is a challenging problem on its own and is called the filtering problem [110]. In general, for the nonlinear state space structure, no analytical solution exists and the filtering solution has to be approximated by an extended Kalman Filter [47] or a particle filter [110] and the states have to be estimated concurrently with the identification procedure. One way to avoid solving the filtering problem is to utilize model structures that allow for an analytical solution for the one-step predictors. In Sect. 2.2.1 linear model structures, e.g. model structures for which $\underline{f}(\underline{x}_s(k))$ and $g(\underline{x}_(k))$ are linear functions, are described.

2.2.1 Identification of Linear Dynamic Models

For linear system identification, a general model structure, which allows for an analytic calculation of the one-step predictions, has been proposed in [67]

$$y(k) = G(q)u(k) + H(q)n(k) \tag{2.71}$$

This structure contains the input $u(k)$ of the system, which is applied to the transfer function $G(q)$. This transfer function calculates a deterministic output signal. A

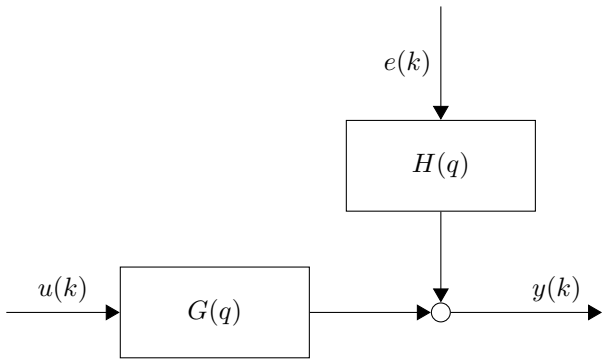

Figure 2.8: General model structure for linear system identification [67].

noise signal $n(k)$ is applied to the transfer function $H(q)$ and is added to the deterministic output for obtaining the measured output $y(k)$ of the system. The operator q is the *shift operator* [48] which, if applied to a signal, increases the time step by 1 so that $qu(k) = u(k+1)$ holds. A graphical representation of the system is shown in Fig. 2.8. The one-step ahead predictor for this system class can be written as [67]

$$\hat{y}(k|k-1) = \frac{G(q)}{H(q)}u(k) + \left(1 - \frac{1}{H(q)}\right)y(k-1) \tag{2.72}$$

and is used in the prediction error method for system identification. We review three basic types, which can be described in this form.

Autoregressive Exogenous (ARX) Models The ARX model considers the same denominator dynamics for noise and the system. For the ARX model $G(q) = \frac{B(q)}{A(q)}$ and $H(q) = \frac{1}{A(q)}$. It is described by

$$y(k) = -a_1 y(k-1) - \ldots - a_n y(k-n) + b_0 u(k) + b_1 u(k-1) + \ldots + b_n u(k-n) \tag{2.73}$$

with the model order n. Its popularity is mainly due to computational reasons, and it has been highlighted that the noise assumption made here is usually unrealistic for many technical systems, which are usually disturbed by white output noise [67]. To solve the identification problem, the parameters of the ARX model are collocated in the $2n+1$ dimensional parameter vector

$$\underline{\theta}^T = \begin{bmatrix} a_1 & a_2 & \ldots & a_n & b_0 & b_2 & \ldots & b_n \end{bmatrix} \tag{2.74}$$

and the $N-n \times 2n+1$ dimensional regressor matrix is constructed according to

$$\underline{X} = \begin{bmatrix} -y(n) & \dots & -y(1) & u(n+1) & \dots & u(1) \\ -y(n+1) & \dots & -y(2) & u(n+2) & \dots & u(2) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -y(N-1) & \dots & -y(N-n) & u(N) & \dots & u(N-n) \end{bmatrix}. \tag{2.75}$$

The output vector used for estimation is

$$\underline{y} = \begin{bmatrix} y(n+1) & \dots & y(N) \end{bmatrix}^T. \tag{2.76}$$

The solution can then be found by the least squares estimator (2.25). For an example of results for ARX identification and the inconsistency problem for simulation models, the reader is referred to Sect. 1.2.

Finite Impulse Response (FIR) Models The inconsistency problem with output noise can be overcome by using FIR models. These models do not consider feedback and $G(q) = B(q)$ and $H(q) = 1$. These models are identified in exactly the same way as ARX models with the difference that the coefficients a_i and the delayed values for $y(k)$ are not used in the parameter vector and the regressor matrix. The problem these models can have is the issue of high parameter variance, as described in Sect.1.2.

Output Error (OE) Models If $G(q) = \frac{B(q)}{A(q)}$ and $H(q) = 1$, then the model is called an output error (OE) model. The predictor for this model is not linear in the parameters anymore, and thus the resulting optimization problem is nonlinear. It is usually solved numerically using second order optimization schemes. For more information, the reader is referred to [90]. The primary obstacle in the application of this type of model is the possibility of local optima and a low robustness in case of order mismatch. This type of model will be compared to novel regularized identification techniques in Sect. 3.5.2. Here, it can be seen that the nonlinear optimization problem possesses local optima.

2.2.2 Regularized Identification of FIR Models

As we have seen, FIR models suffer from a high variance error, especially if the system is excited badly or the SNR is low. To avoid this issue, recently, regularized identification methods for FIR systems have been developed [99, 98]. In the survey [100], the main contributions have been summarized. The idea is to reduce the

variance error by adding a regularization term to the optimization objective. If a quadratic penalty is chosen, the regularized FIR identification problem becomes

$$\underset{\underline{\theta}}{\text{minimize}} \left\| \underline{y} - \underline{X}\,\underline{\theta} \right\|_2^2 + \lambda \underline{\theta}^T \underline{R}\,\underline{\theta} \tag{2.77}$$

with λ determining the strength and the semidefinite matrix $\underline{R}$ determining the form of the penalty. The matrix $\underline{R}$ has to be semidefinite since otherwise arbitrarily negative values can obtained for the loss by choosing $\underline{\theta}$. If $\underline{R}$ is positive definite (2.77) can also be written with $\underline{R} = \underline{P}^{-1}$. Here $\underline{P}$ is another positive definite matrix, which can be interpreted as a covariance matrix. Usually, the matrix $\underline{P}$ is restricted to some subspace of the positive definite matrices. The simplest example is a diagonal matrix. The resulting identification problem is then equivalent to ridge regression [54]. In [97] a specific parameterization of $\underline{P}$ as well as the marginal likelihood algorithm for hyperparameter tuning is proposed. The solution to the problem can be found as [28]

$$\hat{\underline{\theta}} = \left(\underline{X}^T \underline{X} + \lambda \underline{R} \right)^{-1} \underline{X}^T \underline{y}. \tag{2.78}$$

The solution can be computed efficiently using a QR factorization as described in Sect. 2.1.2 for linear regression [26].

The Bayesian Framework Deeper understanding of the role of $\underline{P}$, can be obtained by a Bayesian perspective [28]. The output of an FIR system can be written as

$$\underline{y} = \underline{X}\,\underline{\theta} + \underline{n}. \tag{2.79}$$

The vector $\underline{n}$ contains Gaussian noise with i.i.d. entries and covariance σ^2. As described in Sect. 2.1.5 in a Bayesian framework, prior probabilities are assigned to the parameters. In contrast to the frequentist approach, the Bayesian approach assumes prior knowledge for the parameters $\underline{\theta}$. One could say that the Bayesian approach favors some parameters in advance, while the frequentist approach considers every possible parameter as equally probable. In the case of the regularized FIR identification method, the prior distribution of the parameters is assumed to be Gaussian with

$$p(\underline{\theta}) = \mathcal{N}(\underline{\theta} | \underline{0}, \rho \underline{P}) = \mathcal{N}(\underline{\theta} | \underline{0}, \underline{P}_{\underline{\theta}}). \tag{2.80}$$

The parameter ρ is used to scale the prior distribution. It is not equal to λ in (2.77), since the strength of the regularization in the Bayesian case is influenced by both the variance of the noise and the scaling factor of the parameter covariance. To simplify the notation, we will denote the scaled covariance matrix of the parameters

by $P_{\underline{\theta}} = \rho \underline{P}$ and the unscaled version by $\underline{P}$.

Taking this into account, the joint probability distribution of $\underline{y}$ and $\underline{\theta}$ can be calculated by [28]

$$p\left(\begin{bmatrix}\underline{\theta}\\ \underline{y}\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\underline{\theta}\\ \underline{y}\end{bmatrix} \middle| \underline{0}, \begin{bmatrix}\underline{P_\theta} & \underline{P_\theta}\,\underline{X}^T \\ \underline{X}\,\underline{P_\theta} & \underline{X}\,\underline{P_\theta}\,\underline{X}^T + \sigma^2 \underline{I}_N\end{bmatrix}\right). \tag{2.81}$$

The posterior distribution of the parameters conditioned on the measured output $\underline{y}$ can be calculated as

$$p(\underline{\theta}|\underline{y}) = \mathcal{N}\left(\underline{\theta}|\hat{\underline{\theta}}, \hat{\underline{\Sigma}}_{\underline{\theta}}\right) \tag{2.82}$$

with

$$\hat{\underline{\theta}} = \left(\underline{P_\theta}\,\underline{X}^T\underline{X} + \sigma^2 \underline{I}_{n+1}\right)^{-1} \underline{P_\theta}\,\underline{X}^T\underline{y} \tag{2.83}$$

and

$$\hat{\underline{\Sigma}}_{\underline{\theta}} = \underline{P_\theta} - \underline{P_\theta}\,\underline{X}^T\left(\underline{X}\,\underline{P_\theta}\,\underline{X}^T + \sigma^2 \underline{I}_N\right)^{-1} \underline{X}\,\underline{P_\theta}. \tag{2.84}$$

This result can be derived by using the analytical formula of conditionals for Gaussian distributions, see [95].

The estimation of the posterior distribution for the mean of the parameters can also be written as

$$\hat{\underline{\theta}} = \left(\underline{X}^T\underline{X} + \frac{\sigma^2}{\rho}\underline{P}^{-1}\right)^{-1} \underline{X}^T\underline{y} \tag{2.85}$$

This is the same solution as (2.78) with $\lambda = \frac{\sigma^2}{\rho}$ and $\underline{R} = \underline{P}^{-1}$.

The computation of the variance $\underline{\Sigma}_{\underline{\theta}}$ is computational demanding since (2.84) contains the inverse of a matrix which is $N \times N$. To simplify the relation, a matrix inversion lemma can be employed

$$\hat{\underline{\Sigma}}_{\underline{\theta}} = \underline{P_\theta} - \underline{P_\theta}\,\underline{X}^T\left(\frac{\underline{I}_N}{\sigma^2} - \frac{\underline{I}_N}{\sigma^2}\underline{X}\left(\underline{P_\theta}^{-1} + \underline{X}^T\frac{\underline{I}_N}{\sigma^2}\underline{X}\right)^{-1}\underline{X}^T\frac{\underline{I}_N}{\sigma^2}\right)\underline{X}\,\underline{P_\theta} \tag{2.86}$$

$$= \underline{P_\theta} - \underline{P_\theta}\frac{\underline{X}^T\underline{X}}{\sigma^2}\underline{P_\theta} - \underline{P_\theta}\frac{\underline{X}^T\underline{X}}{\sigma^2}\left(\underline{I}_{n+1} + \underline{P_\theta}\frac{\underline{X}^T\underline{X}}{\sigma^2}\right)^{-1}\underline{P_\theta}\frac{\underline{X}^T\underline{X}}{\sigma^2}\underline{P_\theta}. \tag{2.87}$$

This reduces the computational demand for the computation of $\hat{\underline{\Sigma}}_{\underline{\theta}}$ since it only contains inverses of the size $n+1 \times n+1$. The Bayesian approach has, compared to the regularized approach an additional hyperparameter σ^2 which is the variance of

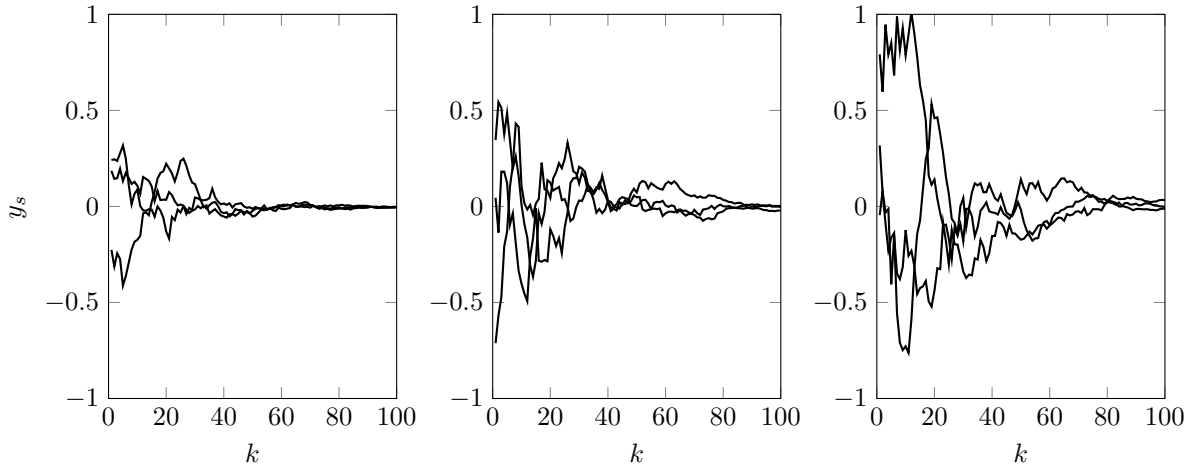

Figure 2.9: Role of the hyperparameter ρ for samples from the covariance matrix. The left responses are sampled from $\rho = 10$, the middle from $\rho = 1$ and the right from $\rho = 0.1$, $\alpha = 0.92$ is held fixed. The lower ρ is chosen, the further the samples deviate from zero.

the noise. Both scaling of the covariance matrix ρ and σ^2 influence the regularization strength λ. The same holds for the Bayesian interpretation of Gaussian process models.

Choice of the Prior Knowledge Several choices for the prior knowledge incorporated by the matrix $\underline{P}$ are possible. In the initial paper [99], the so-called stable-spline kernel is introduced, leading to the covariance matrix

$$P_{ij} = \alpha^{\max(i,j)} \tag{2.88}$$

in the first order case. This kernel has one hyperparameter α. Samples of parameters drawn from the prior distribution, corresponding to this kernel, are shown in Fig. 2.9 for different choices of ρ and in Fig. 2.10 for different choices of α. It is clearly visible that ρ determines the deviation of the coefficients of the impulse response from zero and α influences the decay.

The identification of FIR models can significantly benefit from this approach in practice. Recently, for a polymer reactor, the application of this type of regularization allowed for a remarkably better identification result. The behavior went from a very wiggly to an interpretable behavior of the impulse responses [78].

There are several other options for the design of the covariance matrix. An approach to include several system theoretical properties in the prior knowledge, like non-oscillatory behavior or relative degree, has been described by [36]. An alternative

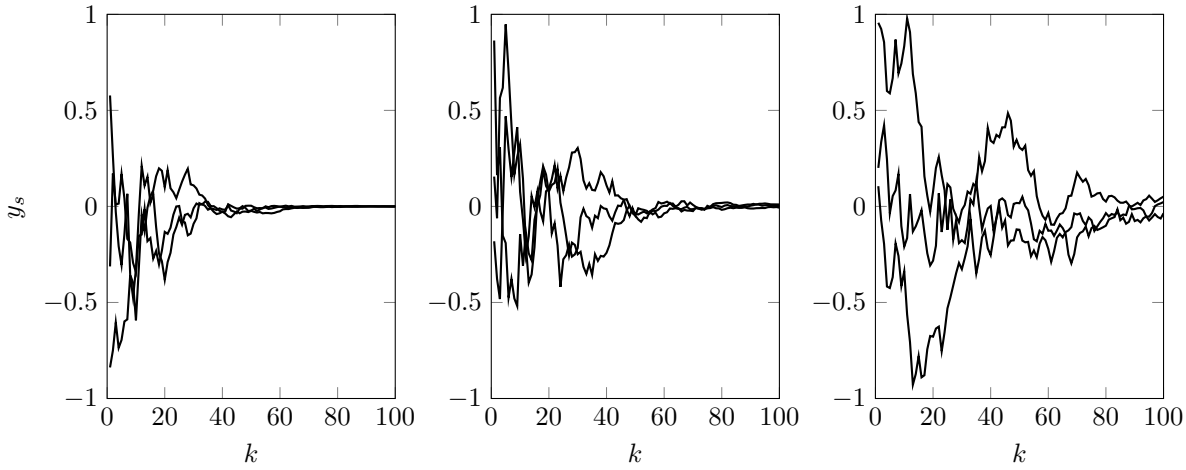

Figure 2.10: Role of the hyperparameter α for samples from the covariance matrix. The left responses are sampled from $\alpha = 0.85$, the middle from $\alpha = 0.9$ and the right from $\alpha = 0.95$, $\rho = 1$ is held fixed. The parameter α determines the decay of the impulse responses.

is based on orthonormal basis function (OBF) kernels [33, 27]. Another approach has been presented in [23]. Here, a linear state space system is excited from random initial conditions to obtain the desired kernel. In [8], an approach is described which trains a neural network, which constructs the kernel matrix for the regularization based on the whole input/output data of the system. From this point of view, the choice of the kernel seems to be vague and somewhat arbitrary. The primary justification for the usage is the performance of the kernels on stochastic benchmarks. However, it is essential to choose these benchmarks carefully, since otherwise, the class of kernels selected on these benchmarks perform bad in the real world [105]. An attractive property, which the TC kernel has in contrast to other kernels, is the maximum entropy property [25, 21]. In some sense, this property can be interpreted as incorporation of the least prior knowledge possible when the covariance of two values following one another should decay exponentially.

One of the major goals of this thesis is to put the design of the kernel back to a systems viewpoint. Therefore, in Chap. 3 impulse response preserving kernels will be introduced, which allow for a better interpretation of the prior knowledge induced.

Extensions There are some notable extensions to the FIR case. In [101], an extension has been proposed which also regularizes the coefficients of the denominator coefficients of an ARX model. In [31] the method is extended to dynamic linear network identification. In [14] a generalization for spectrum identification is given.

Alternative Approaches Beside the usage of the quadratic regularization penalty, there are other approaches which can be applied for regularization. An alternative approach has been described in [125]. Here, the idea is to put a prior on the Hankel singular values of the transfer function. This prior is compared to the Gaussian prior not analytically tractable and thus advanced Bayesian techniques (like Markov chain Monte Carlo) have to be used, which are computationally demanding. Another approach is the usage of the 1-norm instead of the 2-norm of the parameters [104]. This encourages sparsity in the impulse response. A significant drawback of this approach is that this prior knowledge is often not well suited for impulse responses of dynamical systems which are exponentially decaying and thus not sparse in the coefficients. In [106], a sparse method which also considers the outputs of the system is described. Another line of research follows identification of linear networks [31, 135]. The idea here is to impose two regularizers. One, with 2-norm, to regularize the impulse response and another 1-norm regularizer to keep the interaction within the dynamic network sparse.

2.2.3 Prediction Error Methods for Nonlinear Systems

The idea of PEM for linear systems has been described in the previous section. In principle, PEM works for nonlinear systems in the same way [66]. However, finding the optimal predictor for e.g. nonlinear state-space systems is a challenging problem on their own. Several methods combining filtering and optimization for the model parameters exist [110, 64, 45, 44, 122]. These methods increase the computational demand needed for the filtering step significantly. In this contribution, we focus on descriptions allowing a simple computation of the predictor.

NARX system identification As in the linear case, the NARX structure allows for the computation of a simple one-step predictor. This commonly applied model [22] is described by

$$\hat{y}(k) = f(u(k), \ldots, u(k-n), y(k-1), \ldots, y(k-n)). \tag{2.89}$$

If the model structure holds in the real world, it has been shown that when f is chosen as a GP as the number of observations tends to infinity, the function is identified consistently [35]. In [61], identification of NARX models with GPs is described in detail. However, as for the linear case described in Sect. 2.2.1, the noise assumptions made are not valid for many real world systems. This has the consequence that the estimate is biased. In the worst case, it can happen that, due to the inconsistency

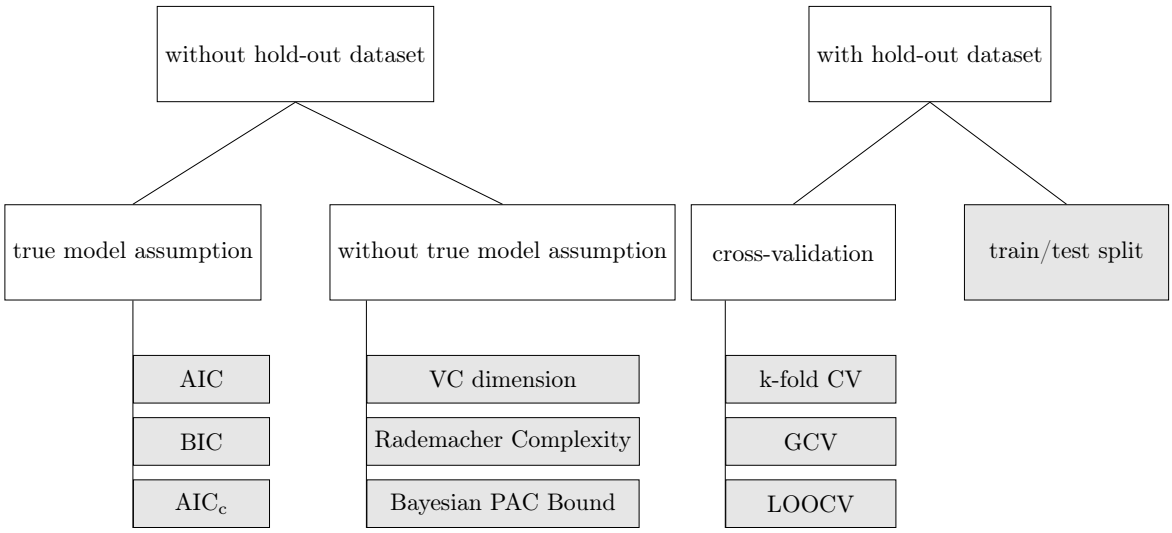

Figure 2.11: Overview of available model selection techniques.

of the NARX method, an unstable model is identified, although the true system is stable.

NOE system identification Nonlinear output error (NOE) identification methods mitigate the inconsistency issue. This idea can be applied for linear system identification as well. There, it is called an output error (OE) model. The significant advantage of this approach is that if the correct order predictor for a simulation model is trained, then the obtained solutions share the advantageous properties of an MLE discussed in Sect. 4.2.2. The significant drawback is that it is possible that the obtained solution is a local minimum. Some results exist for the construction of appropriate input signals to avoid this issue [37]. For many real world systems, however, the input cannot be chosen by the user.

2.3 Complexity Selection

Usually, the parameters of a model are estimated based on data. If the same data is used again to evaluate the performance of the model, then the estimated performance on *these* data will be better than on independently generated test data. The more complex the model is, the higher will be the difference between these two errors. The ability of a model to perform well on independent test data is termed *generalization* ability of the model. The fact that the model is better on training than on test data is called *overfitting*, and the process of selecting a model that performs best on test data is called *complexity selection*. Several techniques for complexity selection have been developed. Several models are fit on data and, subsequently, one of these

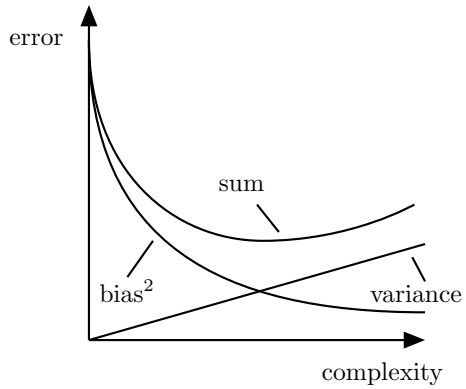

Figure 2.12: Schematic illustration of the bias-variance dilemma. If the model complexity is increased the bias of a model is reduced but the variance of the estimate becomes worse.

multiple models is selected. The taxonomy of different model selection techniques is shown in Fig. 2.11. One way, shown at the right side of the figure, is to keep a hold-out dataset aside to estimate the generalization ability of the model. The classical train/test split technique falls under this category. Here, the training data is split once. The model is trained on the training data. The performance of the model is evaluated on test data kept aside during the training process.

General cross-validation methods are another class of approaches. Here, data is split in several parts, e.g., 5 folds. Subsequently, on a fraction of each of these parts, training is performed and the performance is evaluated on the part of the data left aside. This procedure is repeated until each part has been left aside once. The sum of the loss function of all repetitions is a measure for the performance on test data. Methods falling under this category are discussed in Sect. 2.3.1.

Other approaches circumvent the need for a hold-out dataset. These can be further divided depending on whether the criterion assumes that the true system is contained in the model set or that it is not. Criteria assuming the true model structure to be contained are AIC [5], AIC_c [121], and BIC [116]. Techniques which rely on probabilistic bounds include VC dimension [127], Rademacher complexity [10] and PAC-Bayesian bounds [75] and work without this assumption. Although many attempts have been made, the actual state of science is far from a complete understanding of the generalization phenomena. Recently, it has been discovered in [134], that state-of-the-art neural networks can remember the labels for white-noise generated input images. Classical approaches are not able to explain the performance of these interpolating methods.

Bias-Variance Trade-off In the following, a model with additive noise at the process output is considered. The measured output of the model is described by

$$y(\underline{u}(k)) = y_u(\underline{u}(k)) + n(k) \tag{2.90}$$

with $n(k)$ denoting i.i.d. noise with mean zero and variance σ^2 and $y_u(\underline{u}(k))$ is the undisturbed output of the system. If, for this system, a model is fit and the expected value of the error is analyzed, then three terms occur [90, 43]

$$\begin{aligned}\mathbb{E}_{y\sim p(y),\hat{y}\sim p(\hat{y})}(y(\underline{u}) - \hat{y}(\underline{u}))^2 = &\underbrace{\left[y_u(\underline{u}) - \mathbb{E}_{\hat{y}\sim p(\hat{y})}(\hat{y}(\underline{u}))\right]^2}_{\text{bias}^2} \\ &+ \underbrace{\mathbb{E}_{\hat{y}\sim p(\hat{y})}\left[\hat{y}(\underline{u}) - \mathbb{E}_{\hat{y}\sim p(\hat{y})}\hat{y}(\underline{u})\right]^2}_{\text{variance}} + \underbrace{\sigma^2}_{\text{irreducible error}} .\end{aligned} \tag{2.91}$$

The first term is the bias which describes the error component of the model due to the error between the modeled and the true function. The derivation of the bias-variance decomposition is done at a specific point $\underline{u}$ within the input space. This, however, makes this quantity local. The bias error happens just at one prespecified location and there can and will be a different bias error at every location of the estimated function. A simple method to quantify the global bias error is to calculate the mean of the bias error at all sampled locations. In many practical relevant cases, this will be useful. However, when in a dataset e.g., many low values occur, the bias error in these regions will be emphasized. In Fig. 2.12 it is illustrated that the bias error decreases with increasing complexity of the model.

The second error term is the variance error. This error describes the effect of noise contained in the data on the parameter estimate. In contrast to the bias error, the variance error increases with increasing model complexity, as shown in Fig. 2.12. It is, as the bias, a local quantity depending on the location $\underline{u}$. Thus, the variance error can also be different at different locations within the input space.

The third term is noise variance, the irreducible error. This part is entirely random and cannot be reduced, whatever identification method is employed. In consequence, this means that the choice of an appropriate model structure also means to choose a suitable bias-variance trade-off. This is illustrated by the U-shaped curve, shown in Fig. 2.12 which is the sum of bias and variance error. The optimal complexity of a model lies at the point where the sum of these two terms is minimal. Also, for Bayesian methods in system identification, the appropriate bias-variance trade-off matters. Here, it is controlled by an appropriate amount of regularization [72].

Train/Test Split A simple technique for this type of task is the decomposition of the data in training, validation, and test data. The model is trained on the training data, and model selection is performed on validation data. Finally, model assessment is conducted on the test data. This type of technique is statistically quite sound. It provides estimates for the right statistical quantities for the model selection and the model assessment task. Its most significant drawback, however, is the necessity to leave out data that must not be available for the training phase. This is not a big deal if a large amount of data is available. In an industrial context, it can be a substantial cost driver. This is a significant drawback of this approach for real applications [12].

2.3.1 LOOCV and GCV

Another alternative is to use the leave-one-out cross-validation (LOOCV). Here, one data point is left out and training is performed on the other $N - 1$ points. This procedure is repeated N times and the LOOCV error is the mean error on the data points left out. It is often argued that the variance of the LOOCV error is high since only one point at a time is changed [43]. Thus, it is advantageous to leave out more than one point. A common procedure is to split the data into 5 equally sized parts and estimate 5 models with each of the parts left aside once. Then the loss is evaluated for each model on these left aside parts. This is called k-fold (or in this case 5-fold) cross-validation.

In [12], though, it has been shown that the estimation of this variance for k-fold cross-validation is rather problematic. Especially, it is not clear whether LOOCV has higher or lower variance than k-fold cross-validation. Thus, this argument is at least not valid in general.

Despite possible problems with the variance of the estimate, the practical applicability of LOOCV is limited in general since a brute-force approach of LOOCV would require the estimation of N models which is computationally demanding. Linear models make an exception to this fact. For linear regression the leave-one out cross-validation error J_L can be computed according to [43]

$$J_L = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{y(i) - \hat{y}(i)}{1 - S_{ii}} \right)^2 \tag{2.92}$$

with S_{ii} denoting the i-th diagonal element of the smoothing matrix

$$\underline{S} = \underline{X}\left(\underline{X}^T\underline{X}\right)^{-1}\underline{X}^T. \tag{2.93}$$

This quantity is not invariant to rotations of the measurement coordinate system [50]. In contrast the generalized cross-validation (GCV) criterion

$$J_G = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y(i)-\hat{y}(i)}{1-\frac{\mathrm{tr}(\underline{S})}{N}}\right)^2. \tag{2.94}$$

fulfills this property. Instead of the diagonal entries which weight each error only the trace of the smoothing matrix has to be calculated. This technique has been derived for the estimation of a good ridge parameter [50]. It is also possible to apply this technique for Bayesian system identification. In this case, the smoothing matrix is calculated according to

$$\underline{S} = \underline{X}\left(\underline{X}^T\underline{X} + \lambda\underline{R}\right)^{-1}\underline{X}^T. \tag{2.95}$$

If (2.94) is optimized with respect to hyperparameters (like λ or parameters of $\underline{R}$), the hyperparameters are identified consistently [76].

2.3.2 Information Criteria

Besides cross-validation, another option to deal with the bias-variance dilemma are information criteria. These criteria penalize the number of parameters by adding a penalty term to the log-likelihood [119]. The loss function then reads as

$$J = \log\hat{p}(\underline{y},\underline{\theta}) + \chi(n,N)n \tag{2.96}$$

with N denoting the number of samples and n denoting the number of parameters. The function $\chi(n,N)$ is chosen according to the used information criterion. An overview of popular choices is given in Tab. 2.1.

To deal with the bias-variance trade-off, an information criterion has been derived by Akaike in a series of papers [4, 6, 5]. The idea of the AIC is to consider the loss term

$$J_{\mathrm{AIC}} = \mathbb{E}_{\underline{y}_v\sim p(\underline{y})}\mathbb{E}_{\underline{y}_t\sim p(\underline{y})}\log\hat{p}(\underline{y}_v|\hat{\underline{\theta}}(\underline{y}_t)). \tag{2.97}$$

Name (long)	Name (short)	$\chi(n, N)$
Akaike's Information Criterion	AIC	2
Corrected Information Criterion	AIC_c	$2\ \frac{N}{N-n-1}$
Bayesian Information Criterion	BIC	$\ln N$

Table 2.1: Values of the penalty function $\chi(n, N)$ depending on the chosen information criterion [119]

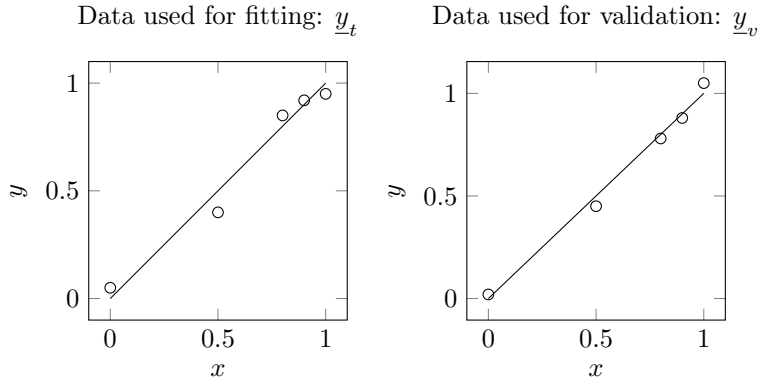

Figure 2.13: Illustration of different expected values used for computation of J_{AIC}. The left side shows a sample from the training data $\underline{y}_t$ employed to estimate the parameters $\hat{\underline{\theta}}(\underline{y}_t)$ and the right side shows a sample from the validation data employed to estimate the performance of the model.

Here, it is important to distinguish between two probability distributions. The first is the true underlying data generating distribution $p(\underline{y})$ and the second $\hat{p}(\underline{y}_v|\hat{\underline{\theta}}(\underline{y}_t)))$ is the model of the probability distribution. The modeled probability distribution depends on the ML estimate of the parameters $\hat{\underline{\theta}}(\underline{y}_t)$ which themselves depend on the vector of training data $\underline{y}_t$. The likelihood is evaluated on independently drawn validation data $\underline{y}_v$. This expectation $\mathbb{E}_{\underline{y}_v \sim p(\underline{y})}\mathbb{E}_{\underline{y}_t \sim p(\underline{y})}$ means that validation data $\underline{y}_v$ and training data $\underline{y}_t$ are sampled independently from the same probability distribution. This is illustrated in Fig. 2.13. The left plot shows a sample from the training data $\underline{y}_t$ and the right plot shows a sample from the validation data $\underline{y}_v$. In other words, (2.97) means that the AIC estimates the expected valued of the likelihood on independently generated validation data. Another important fact is that the locations of the data within the input space remains the same for $\underline{y}_t$ and $\underline{y}_v$. A fact, commonly ignored about AIC, is the distribution of the input data. One common assumption is that input data is generated from a distribution depending on the input space $p(\underline{x})$. The error for the estimation of AIC is then only estimated *in sample* [43]. This means that for the estimation only a resampling of the points at exactly the same locations as in the training data is considered.

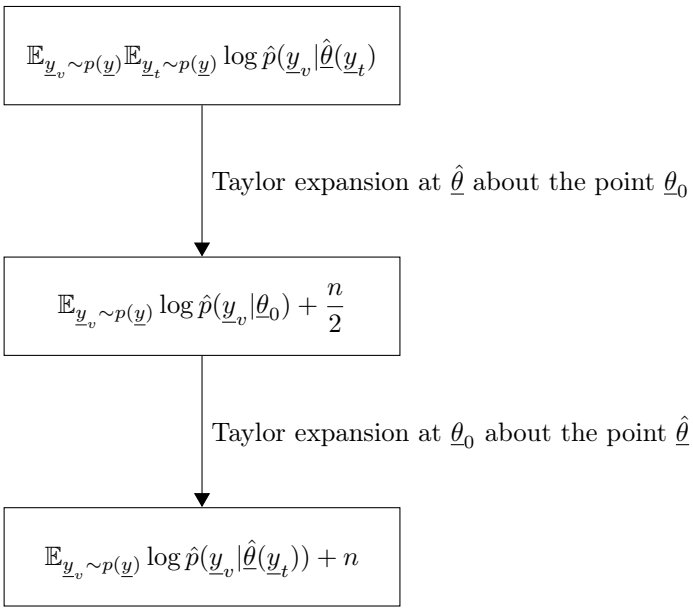

Figure 2.14: Route for the derivation of the AIC. The derivation requires two Taylor expansions of the log likelihood.

The goal of the AIC is to get rid of the expectation term depending on the training data $\mathbb{E}_{\underline{y}_t \sim p(\underline{y})}$ [19]. To enable this, two Taylor expansions are needed. This process is illustrated in Fig. 2.14. The first steps allows to eliminate the dependence on the training data, but the result depends on the (unknown) true parameters $\underline{\theta}_0$. The second Taylor expansion removes this dependence. These two steps for the derivation are examined in detail.

The first Taylor expansion The second order Taylor approximation of $p(\underline{y}_v, \hat{\underline{\theta}})$ is formed according to

$$\begin{aligned} \log \hat{p}(\underline{y}_v | \hat{\underline{\theta}}(\underline{y}_t)) \approx \log \hat{p}(\underline{y}_v, \underline{\theta}_0) + \left[\left. \frac{\partial \log \hat{p}(\underline{y}_v | \underline{\theta})}{\partial \underline{\theta}} \right|_{\underline{\theta}=\underline{\theta}_0} \right]^T \left[\hat{\underline{\theta}} - \underline{\theta}_0 \right] \\ + \frac{1}{2} \left[\hat{\underline{\theta}} - \underline{\theta}_0 \right]^T \left. \frac{\partial^2 \log \hat{p}(\underline{y}_v, \underline{\theta})}{\partial^2 \underline{\theta}} \right|_{\underline{\theta}=\underline{\theta}_0} \left[\hat{\underline{\theta}} - \underline{\theta}_0 \right]^T . \end{aligned} \tag{2.98}$$

Now, the expectation with respect to the validation data is applied to the three terms. The first term depends only on $\underline{\theta}_0$ and is thus not influenced by the training data. The second term is the gradient of the log-likelihood function. The parameter $\hat{\underline{\theta}}$ maximizes the log-likelihood function. So

$$\mathbb{E}_{\underline{y}_v \sim p(\underline{y})} \left[\left. \frac{\partial \log p(\underline{y}_v | \underline{\theta})}{\partial \underline{\theta}} \right|_{\underline{\theta}=\underline{\theta}_0} \right]^T \left[\hat{\underline{\theta}} - \underline{\theta}_0 \right] = 0 \tag{2.99}$$

The third term requires some analysis. First, it is noted that the third term can be written as

$$\begin{aligned}&\frac{1}{2}\left[\hat{\underline{\theta}}-\underline{\theta}_0\right]^T\left.\frac{\partial^2\log\hat{p}(\underline{y}_v,\underline{\theta})}{\partial^2\underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}_0}\left[\hat{\underline{\theta}}-\underline{\theta}_0\right]^T\\ =&\frac{1}{2}\mathrm{tr}\left(\left.\frac{\partial^2\log\hat{p}(\underline{y}_v,\underline{\theta})}{\partial^2\underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}_0}\left[\hat{\underline{\theta}}-\underline{\theta}_0\right]\left[\hat{\underline{\theta}}-\underline{\theta}_0\right]^T\right)\end{aligned}\tag{2.100}$$

The expected value of the term $\left.\frac{\partial^2\log\hat{p}(y_v,\underline{\theta})}{\partial^2\underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}_0}$ is the Fisher information matrix. The term $\left[\hat{\underline{\theta}}-\underline{\theta}_0\right]\left[\hat{\underline{\theta}}-\underline{\theta}_0\right]^T$ is the covariance of the parameters which is the inverse Fisher information matrix [19]. If the Fisher information matrix has full rank, e.g. the system is identifiable, the trace of a matrix times its inverse will be equal to the number of the parameters. For the quantity J_{AIC}, this leads to

$$J_{\mathrm{AIC}}\approx\mathbb{E}_{\underline{y}_v\sim p(\underline{y})}\log p(\underline{y}_v|\underline{\theta}_0)+\frac{1}{2}n\tag{2.101}$$

Now, the quantity depends on the true parameters $\underline{\theta}_0$. Thus, the loss term is not influenced by the effect the training data has on the parameter estimate. Unfortunately the true value of the parameters $\underline{\theta}_0$ is unknown.

The second Taylor expansion To circumvent this issue, the same procedure as before is applied again and it is found that

$$\log\hat{p}(\underline{y}_v|\underline{\theta}_0))=\log\hat{p}(\underline{y}_v,\hat{\underline{\theta}})+\left[\left.\frac{\partial\log\hat{p}(y_v|\underline{\theta})}{\partial\underline{\theta}}\right|_{\underline{\theta}=\hat{\underline{\theta}}}\right]^T\left[\underline{\theta}_0-\hat{\underline{\theta}}\right]\tag{2.102}$$

$$+\frac{1}{2}\left[\underline{\theta}_0-\hat{\underline{\theta}}\right]^T\left.\frac{\partial^2\log\hat{p}(\underline{y}_v,\underline{\theta})}{\partial^2\underline{\theta}}\right|_{\underline{\theta}=\hat{\underline{\theta}}}\left[\underline{\theta}_0-\hat{\underline{\theta}}\right]^T.\tag{2.103}$$

Here, with the same arguments as above, it can be shown that [19]

$$\mathbb{E}_{\underline{y}_v\sim p(\underline{y})}\log\hat{p}(\underline{y}_v|\underline{\theta}_0)\approx\mathbb{E}_{\underline{y}_v\sim p(\underline{y})}\log p(\underline{y}_v|\hat{\underline{\theta}})+\frac{1}{2}n\tag{2.104}$$

so in total for J_{AIC} the quantity

$$J_{\mathrm{AIC}}\approx\mathbb{E}_{\underline{y}_v\sim p(\underline{y})}\log p(\underline{y}_v|\hat{\underline{\theta}})+n\tag{2.105}$$

is found. If all terms are multiplied by two, this is the classical AIC criterion of Akaike [4]. In the orignal paper, multiplication by two has been done for simplifi-

cation. If a Gaussian noise model is assumed, multiplication by two has the effect that the term $\frac{1}{2}$ occuring in the exponent of the Gaussian pdf is canceled in the first term of the AIC.

To summarize the results, three steps are applied to derive the AIC. The first is to approximate the likelihood with a quadratic expansion. Secondly, the expectation with respect to the training data is calculated. Thirdly, again a quadratic approximation is applied to eliminate the dependence on the true parameters.

The AIC is derived for $N \to \infty$. For low ratios of $\frac{N}{n}$ the estimate is biased. To deal with this bias the corrected AIC or $\mathrm{AIC_c}$ [121]

$$\mathrm{AIC_c} = \log \hat{p}(\underline{y}, \hat{\underline{\theta}}) + 2\frac{N}{N-n-1}n \tag{2.106}$$

has been proposed. The term $\frac{N}{N-n-1}$ tends to 1 if $N >> n$ and increases the less N and n deviate.

In many practical scenarios with local model networks, the $\mathrm{AIC_c}$ has been observed to work quite well, even when it is the case that some assumptions for its derivation do not hold. The reason is that the curve of the training error for the local model networks is monotonically decreasing with an elbow-like shape. At the same time, the number of parameters increases linearly. It is often not too crucial, whether 20 or 25 models are estimated so that many of the information criteria work well in practice [52].

2.3.3 Statistical Learning Theory and VC-Dimension

Information criteria are important since they allow for easy handling of the penalty term and often yield good results, especially for local model networks, see e.g. [52], for an application to LOLIMOT and HILOMOT. There are two drawbacks, however. The first is that the true model is assumed to lie in the model class parameterized by $\underline{\theta}$. The other is that only the in-sample error is investigated. The locations in the input space the model is evaluated at is kept fixed. These drawbacks can be circumvented considering the second possible interpretation of regression described in Sect. 2.1. These methods find a probabilistic inequality, setting training and test error in relation [127]. Therefore the risk $R(\underline{\theta})$, described by (2.2), is considered.

It is essential to notice that $R(\underline{\theta})$ is the expectation with respect to both the input and the output, in the joint distribution $p(\underline{x}, y)$. This means that the risk calculated

depends on the distribution of the input and output data. This is crucial for all bounds derived. For a nonlinear dynamic system, this means that if the input signals for training and testing are different regarding their distribution in the input space, then the value of the risk will change significantly. All the bounds derived by this theory are only valid if the joint distribution of input and output is the same for training and test data.

Now, the bounds derived assume that $\hat{y}$ lies within a class of functions $\mathcal{H}$. For example, $\mathcal{H}$ can be every linear function or a neural network. In reality only samples from $p(\underline{y}, \underline{x})$ are available. Thus the function $\hat{y}$ is chosen which minimizes the empirical risk

$$R_S(\underline{\theta}) = \frac{1}{N} \sum_{k=1}^{N} \left(y(k) - \hat{y}(\underline{x}(k), \underline{\theta}) \right)^2 . \tag{2.107}$$

The empirical risk is obtained when the available samples of $p(\underline{x}, y)$ are used to approximate the expectation within the definition of $R(\underline{\theta})$. The goal of learning theory is to provide bounds for the difference between the risk R_S given by (2.107) and R given by (2.2).

Since everything involved in this problem is random, the difference between these two risks is also random. Thus any inequality will only hold with some probability of $1-\delta$. These bounds depend on a measure of the complexity of the set of functions $\mathcal{H}$ from which a function minimizing the risk $R_S(\underline{\theta})$ can be selected. The most common measure of complexity is the VC dimension of $\mathcal{H}$ [127]. The VC dimension is a measure of size for the, possibly infinite, set of hypotheses $\mathcal{H}$. It has been originally developed for the characterization of sets of indicator functions but can be extended to sets of real-valued functions, too [127]. The most important case of functions for which an explicit formula for the VC dimension can be provided are functions linear in their parameters. For this function class the VC dimension is $n_{VC} = n + 1$. The number of parameters is thus (up to the summand 1) equal to the VC dimension. It is important to notice that this does not necessarily hold for the nonlinear case, see [126] for some illustrative examples. If the VC dimension of an estimator is known, then usually bounds of the following type are proposed [127]

$$R(\underline{\theta}) \leq R_S(\underline{\theta})(1 - \sqrt{\chi})^{-1}_{\infty}, \tag{2.108}$$

where

$$x^{-1}_{\infty} = \begin{cases} \frac{1}{x} & x > 0 \\ \infty & x \leq 0 \end{cases} \tag{2.109}$$

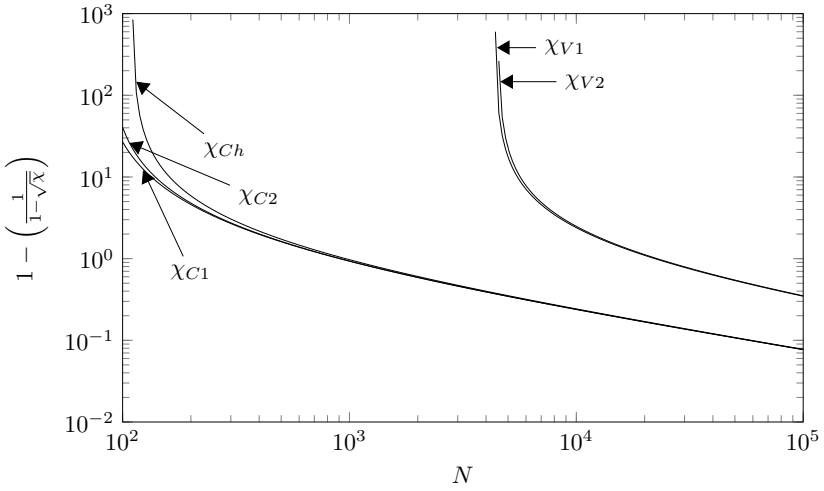

Figure 2.15: Different VC bounds for regression depending on the number of samples. The VC dimension is fixed to 60. For χ_{C1} and χ_{V1} the certainty is $\delta = 0.99$ and for χ_{C2} and χ_{V2} the certainty is $\delta = 1 - 10^{-6}$. For χ_{Ch} the certainty depends on the number of samples according to $\delta = \frac{1}{\sqrt{N}}$.

is defined to simplify notation. Furthermore, χ depends on the number of samples N and the VC dimension n_{VC}. The original bound stems from Vapnik [127] and reads

$$\chi_V = \tau \frac{n_{VC}\left(\ln\left(2\frac{N}{n_{VC}}\right)+1\right) - \ln\frac{\delta}{4}}{N}. \tag{2.110}$$

Here, the constant τ depends on form and existence of the moments of the probability distribution. If arbitrary moments exist (which is the case for most distributions including the uniform and the Gaussian distribution), then $\tau < \sqrt{3}$. Another practical bound, which seems to be a looser version of Vapnik's bound, has been described in [29]. The χ term for the bound reads

$$\chi_C = \frac{n_{VC}\left(\ln\left(\frac{N}{n_{VC}}\right)+1\right) - \ln\delta}{N} \tag{2.111}$$

There is also a version where the safety factor depends on the available amount of data and $\delta = \frac{1}{\sqrt{N}}$ is chosen. This results in

$$\chi_{Ch} = \frac{n_{VC}}{N} - \frac{n_{VC}}{N}\ln\frac{n_{VC}}{N} + \frac{\ln N}{2N} \tag{2.112}$$

where n_{VC} is the so-called VC dimension of an function [29]. The different bounds based on VC dimension are shown in Fig. 2.15. It can be seen that the bound derived by Vapnik is significantly looser than the later bounds given by [29]. Also, the asymptotic behavior is interesting. It can be seen that the quotient of the error

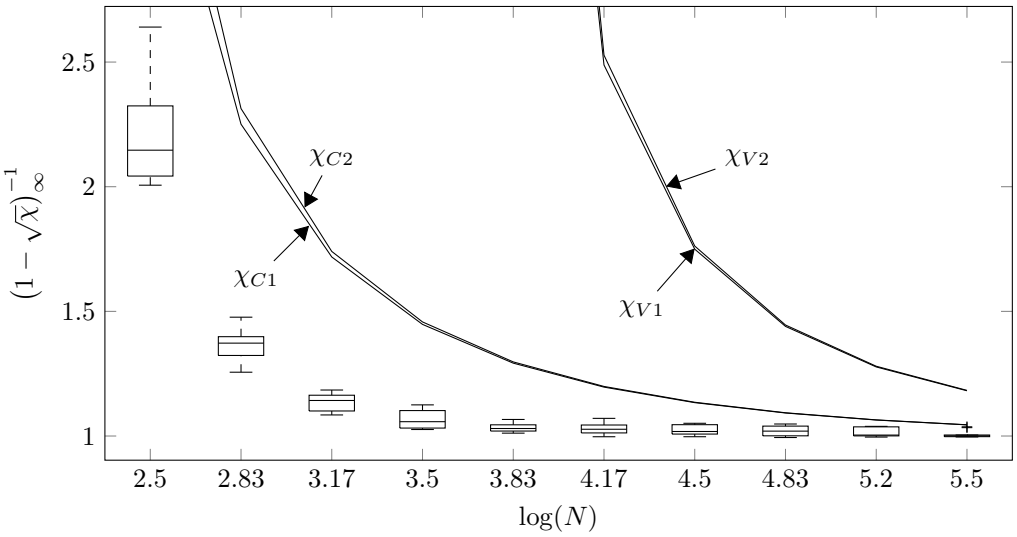

Figure 2.16: Comparison of VC-bounds for regression to an FIR system of a second-order system. For χ_{C1} and χ_{V1} the certainty is $\delta = 0.99$ and for χ_{C2} and χ_{V2} the certainty is $\delta = 1 - 10^{-6}$.

decreases by a rate depending on $\sqrt{N}$.

For further investigation of the bounds, a second order system oscillatory system

$$G(z) = \frac{0.476z + 0.456}{z^2 - 1.79 + 0.882} \tag{2.113}$$

is considered. The system is excited by independent white noise with unit variance and also the output of the system is disturbed by white noise with unit variance. An FIR system with $n_{\mathrm{VC}} = 60$ is identified with the generated data. The experiment is repeated 20 times for logarithmically spaced values of N. The resulting bounds are shown in Fig. 2.16. It can be seen that all bounds are always valid. However, especially in the areas which are practically very relevant $N = 1\,000$ up to $N = 10\,000$, the bounds are very loose. This is a significant drawback for the practical application of this method. Another disadvantage of the bounds is that sequential tree construction algorithms like LOLIMOT, do not perform an exhaustive search over the space of possible models. Instead, these algorithms as described in Sect. 2.1.4, follow a greedy strategy. To calculate the VC dimension of tree-based algorithms, the number of possible models has to be considered. This is not a good estimator since the reduction of complexity created by the greedy strategy is not taken into account. This makes the bound obtained even worse. Using this bound for model selection has the significant disadvantage that it will prefer very simple models and not perform as well as other more optimistic estimates, like the $\mathrm{AIC_c}$.

2.3.4 The Maximum Likelihood Approach for Bayesian Methods

If Bayesian methods are applied for identification, hyperparameters $\underline{\gamma}$ have to be tuned. One common way is the GCV approach described in Sect. 2.3.1. The other option is the ML method to identify the hyperparameters. This works as follows. The probability of the measured output

$$p(\underline{y}) = \int p(\underline{y}|\underline{\theta})p(\underline{\theta})d\underline{\theta} \tag{2.114}$$

is computed. Since it involves the computation of a complicated integral, this is analytically only possible for some special cases. These cases include Gaussian processes and the regularized FIR case. For the later it holds that [28]

$$p(\underline{y}) = \mathcal{N}\left(\underline{y}|\underline{0}, \rho\underline{X}\underline{P}\,\underline{X}^T + \sigma^2\underline{I}_N\right) = \mathcal{N}(\underline{y}|\underline{0}, \underline{\Sigma}_{\underline{y}}). \tag{2.115}$$

The log-likelihood can then be written as

$$\log p(\underline{y}) = -\frac{1}{2}\log\det\underline{\Sigma}_y - \underline{y}^T\underline{\Sigma}_{\underline{y}}^{-1}\underline{y} + \text{const.} \tag{2.116}$$

A significant burden lies in the computation of the second term since it includes terms with the inverse of $\underline{\Sigma}_{\underline{y}}$. The resultant matrix is $N \times N$ which hinders efficient computation. This can be overcome by the usage of efficient algorithms [26].

Some authors argue that the marginal likelihood approach works better than generalized cross-validation or information criteria [96]. This discussion is not exclusive for regularized FIR identification. It has already been discussed for the general case of kernel methods [128]. In fact, it seems that differences between these methods are marginal and strongly dependent on the random test systems employed for benchmarking.

Optimization of the marginal likelihood or GCV is a non-convex problem [71]. In [24], an approach for optimization of the hyperparameters is described, which can be formulated as a difference of convex optimization problems. For this type of problems, reliable heuristics for global optimization exist, see [65] for a recent overview.

2.3.5 Practical Recommendation

For practical purposes, it holds that estimates based on data unseen by the training algorithm are the safest alternative to asses the performance of a model. In the

end, this should be the method of choice if the number of available data allows for it. If this is not the case and it is computationally feasible (as for linear models), cross-validation methods provide a good compromise, especially for the tuning of hyperparameters. The same holds for ML methods for Bayesian identification. The

	Marginal Likelihood	**Generalized Cross-Validation**
Philosophy	The hyperparameters are estimated using the maximum likelihood approach	A rotation invariant formulation of the cross-validation error is applied for complexity selection
Model Mismatch	ML theory assumes that the model is true	No assumption about the model is required
Low rank $\underline{R}$ matrix	Additional projection step required, see Sect. 3.1.2	No additional calculations required
Uncertainty Estimate	integrated	not integrated

Table 2.2: Comparison of GCV and marginal likelihood for hyperparameter tuning of regularized FIR models.

usage of methods involving complexity measures seems to be the most sound approach from a first view. There are, however, two significant drawbacks. First, the bounds, as demonstrated, can be very loose, and second, it is challenging to determine the effect of regularization methods on the VC dimension of a model class.

3 Impulse Response Preserving Identification

Regularized identification of FIR systems offers, as described in Sect. 2.2.2, several advantages. In this section, a novel technique for the construction of penalty matrices is presented [77]. By this construction, penalty matrices can be tailored problem specific. This enhances the interpretability of the induced prior knowledge. Furthermore, the matrices can be applied to solve problems like order selection [83] and gray box identification [84].

Usually, the penalty matrix is constructed using Bayesian methods. This allows for the interpretation of sampling an impulse response from a prior Gaussian distribution. However, the choice of the kernel for the construction of this distribution seems to be somewhat arbitrary. For the stable-spline kernel [99], for instance, it is not clear why the form of the kernel is exactly the way it is. Furthermore, it remains unclear whether a first-order or a second-order spline kernel is the appropriate choice. It has been shown that the first-order stable spline kernel is the solution to a maximum entropy problem [25]. This provides some justification, but still, a relation to classical models used in signals and systems is not provided. This lack of understanding of the penalty term shall be overcome by the methods described in this chapter. Instead of relying on some space of prior functions, a class of penalty matrices is constructed, which penalizes deviations from systems of a specified order. This allows for enhanced integration of prior knowledge and also for the ability to integrate a first principles linear model into gray box identification techniques.

3.1 Filter Based Regularized FIR Model Identification

If the penalty matrix $\underline{R}$ for regularized identification is obtained from the Bayesian perspective, it holds that $\underline{R} = \underline{P}^{-1}$. Since the matrix $\underline{P}$ is a kernel matrix, it is positive definite by definition. Furthermore, the inverse will be positive definite as

well. The matrix $\underline{R}$ can thus be decomposed into $\underline{R} = \underline{F}^T \underline{F}$. This has an interesting consequence for the penalty term. It can be rewritten as

$$\underline{\theta}^T \underline{R}\, \underline{\theta} = \|\underline{F}\, \underline{\theta}\|_2^2 . \tag{3.1}$$

This result has been derived in [74]. There, it is shown that the different rows of $\underline{F}$ can be interpreted as FIR filters themselves. These filters act on the different components of $\underline{\theta}$. It is also argued that standard filter design schemes for low-pass or band-pass filters can be employed to construct the entries of the filter matrix. In [73], methods for appropriate hyperparameter tuning are addressed.

3.1.1 Computation of the Estimate with Low Rank Matrices

As described, two methods allow for a regularized formulation of the FIR identification problem. The first constructs a covariance matrix $\underline{P}$ with a kernel function, and the second constructs the penalty matrix $\underline{R}$ directly from the filter matrix $\underline{F}$. If both matrices are full rank, it holds that $\underline{R} = \underline{P}^{-1}$. This case can be handled by the methods described in Sect. 2.2.2. It is obvious that the matrix $\underline{R}$ cannot be allowed to be negative definite. If this was the case, there would be a linear combination of the entries of the parameter vector, which would make the solution of the optimization problem (2.77) arbitrarily small. A semidefinite $\underline{R}$, though, is possible. The same holds true for $\underline{P}$. Since, if $\underline{P}$ has negative eigenvalues, its inverse $\underline{R}$ has negative eigenvalues, too. This leads to the same undesirable consequences described above. It is also possible for $\underline{P}$ to be semidefinite. These cases of $\underline{P}$ or $\underline{R}$ being semidefinite are analyzed in detail.

Singular kernel matrix The first case is that the *kernel* or *covariance* matrix is not full rank. The rank of the matrix $\underline{P}$ is denoted as

$$\operatorname{rank} \underline{P} = r_P < n + 1. \tag{3.2}$$

This means that the prior distribution is collapsed to one or several directions. The number of collapsed directions will be equal to $n + 1 - r_P$, which is the rank deficiency of the covariance matrix. The fact that there is no variance along this direction restricts the solution of the optimization problem (2.77). It is equivalent to an additional linear constraint.
It can thus be understood as an extreme form of applying prior knowledge by assuming certainty along with this linear combination of parameters. The mathematical

analysis of this is done by using the singular value decomposition of $\underline{P}$. It can be computed as

$$\begin{aligned}\underline{P} &= \underline{U}_P \underline{S}_P \underline{U}_P^T \\ &= \begin{bmatrix}\underline{U}_{P1} \\ \underline{U}_{P2}\end{bmatrix} \begin{bmatrix}\underline{S}_P^* & \underline{0} \\ \underline{0} & \underline{0}\end{bmatrix} \begin{bmatrix}\underline{U}_{P1}^T & \underline{U}_{P2}^T\end{bmatrix}. \end{aligned} \tag{3.3}$$

The right and left singular vectors are equal since the kernel matrix is symmetric. The matrix $\underline{U}_{P1}$has as many rows as there are non-zero singular values. The matrix $\underline{U}_{P2}$ as many as there are zero singular values. Within the space spanned by the matrix $\underline{U}_{P2}$, there is no variance in the prior distribution for the parameters. This means for the optimization problem that these directions result in a constraint [100]. A formal proof of this argument can be found by setting the zero components on the diagonal of $\underline{S}_P$ to δ. The inverse can then be found as

$$\lim_{\delta \to 0} \begin{bmatrix}\underline{S}_P^* & \underline{0} \\ \underline{0} & \delta \underline{I}_{r_P - n - 1}\end{bmatrix}^{-1} = \lim_{\delta \to 0} \begin{bmatrix}\underline{S}_P^{*-1} & \underline{0} \\ \underline{0} & \underbrace{\frac{1}{\delta} \underline{I}_{r_P - n - 1}}_{\to \infty}\end{bmatrix}. \tag{3.4}$$

The penalty of the linear combinations described by $\underline{U}_{R2}$ tend to infinity, then. Thus, the optimization problem (2.83) can be rewritten as

$$\begin{aligned} \underset{\underline{\theta}}{\text{minimize}} \quad & \|\underline{y} - \underline{X}\,\underline{\theta}\|_2^2 + \lambda \underline{\theta}^T \underline{U}_{P1}^T \underline{S}_P^{*-1} \underline{U}_{P1} \underline{\theta} \\ \text{subject to} \quad & \underline{U}_{P2} \underline{\theta} = \underline{0} \end{aligned} \tag{3.5}$$

The optimal solution to this problem is the same as for the full rank optimization problem and is given by (2.83).

Singular penalty matrix The other possibility is that the rank of the *penalty* matrix is deficient. Rank deficiency has the consequence that several linear combinations of the parameters are not penalized. Also here, the singular value decomposition of $\underline{R}$ can be obtained to understand the consequences. The singular value decomposition of the penalty matrix is given as

$$\underline{R} = \underline{U}_R \underline{S}_R \underline{U}_R^T = \begin{bmatrix}\underline{U}_{R1} & \underline{U}_{R2}\end{bmatrix} \begin{bmatrix}\underline{S}_R^* & \underline{0} \\ \underline{0} & \underline{0}\end{bmatrix} \begin{bmatrix}\underline{U}_{R1}^T \\ \underline{U}_{R2}^T\end{bmatrix} = \underline{U}_{R1} \underline{S}_R^* \underline{U}_{R1}^T. \tag{3.6}$$

The rank of $\underline{R}$ is denoted as r_R. The matrix $\underline{U}_{R1}$ is $n+1 \times r_R$ and the matrix $\underline{U}_{R2}$ is $n+1 \times n+1-r_R$. The optimization problem can, in this case, be rewritten as

$$\underset{\underline{\theta}}{\text{minimize}} \quad \|\underline{y} - \underline{X}\ \underline{\theta}\|_2^2 + \lambda \underline{\theta}^T \underline{U}_{R1} \underline{S}_R^* \underline{U}_{R1}^T \underline{\theta}. \tag{3.7}$$

It can be seen that no terms of the form $\underline{U}_{R2}^T\ \underline{\theta}$ occur in the regularization term. Thus, this linear combination of the parameters can be chosen arbitrarily, such that the loss term $\|\underline{y} - \underline{X}\ \underline{\theta}\|_2^2$ is fit best.

3.1.2 Hyperparameter Estimation

If the penalty matrix is full rank, then both, generalized cross-validation and the marginal likelihood described in Sect. 2.2.2 can be applied. In the other case, though, it is not possible to compute the marginal likelihood since $\underline{P} = \underline{R}^{-1}$ does not exist. The likelihood is thus not well defined. This means that there is a linear combination of impulse response coefficients that can be arbitrary without affecting the penalty term. For this linear combination, no prior knowledge is imposed. The prior variance in this direction is infinitely large. If the marginal likelihood should be utilized as a mean of hyperparameter estimation, the problem has to be split into two parts. The frequentist parameters, for which no prior knowledge is induced

$$\underline{\theta}_F = \underline{U}_{R2}^T \underline{\theta} \tag{3.8}$$

and the Bayesian parameters for which prior knowledge is available

$$\underline{\theta}_B = \underline{U}_{R1}^T \underline{\theta}. \tag{3.9}$$

In [101], the prior knowledge free part of the impulse response parameters is introduced as a bias space. The directions not penalized by the kernel have to be dealt with separately. The output of the system can now be calculated, according to

$$\hat{\underline{y}} = \underline{X}\ \hat{\underline{\theta}} = \underline{X}\ \underline{U}_R \underline{U}_R^T \hat{\underline{\theta}} = \underline{X}\ \underline{U}_R \begin{bmatrix} \underline{U}_{R1}^T \\ \underline{U}_{R2}^T \end{bmatrix} \hat{\underline{\theta}} \tag{3.10}$$

$$= \underline{X}\ [\underline{U}_{R1}\ \underline{U}_{R2}] \begin{bmatrix} \underline{U}_{R1}^T \hat{\underline{\theta}} \\ \underline{U}_{R2}^T \hat{\underline{\theta}} \end{bmatrix} = \underbrace{\underline{X}\ \underline{U}_{R1} \hat{\underline{\theta}}_B}_{\hat{\underline{y}}_B} + \underbrace{\underline{X}\ \underline{U}_{R2} \hat{\underline{\theta}}_F}_{\hat{\underline{y}}_F} \tag{3.11}$$

Subtraction of the Frequentist part leads to the equation of the measured Bayesian part, according to

$$\underline{y}_B = \underline{y} - \underline{X}\ \underline{U}_{R2}\hat{\underline{\theta}}_F. \tag{3.12}$$

The Bayesian parameters $\underline{\theta}_B$ are a solution to the optimization problem

$$\underset{\underline{\theta}_B}{\text{minimize}} \quad \|\underline{y}_B - \underline{X}_B\ \underline{\theta}_B\|_2^2 + \lambda \underline{\theta}_B^T \underline{S}_R^* \underline{\theta}_B. \tag{3.13}$$

with the transformed $N \times r_R$ regressor matrix $\underline{X}_B = \underline{X}\ \underline{U}_{R1}$. For this problem, the penalty matrix $\underline{S}_R^*$ is of full rank. Thus, it has the same probabilistic interpretation as (2.81). According to (2.115), the marginal distribution of the output $\underline{y}_B$ is

$$\underline{y}_B \sim \mathcal{N}\left(\underline{0}, \underbrace{\rho \underline{X}\ \underline{U}_{R1} \underline{S}_R^{*-1} \underline{U}_{R1}^T \underline{X}^T + \sigma^2 \underline{I}_N}_{\underline{\Sigma}_{yB}}\right). \tag{3.14}$$

The log marginal likelihood of the Bayesian part can then be calculated as

$$\log p(\underline{y}_B) = -\frac{1}{2} \log\det \underline{\Sigma}_{yB} - \frac{1}{2} \underline{y}_B^T \underline{\Sigma}_{yB}^{-1} \underline{y}_B + \text{const.} \tag{3.15}$$

The algorithm for the computation of the marginal likelihood for rank deficient penalty matrices is summarized in Algorithm 1. Though in principle possible, the

Algorithm 1 Calculation of the marginal likelihood for rank deficient penalty matrices

Require: Penalty matrix $\underline{R}$, regressor $\underline{X}$, penalty matrix scaling ρ, noise variance σ^2, and output $\underline{y}$

1: Calculate the singular value decomposition of $\underline{R}$.
2: Compute the optimal parameters $\hat{\underline{\theta}}$ solving (2.78).
3: Transform $\hat{\underline{\theta}}$ to $\hat{\underline{\theta}}_F$ and $\hat{\underline{\theta}}_B$ with (3.8) and (3.9).
4: Calculate $\underline{y}_B$ with (3.12).
5: Compute $\underline{\Sigma}_{yB}$, according to (3.14).
6: **return** The logarithm of the marginal likelihood computed by (3.15).

derivation of the marginal likelihood requires computations of terms which depend on the inverse of $\underline{\Sigma}_{yB}$. Since this matrix is $N \times N$ calculation of these terms is computationally demanding. For the computation of the GCV error, these problems do not occur. Thus, for the techniques developed in this chapter GCV is used.

Example To exemplify the methodology, the system

$$y(k) = au(k) + b + n(k) \tag{3.16}$$

with i.i.d. Gaussian noise $n(k)$ and parameters a and b is considered. To solve the ML estimation problem, the regressor, the output vector, and the parameters are found as

$$\underline{X} = \begin{bmatrix} u(1) & 1 \\ \vdots & \vdots \\ u(N) & 1 \end{bmatrix} \qquad \underline{y} = \begin{bmatrix} y(1) \\ \vdots \\ y(N) \end{bmatrix} \qquad \underline{\theta} = \begin{bmatrix} a \\ b \end{bmatrix}. \tag{3.17}$$

Furthermore, the term $\lambda(a+b)^2$ is used as a regularization term for the identification. The loss function then reads as

$$J = \|\underline{y} - \underline{X}\theta\|^2 + \lambda \underline{\theta}^T \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}}_{\underline{R}} \underline{\theta}. \tag{3.18}$$

With this penalty term, regularization applies only to the sum of the parameters a and b. If the difference between a and b changes, the penalty is not affected. From a Bayesian perspective, this means that for the difference between a and b, there is no prior knowledge that has been integrated into the identification process. This, intuitively clear result, can also be found by the singular value decomposition of the penalty matrix

$$\underline{R} = \underbrace{\frac{\sqrt{2}}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{\underline{U}_R} \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \frac{\sqrt{2}}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3.19}$$

The transformed parameters are

$$\theta_F = \frac{\sqrt{2}}{2}(a-b) \qquad \theta_B = \frac{\sqrt{2}}{2}(a+b). \tag{3.20}$$

This confirms the intuition described above. The difference between the parameters is the non-Bayesian part θ_F, while the sum of both corresponds to the Bayesian part. For the computation of the projected ML value, the optimal parameters are estimated from the data. Then the Bayesian part of the output is calculated as

$$\underline{y}_B = \underline{y} - \underline{X}\frac{1}{2}\begin{bmatrix} a-b \\ b-a \end{bmatrix} \tag{3.21}$$

and used for the computation of

$$\underline{\Sigma}_{yB} = \rho \underline{X} \, \frac{\sqrt{2}}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \frac{1}{2} \frac{\sqrt{2}}{2} \begin{bmatrix} 1 & 1 \end{bmatrix} \underline{X}^T + \sigma^2 \underline{I}_N = \rho \underline{X} \frac{1}{4} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \underline{X}^T + \sigma^2 \underline{I}_N \tag{3.22}$$

For the application of this methodology to rank deficient penalty matrices for regularized FIR system identification, it will be the case that the entries of $\underline{U}_R$ change with the choice of the hyperparameters. Thus, a recomputation of $\underline{U}_R$ for each optimization step is necessary.

3.1.3 Analysis of the Optimal Kernel

If the true impulse response parameters $\underline{\theta}_0$ are known, it has been shown in [28] that the optimal kernel is given by

$$\underline{P} = \underline{\theta}_0 \, \underline{\theta}_0^T. \tag{3.23}$$

This formulation of the kernel matrix is a candidate for the tailored incorporation of prior knowledge. According to (3.23), a natural choice would be to set $\underline{\theta}_0$ to an a priori available estimate of $\underline{\theta}$ and construct $\underline{P}$. The impulse response coefficients are then estimated by a regularized identification approach with this choice of $\underline{P}$.

Low Rank Property of the Optimal Kernel The kernel, however, possesses an undesirable property, which will be referred to as rank one property. It holds that rank($\underline{P}$) = 1due to (3.23). Thus, the singular value decomposition of the optimal kernel matrix $\underline{P}_0$ reads

$$\underline{P}_o = \begin{bmatrix} \frac{\underline{\theta}_o}{\|\underline{\theta}_o\|_2} & \underline{U}_2 \end{bmatrix} \begin{bmatrix} \|\underline{\theta}_o\|_2^2 & \underline{0} \\ \underline{0} & \underline{0} \end{bmatrix} \begin{bmatrix} \frac{\underline{\theta}_o^T}{\|\underline{\theta}_o\|_2} \\ \underline{U}_2^T \end{bmatrix}. \tag{3.24}$$

This implies that if problem (3.7) is solved and the optimal solution is denoted by $\underline{\theta}^*$, then there exist n constraints (since the FIR model has $n+1$ coefficients) ensuring that

$$\underline{U}_2^T \underline{\theta}^* = \underline{0}. \tag{3.25}$$

Thus, the identified parameters depend linearly on the known parameters $\underline{\theta}^* = \alpha \underline{\theta}_0$. Since multiplying every FIR coefficient of an LTI system with a constant factor means nothing else than changing the gain of the system by $(n+1)\,\alpha$ the optimization problem is the same as linear regression of the gain of that system. This is illustrated in Fig. 3.1 for a second order system with two non-oscillatory poles. The

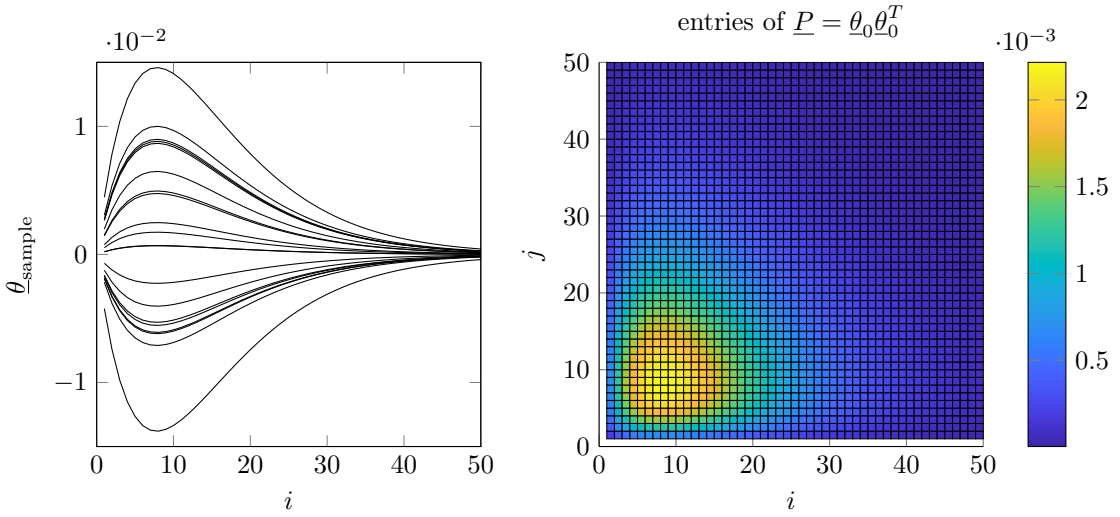

Figure 3.1: Samples from the optimal kernel for a second order system with two non-oscillatory poles.

left plot shows samples of the prior distribution described by a system with two non-oscillatory poles. On the right side of the figures, the entries of the resulting kernel matrix are shown. Solving problem (3.7) with the hyperparameters as additional optimization variables is, in this case, approximatively equivalent (due to the truncation of the impulse response) to the identification of an output error model. Regularized identification is not better than known methods, like OE identification.

3.2 Impulse Response Preserving Matrices

In this section, a novel approach for the incorporation of prior knowledge is described. To extend the possible forms of prior information which can be incorporated in regularization techniques for system identification, the impulse response preserving (IRP) matrices are introduced. These matrices allow for incorporating prior knowledge of the dynamic behavior of the system. In contrast to the optimal kernel described in Sect. 3.1.3, this method does not lead to a singular $\underline{P}$ matrix.

3.2.1 Example: Second Order System

To demonstrate the construction of IRP matrices, a second order system with the transfer function

$$G(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{3.26}$$

is considered. Transformation in the discrete time domain results in the following equation

$$y(k) = b_0u(k) + b_1u(k-1) + b_2u(k-2) - a_1y(k-1) - a_2y(k-2). \tag{3.27}$$

The impulse response of the system can be calculated using an input equal to the discrete Dirac function. It is one at $k = 0$ and zero everywhere else. The impulse response coefficients are thus

$$\begin{aligned} \theta(0) &= b_0 & \theta(3) &= -a_2\theta(1) - a_1\theta(2) \\ \theta(1) &= b_1 - a_1\theta(0) & &\vdots \\ \theta(2) &= b_2 - a_2\theta(0) - a_1\theta(1) & \theta(k) &= a_2\theta(k-2) - a_1\theta(k-1). \end{aligned} \tag{3.28}$$

Here, two properties of the impulse response of a second order system can be observed. The first property is that the first three coefficients $\theta(0), \theta(1)$, and $\theta(2)$ can be chosen arbitrarily by the parameters b_0, b_1, and b_2. The second fact is that by rearranging the last equation of (3.28) it can be seen that

$$a_2\theta(k-2) + a_1\theta(k-1) + \theta(k) = 0 \text{ for } k \geq 3. \tag{3.29}$$

This is a linear equation for the parameters of the impulse response coefficients. Such relations will be called *impulse response preserving conditions*. If this condition is fulfilled, then the system possesses the exact impulse response which was assumed a priori. These conditions can be rewritten in matrix form as

$$\underbrace{\begin{bmatrix} a_2 & a_1 & 1 & 0 & \dots & 0 \\ 0 & a_2 & a_1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & \dots & 0 & a_2 & a_1 & 1 \end{bmatrix}}_{\underline{F}} \underline{\theta} = \underline{0}. \tag{3.30}$$

For $n+1$ FIR coefficients, there will be $n-1$ impulse response preserving conditions. Thus, the size of $\underline{F}$ is $n-1 \times n+1$. This matrix will be referred to as an **IRP matrix**. The idea of the described approach is to calculate $\underline{R} = \underline{F}^T\underline{F}$ with this matrix and solve the optimization problem (3.7). This means that deviations from the impulse response preserving equations will be penalized. If the impulse response is exactly equal to the true impulse response, then the penalty will also be zero.

3.2.2 Systems of Arbitrary Order

The result from the previous subsection can be extended to LTI systems of arbitrary order. Two kind of orders have to be distinguished now. The first is the order of the FIR system, which will be denoted as regularly with n. The second is the prior order o of the infinite impulse response (IIR) system available a priori for the integration of prior knowledge. Given an IIR system of the order o, the recursive equations for the computation of the impulse response read as

$$\begin{aligned} \theta(0) &= b_0 & \theta(3) &= b_3 - a_1\theta(2) - a_2\theta(1) - a_3\theta(0) \\ \theta(1) &= b_1 - a_1\theta(0) & &\vdots \\ \theta(2) &= b_2 - a_1\theta(1) - a_2\theta(0) & \theta(i) &= b_i - \sum_{j=1}^{i} a_i\theta(i-j) \end{aligned} \tag{3.31}$$

for all $i \leq o$ and

$$\theta(i) = -\sum_{j=1}^{o} a_j\theta(i-j) \tag{3.32}$$

for all others. The preserving conditions described by (3.32) can be summarized in a set of linear equations $\underline{F}\,\underline{\theta} = \underline{0}$ with

$$\underline{F} = \begin{bmatrix} a_o & a_{o-1} & \dots & a_1 & 1 & 0 & \dots & 0 \\ 0 & a_o & a_{o-1} & \dots & a_1 & 1 & \dots & 0 \\ \vdots & & \ddots & \ddots & & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & a_o & a_{o-1} & \dots & a_1 & 1 \end{bmatrix}. \tag{3.33}$$

For $o = 2$, this results in exactly the same set of equations as for a second order system given by (3.30). There are $n + 1 - o$ impulse response preserving equations for $n+1$ coefficients. Thus the size of $\underline{F}$ is $n+1-o \times n+1$. By (3.33) it is possible to assign an IRP matrix to each IIR model. The matrix described by (3.33) will be referred to as FIRP matrix of order o. This choice, however, is not unique, as will be elaborated on in Sect. 3.2.3.

If FIR systems are employed to represent IIR systems, the resulting system will always be an approximation. The coefficients subsequent to the n-th coefficient are zero in the FIR representation but are non-zero for the IIR system (although they are exponentially decaying and thus very small). Thus, the $\underline{F}$ matrix described by (3.33) will be referred to as *finite impulse response preserving* (FIRP) matrix.

If the assumption $\theta(n+1) = \theta(n+2) = \ldots = 0$ is considered, there are n further equations that can be derived from (3.32). For the last but one coefficients, for example, it holds that $a_1\theta(n-1)+a_2\theta(n) = 0$. For the last, it holds that $a_1\theta(n) = 0$. If these equations are put in matrix form, the resulting preserving matrix can be extended to

$$\begin{bmatrix} a_o & a_{o-1} & \ldots & a_1 & 1 & 0 & \ldots & 0 \\ 0 & a_o & a_{o-1} & \ldots & a_1 & 1 & \ldots & 0 \\ \vdots & & \ddots & \ddots & & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & a_o \end{bmatrix}. \tag{3.34}$$

This matrix contains the rows of the matrix (3.33) and has o additional rows at the end. In general, the last equations will not be zero for the true coefficients. It can be shown, see Sect. 3.3, that for $n \to \infty$ these penalty terms tend to zero. This matrix is thus called *infinite impulse response preserving* (IIRP). It is of quadratic $n+1 \times n+1$ size. In general, $\underline{F}$ and in consequence also $\underline{R} = \underline{F}^T\underline{F}$ has full rank. This enables the computation of $\underline{P}$. Therefore, a Bayesian interpretation is possible. The properties of the FIRP and IIRP matrices are summarized in Tab. 3.1.

property	**FIRP**	**IIRP**
rank of $\underline{R}$	deficient	full
penalty for finite n	zero	non-zero
Bayesian interpretation	requires a non-Bayesian bias space	possible

Table 3.1: Properties of FIRP and IIRP matrices.

The FIR models considered here are proper, but not strictly proper. Thus, $u(k)$ can influence the output via the first FIR coefficient $\theta(0)$ directly. If strictly proper FIR models, where $\theta(0)$ is forced to be zero, are considered, then one difference is that the dimension of $\underline{\theta}$ is n instead of $n+1$, since $\theta(0)$ is left out. Furthermore, the IIRP or FIRP matrices will contain one row and column less. This does not affect or even hinder the applicability of the proposed method.

In practice, however, it seems that whether (3.33) or (3.34) is chosen for regularization does not differ much with regard to the performance on test data.

3.2.3 Possible Weightings

In fact, a preserving matrix can be multiplied by an arbitrary non-zero diagonal weighting matrix and still be preserving for the corresponding impulse response.

This holds since for any non-zero diagonal matrix if $\underline{W}\ \underline{F}\ \underline{\theta} = \underline{0}$ is a valid statement, $\underline{F}\ \underline{\theta} = \underline{0}$ is also valid. For the purpose of regularized identification, several choices are possible. The first possibility is an exponentially increasing weighting matrix. This leads to the matrix

$$\underline{W} = \mathrm{diag}(1, \alpha^{-\frac{1}{2}}, \alpha^{-1}, \ldots, \alpha^{-\frac{n}{2}}) \tag{3.35}$$

with $\alpha < 1$. This choice is related to the exponential decay of the TC kernel. This exponential choice puts high penalties on the terms formed by the last rows of the IRP matrix and, therefore, on the last coefficients. To avoid these high weights, a linear weighting

$$\underline{W} = \mathrm{diag}\left(1, 1 + \frac{a-1}{n}, 1 + \frac{2(a-1)}{n}, \ldots, a\right) \tag{3.36}$$

with $a > 1$ can be used. A single parameter a for the control of the linear weighting is sufficient since the penalty matrix itself is additionally multiplied by λ. Furthermore, the most trivial alternative is to use constant weighting

$$\underline{W} = \underline{I}. \tag{3.37}$$

To analyze the effect of weighting matrices on the estimated impulse responses, the singular value decomposition of the penalty matrix $\underline{R} = \underline{U}_R\, \underline{S}_R\, \underline{U}_R^T$ is considered again. The parameter vector of the optimization problem is transformed into

$$\underline{\theta}_T = \underline{U}_R^T\, \underline{\theta}. \tag{3.38}$$

Since $\underline{U}_R$ is unitary, e.g. that $\underline{U}_R^T\, \underline{U}_R = \underline{I}$, it holds that

$$\underline{\theta} = \underline{U}_R \underline{\theta}_T. \tag{3.39}$$

If this is used to transform (2.77), it follows that the identification problem can be rewritten as

$$\min_{\underline{\theta}_T} \left\| \underline{y} - \underline{X}\, \underline{U}_R \underline{\theta}_T \right\|_2^2 + \lambda \sum_{i=1}^{r_R} S_{R,ii}^2 \theta_{T,i}^2. \tag{3.40}$$

The last sum follows from the fact that

$$\underline{\theta}^T\, \underline{R}\, \underline{\theta} = \underline{\theta}_T^T \underline{U}_R^T \underline{U}_R\, \underline{S}_R\, \underline{U}_R^T\, \underline{U}_R \underline{\theta}_T = \underline{\theta}_T^T \underline{S}_R \underline{\theta} = \sum_{i=1}^{r_R} S_{R,ii}^2 \theta_{T,i}^2. \tag{3.41}$$

This derivation shows the relation of regularized FIR models to OBF models as discussed by [27, 33]. In fact, the construction of appropriate orthonormal basis functions and how to incorporate prior knowledge in the form of poles have also been addressed [92]. Here, the OBF system is constructed considering available prior knowledge on the poles of the system. In contrast to our method, this approach requires these poles to be fixed. It is possible to include the poles for OBF construction in an optimization problem, but the risk that the identification result is poor due to local optima remains.

The matrix $\underline{X}\,\underline{U}_R$ contains in its i,j-th entry the convolution of the delayed inputs contained in the i-th column of $\underline{X}$ with a filter described by the j-th row of $\underline{U}_R$. The columns of the matrix $\underline{U}_R$ can thus be interpreted as orthonormal basis functions. The identification problem is then the identification problem with a ridge regularization on the parameters for the OBFs characterized by the columns of $\underline{U}_R$. If the OBFs are ordered in descending order with respect to the singular value, reordering of the indices results in the impulse response coefficients

$$g_i^{(\mathrm{OBF})}(k) = [\underline{U}_R]_{k,n+2-i} \tag{3.42}$$

for the i-th least penalized OBF. In Fig. 3.2 the first 6 least penalized OBFs for the TC kernel are shown. This means that the first upper left plot shows the last column of $\underline{U}_R$. The basis functions are plotted in descending order by their singular value. To make the penalty comparable, the caption of each figure shows the singular value relative to the least non-zero value of the singular values. The basis functions for the TC kernel show clearly where most flexibility is put. Due to the exponentially increasing penalty, most flexibility is in the first basis functions. It can also be seen that already the third basis function shows oscillatory behavior at the beginning. This explains the ability of the TC kernel to approximate even oscillatory systems well. Also, the first order FIRP matrix with exponential weighting and $\alpha = -a_1 = 0.95$ is analyzed. It can be seen in Fig. 3.3 that the behavior is very similar to the TC kernel, except for the penalty of the first basis function. This penalty is zero in the FIRP matrix case.

To demonstrate the effect of higher order FIRP matrices, in Fig. 3.4 an exponentially weighted second order FIR matrix with $\alpha = 0.95$, $a_1 = -1.92$, and $a_2 = 0.957$ is shown. The corresponding transfer function has two oscillatory poles. In this case, the resulting OBFs differ significantly compared to the TC kernel or exponentially weighted IRP matrix case. For this penalty matrix, the oscillatory behavior is clearly

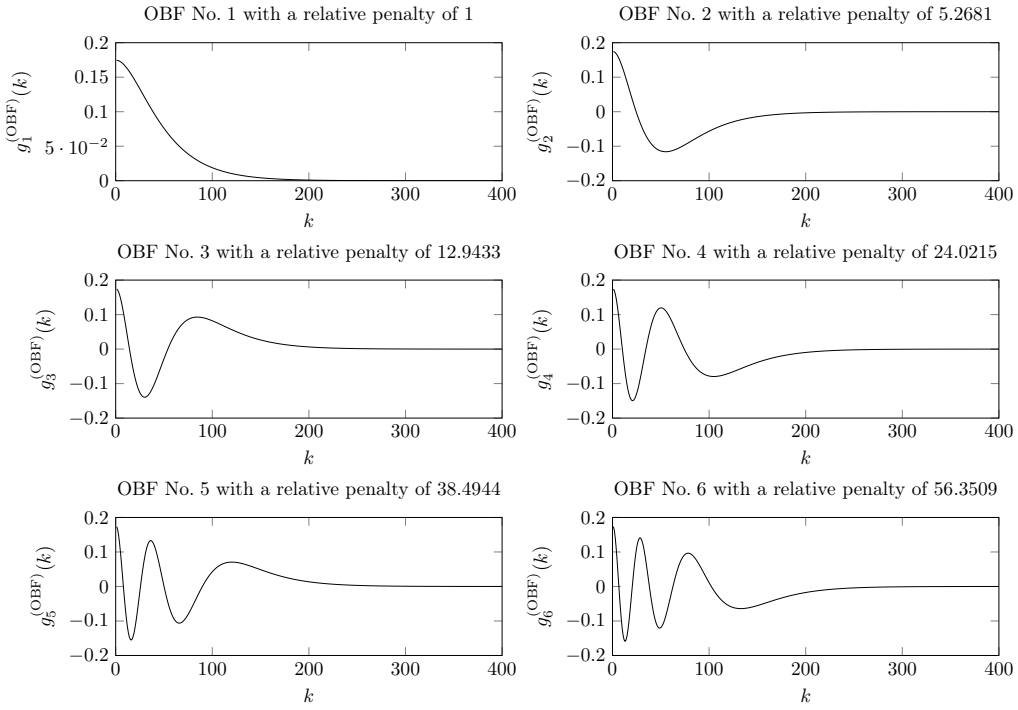

Figure 3.2: The 6 least penalized OBFs ($i = 1, 2, \ldots, 6$) with corresponding relative penalty weightings of the TC kernel with $\alpha = 0.95$

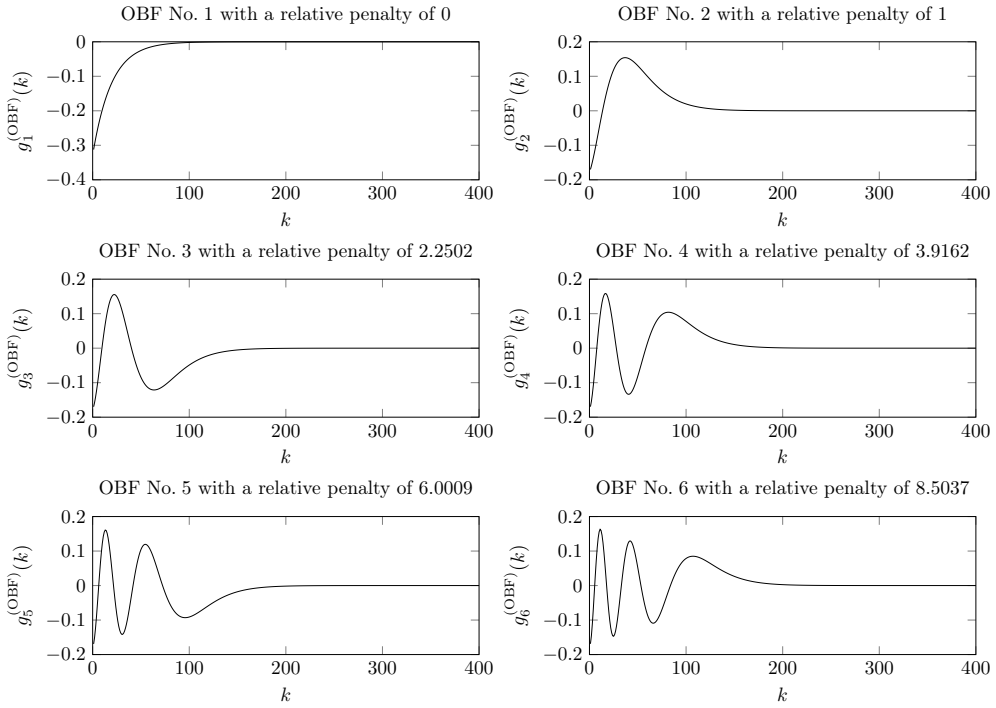

Figure 3.3: The 6 least penalized OBFs ($i = 1, 2, \ldots, 6$) with corresponding relative penalty weightings of the exponentially weighted first order FIRP matrix with $a_1 = -0.95$ and $\alpha = 0.95$

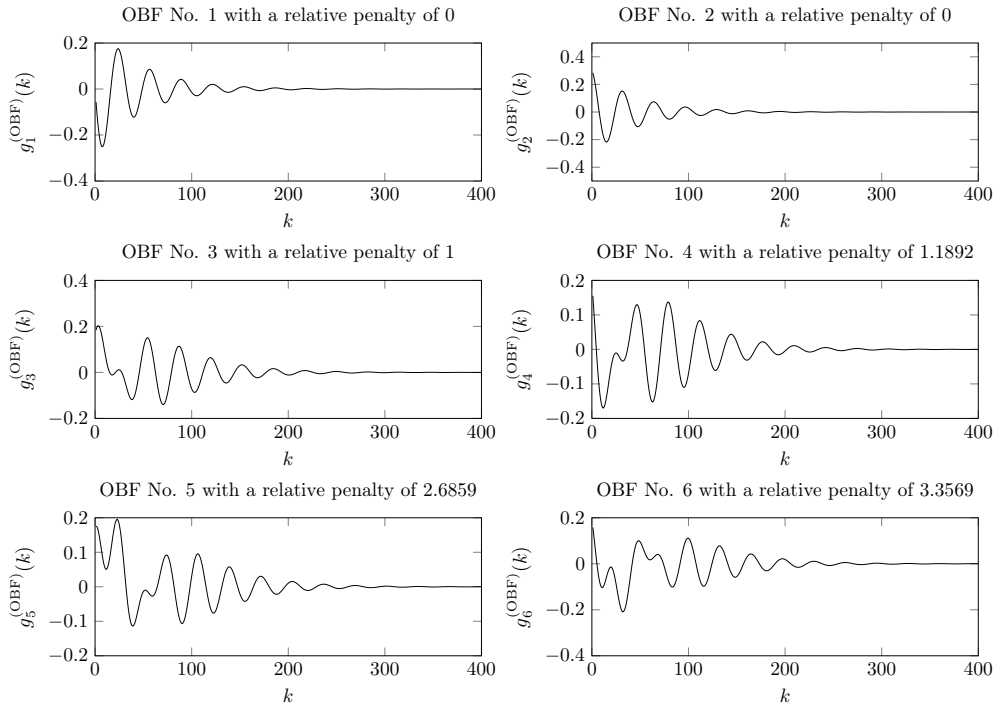

Figure 3.4: The 6 least penalized OBFs ($i = 1, 2, \ldots, 6$) with corresponding relative penalty weightings of the exponentially weighted second order FIRP matrix with $\alpha = 0.95$, $a_1 = -1.92$, and $a_2 = 0.957$

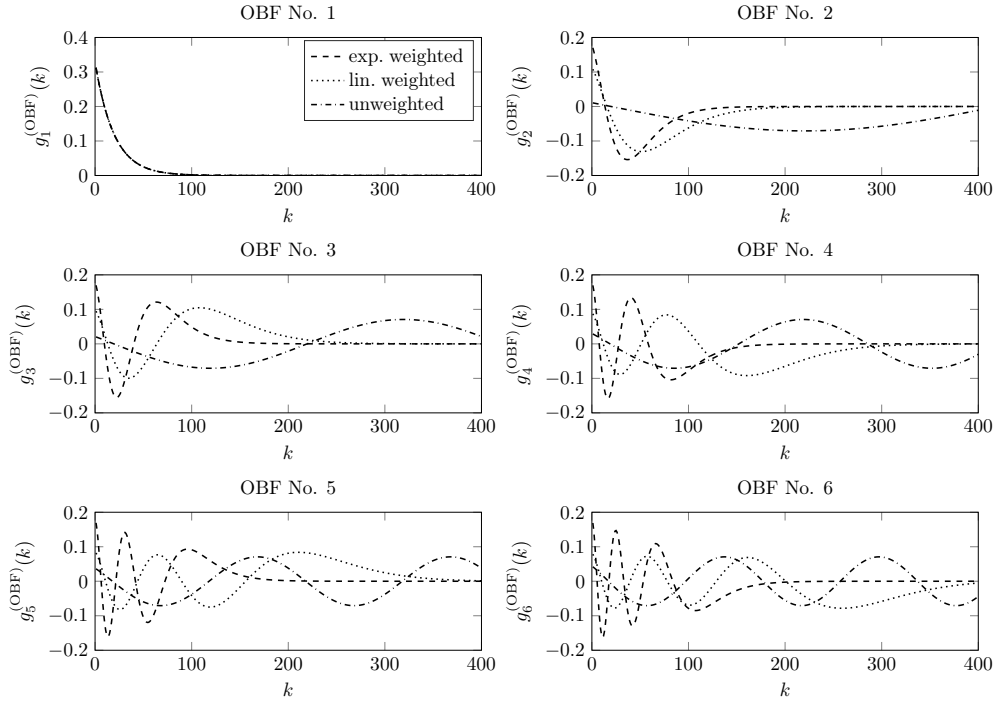

Figure 3.5: The 6 least penalized OBFs ($i = 1, 2, \ldots, 6$) for different weightings (dashed: exp weighted with $\alpha = 0.95$, dotted: linear weighted with $a = 10$, dash-dotted: unweighted) of the first order FIRP matrix.

visible. Also, the higher order OBFs (≥ 3) show oscillatory behavior. If this matrix is utilized to impose prior knowledge, oscillatory parts of the impulse response will be penalized least.

The effect of the weighting is shown in Fig. 3.5. The first order FIRP matrix is analyzed with $a_1 = 0.95$ without weighting, with exponential weighting ($\alpha = 0.95$), and with linear weighting ($a = 10$). Without weighting, the higher order terms look very similar to sinusoidal functions. This is not very realistic for impulse responses. The usage is thus not recommended for practical problems. Both the linear weighting and the exponential weighting generate reasonable OBFs and can be applied for practical problems.

3.2.4 Example

To illustrate the principle of IRP matrix based RFIR identification, the following example composed out of two parts is considered. The transfer function of the first part

$$G_p(z) = \frac{(1 - 0.94)z}{z - 0.94} \tag{3.43}$$

describes a first order system with unit gain. It is assumed that for this part of the system, perfect prior knowledge is available. The second transfer function

$$G_u(z) = \frac{-0.0148z^2 + 0.0148z}{z^2 - 1.774z + 0.922} \tag{3.44}$$

has an oscillatory pole pair at $p_{1,2} = 0.96e^{\pm i\frac{\pi}{8}}$ and zero gain (differential characteristic). For the example, no prior knowledge shall be available for this part. In practice, it could be that this oscillatory behavior is completely unrecognized during physical model building. The overall transfer function is

$$G(z) = G_p(z) + G_u(z). \tag{3.45}$$

The system is excited by 1 000 samples of a PRBS signal and disturbed by i.i.d. Gaussian noise with variance $\sigma^2 = 2.5 \cdot 10^{-3}$. A first order FIRP matrix with $a_1 = -0.94$ is constructed and weighted by an exponential weighting with $\alpha = 0.94$. This corresponds to the weighting of the TC kernel. In Fig. 3.6 the identification results for different values of the regularization parameter λ and FIR model order $n = 100$ are depicted. The effective number of parameters which is the trace of $\underline{S}$ is also shown. If $\lambda = 0$, then the identification result is the same as for unregularized FIR. This is

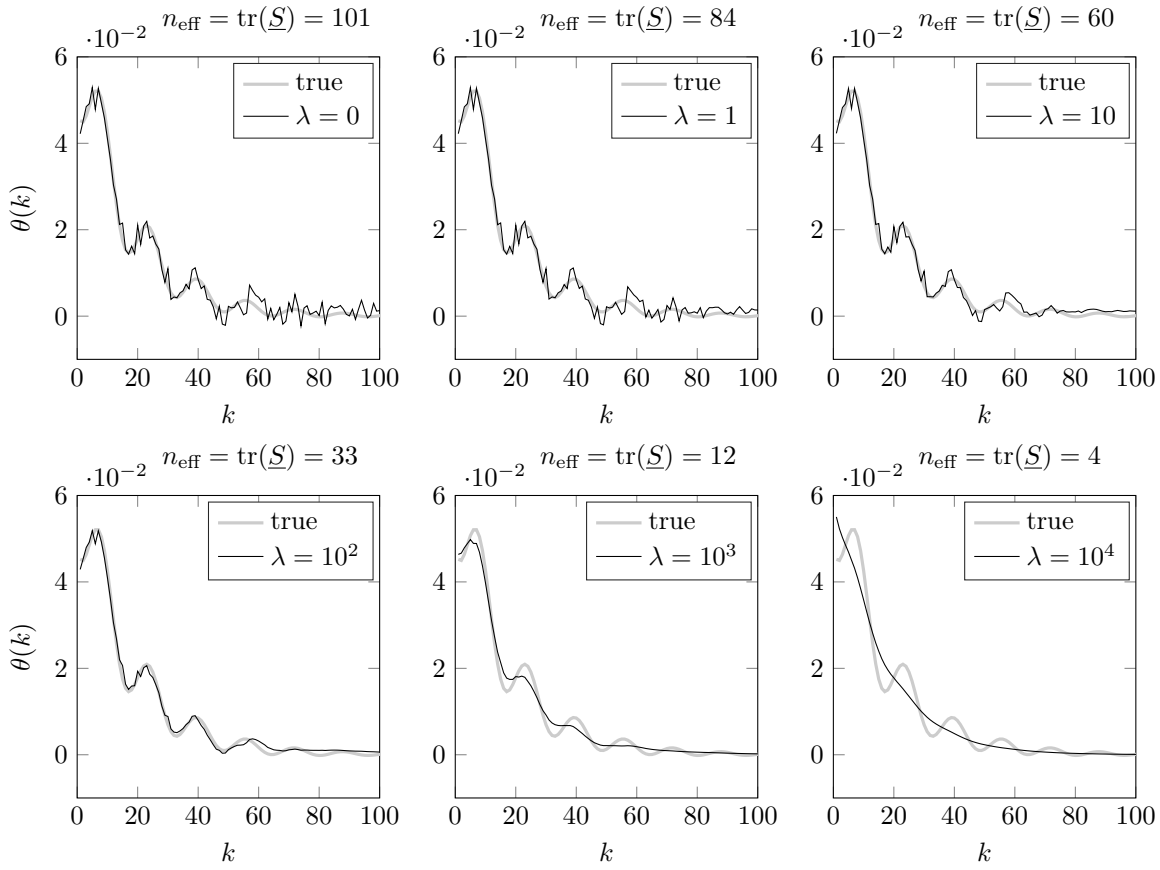

Figure 3.6: Identified impulse responses with an exponentially weighted FIRP matrix of first order. The regularization parameter is increased from 0 to 10^4, and the effective number of parameters for different impulse response estimates are depicted.

shown in the upper left plot in the figure. The variance error of the FIR model is clearly visible, especially at the end of the impulse response. As the regularization parameter λ is continuously increased, the variance error is reduced. The higher λ is chosen, the smaller the effective number of parameters becomes. This explains the reduction of variance error. A high λ of 10^4 limits the capacity of the model to 4.05 effective parameters. If λ is increased further, then the effective number of parameters is equal to 1. The impulse response is equal to a first order system, and only the gain of the impulse response is identified. The optimal value of the effective number of parameters lies between the extremes of 101 and 1. For the example, $\lambda = 10^2$ is an appropriate choice. The example demonstrates the power of prior knowledge for system identification since without prior assumptions, one cannot become better than the dissatisfying response of the upper left subfigure.

3.3 Mathematical Analysis of Impulse Response Preservation

In this subsection, the behavior of impulse response preserving penalty matrices is analyzed in more detail. True impulse response coefficients for first or higher order IIR systems are never exactly zero for any finite time step k. If impulse response preserving matrices are constructed according to the IIRP construction rules given by (3.34), an interesting question is what the penalty will look like if the number of observations tends to infinity. To analyze the behavior of the parameters of the impulse response, the following notation is introduced. The vector

$$\underline{\theta}_n = [g(0)\ g(1)\ \cdots\ g(n)]^T \tag{3.46}$$

contains the first $n+1$ impulse response coefficients. Now, an impulse response coefficient vector $\underline{\theta}_n$ is said to be *preserved* by an IRP filter matrix with n rows $\underline{F}_n$, if $\|\underline{F}_n\underline{\theta}_n\|_2 = 0$ holds. The FIRP matrix given by (3.33) is constructed to ensure this. If the prior covariance matrix $\underline{P}_n$ for an FIR system of order n has full rank, like each IIRP matrix, then there is no $\underline{\theta}_n$ which is preserved by the corresponding $\underline{F}_n$. For penalty matrices, we call a sequence of filter matrices $\underline{F}_n$ *asymptotically preserving* for a vector of impulse response coefficients $\underline{\theta}_n$ if

$$\lim_{n\to\infty} \|\underline{F}_n\underline{\theta}_n\|_2 = 0. \tag{3.47}$$

As will be discussed in the next subsections, an IIRP matrix is asymptotically preserving for the true coefficients of the corresponding impulse response.

3.3.1 Preservation Properties of the TC Kernel

Before impulse response preserving properties are made more explicit mathematically, the behavior of penalty matrices is analyzed in a simple experiment. Therefore two different impulse responses, shown in the upper part of Fig. 3.7, are considered. The first one is simply constant, and the second one is exponentially decaying and thus the impulse response of a stable first order LTI system. For each of these impulse responses, the entries of the products $\underline{F}\,\underline{\theta}$ are considered. In the second row of the figure, the TC kernel is considered. The inverse and the Cholesky decomposition of the TC kernel can be computed analytically [74]. The filter matrix is the result

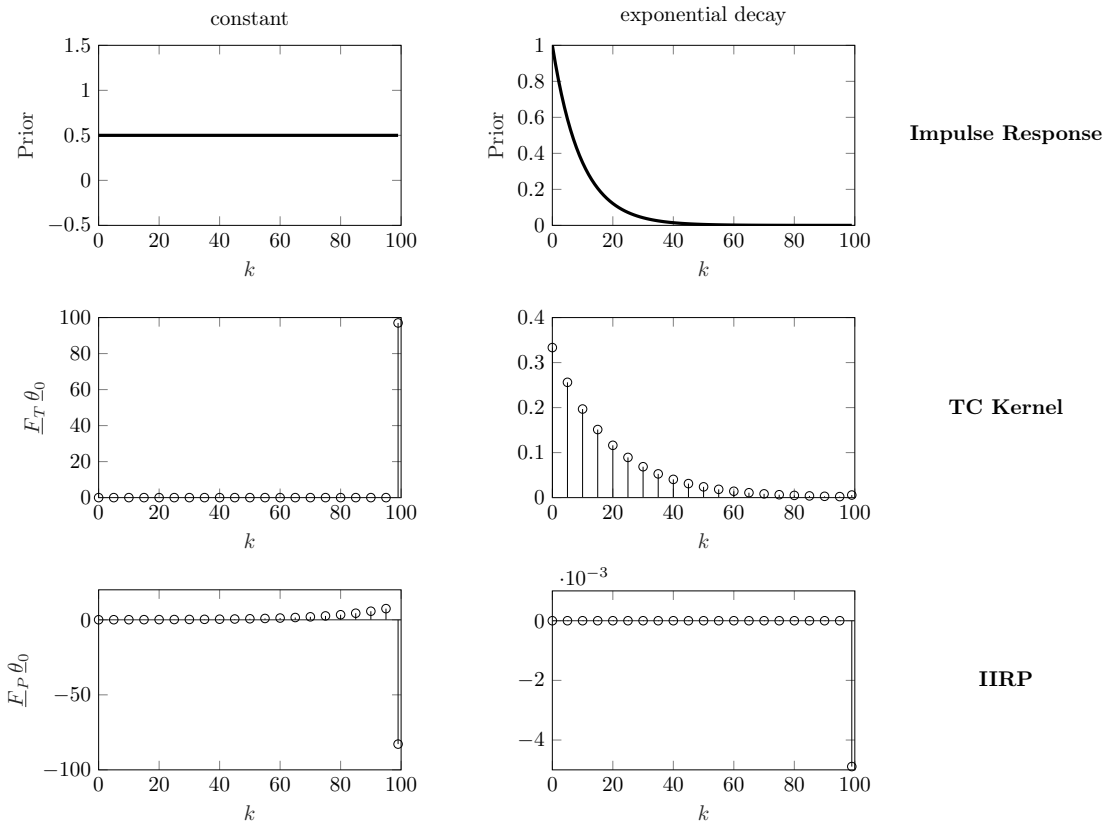

Figure 3.7: Illustration of the effect of different filter matrices on the true impulse response coefficients of a constant (left) and exponentially decaying impulse response (right). The entries of the penalties for the filter matrix of the TC kernel (second row) and the IIRP matrix of first order with exponential weighting (third row) are shown.

of the Cholesky decomposition for an impulse response with n entries and reads

$$\underline{F}_n = \begin{bmatrix} 1 & -1 & 0 & 0 & \dots & 0 \\ 0 & \sqrt{\frac{1}{\alpha}} & -\sqrt{\frac{1}{\alpha}} & 0 & \dots & 0 \\ 0 & 0 & \sqrt{\frac{1}{\alpha^2}} & -\sqrt{\frac{1}{\alpha^2}} & \dots & 0 \\ \dots & \dots & \dots & \dots & \ddots & \dots \\ 0 & 0 & 0 & 0 & \dots & \sqrt{\frac{1}{\alpha^n}} \end{bmatrix}. \tag{3.48}$$

This filter matrix is applied to both the constant and the exponentially decaying impulse response. In the constant impulse response case, all resulting penalty terms are zero, except for the last one. For the exponential decay, the first terms are penalized, although a first order system has an ideal exponential decay. In the last row of the figure, a first order IRP matrix, which matches the coefficients of the exponentially decaying impulse response, is constructed. The same analysis as for the TC kernel is conducted. It can be seen that now the penalty term entries are non-zero for the constant impulse response. For the exponentially decaying case, all penalty terms, except for the last one, are zero.

3.3.2 Preserving Filter Matrices for Stable LTI Systems

Here, we will show that the proposed matrix is asymptotically preserving for arbitrary stable LTI systems. To analyze the preserving conditions for the IRP matrices given by (3.34), the entries of the vector $\underline{F}_n\underline{\theta}_n$ are considered. The index n emphasizes the dependence of the matrix size on the order of the FIR system. The entries of the vector can be calculated as

$$[\underline{F}_n\underline{\theta}_n]_i = \begin{cases} s_i & i \leq n-o+1 \\ g_i & \text{else} \end{cases} \tag{3.49}$$

with

$$s_i = \theta_n(i+o-1) + \sum_{j=1}^{o} a_j\theta_n(i+o-j-1) \tag{3.50}$$

and

$$g_i = \sum_{j=0}^{n-i+1} a_{o-j}\theta_n(j+i-1). \tag{3.51}$$

The quantity s_i is exactly the deviation from the IRP conditions. Since the true coefficients $\underline{\theta}_0$ fulfill these conditions exactly, it holds that $s_i = 0$. This is different for the g_i coefficients. Here, some terms from the IRP conditions are missing. It

is thus interesting to analyze the case that $n \to \infty$. If g_i converges to zero, in this case, it means that the penalty imposed on the true impulse response can be made arbitrarily small by choosing an appropriately long impulse response to be fit. For arbitrary stable LTI systems, it holds that there is always an β_s with $0 < \beta_s < 1$ and a constant G such that it

$$|\theta_0(k)| \leq G\beta_s^k \tag{3.52}$$

with $\theta_0(k)$ denoting the true impulse response coefficients [124]. Now, for the limit of $n \to \infty$, for the last coefficients g_i it is found that

$$\lim_{n\to\infty} |g_i| = \lim_{n\to\infty} \sum_{j=1}^{n-i+1} |a_{o-j}\theta_n(j+i-1)| \leq \lim_{n\to\infty} (n-i+1)G\beta_s^n = 0 \tag{3.53}$$

for the true impulse response. The penalty for the true impulse response thus converges to zero if n is chosen high enough. If exponential weighting is considered, the quantities change. Then the weighted version of g_i is found as $g_i^* = \alpha^i g_i$. Finally, it holds for the limit that

$$\lim_{n\to\infty} |g_i^*| = \lim_{n\to\infty} \alpha^{-i} \sum_{j=1}^{n-i+1} |a_{o-j}\theta_n(j+i-1)| \leq \lim_{n\to\infty} \alpha^{-n}(n-i+1)G\beta_s^n. \tag{3.54}$$

If $0 < \alpha < \beta_s$ the last term sure converges to zero. This means that exponential preservation can be guaranteed if α for the exponential weighting is chosen lower in magnitude than the dominant pole of the system.

3.4 Performance on Numerical Benchmark Systems

For the identification of regularized FIR systems, the TC kernel [99, 28] achieves state-of-the-art performance on numerical benchmarks. The behavior of the proposed method is now evaluated on numerical benchmark problems. The investigated hypothesis is that additional prior knowledge added by the proposed penalty matrices results in better identification performance. The applied method is summarized in Algorithm 2.

The hyperparameters of the algorithm are $\underline{\gamma} = [a_1, a_2, \ldots, a_o, \alpha, \lambda]^T$. Thus, for an o-th order IRP matrix, $o+2$ hyperparameters are optimized. The calculations in this section are done with IIRP matrices, and for hyperparameter tuning, the GCV method is employed. For an identification problem with $n = 100$ and $N = 500$, identification, including hyperparameter tuning, takes in mean 10 s on a standard

Algorithm 2 Regularized FIR identification based on GCV and IRP matrices

Require: Order o of the IRP system, order n of the FIR system, output $\underline{y}$, and FIR regressor matrix $\underline{X}$

1: Optimize the GCV given by (2.94) with respect to the hyperparameters $\underline{\gamma} = [a_1, \ldots, a_n, \alpha, \lambda]^T$
2: Use the optimal $\hat{\underline{\gamma}}$ from the previous step to construct the optimal IRP matrix $\underline{R}(\hat{\underline{\gamma}})$ with the filter matrix from (3.33).
3: Calculate the optimal parameters $\hat{\underline{\theta}}$ by solving (3.7)
4: **return** The optimal FIR parameters $\hat{\underline{\theta}}$

desktop PC.

3.4.1 Random First and Second Order Systems

Therefore, we investigate the case that the order of the investigated system and the applied kernel coincide. The investigation is started with first order dynamic systems. A first order system is described by the transfer function

$$G(z) = \frac{K(1-p)z}{z-p} \tag{3.55}$$

with pole p and gain K. The input signal for the system is chosen to be a PRBS signal with $N = 500$. The output of the system is disturbed by i.i.d. Gaussian white noise. The noise is chosen such that the signal to noise ratio

$$S = 10 \log_{10} \frac{\sum_{k=1}^{N} (y(k) - \bar{y})^2}{\sigma_n^2} \tag{3.56}$$

is equal to 3 dB. This high noise level makes the identification challenging in this case. Ten different models for fixed p and K but different noise realizations are depicted in Fig, 3.8. Both penalty terms are able to guarantee that the impulse response decays near to zero after 100 steps. The behavior induced by the first order IIRP matrix is, however, smoother than the behavior induced by the TC kernel. This can be seen between $k = 20$ and $k = 30$, where one of the identified impulse responses does not accurately describe the data. Both do not match the true impulse response well there, but the behavior of the impulse response identified with the IIRP penalty matrix is less spiky.

Now, the system considered for the identification is created randomly. Therefore,

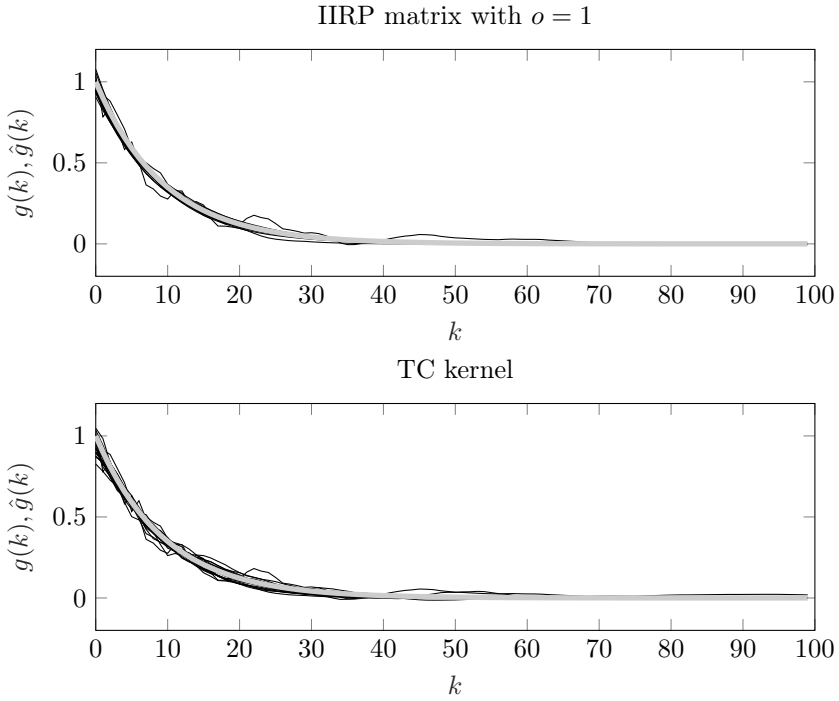

Figure 3.8: Impulse responses of ten different models for the experiment for the identification of random first order systems. The upper subfigure shows the results obtained by a first order IRP matrix penalty and the lower subfigure results obtained by the TC kernel.

the pole and gain are sampled from the uniform probability distributions

$$p \sim \mathcal{U}(p_{\min}, p_{\max}) \qquad K \sim \mathcal{U}(K_{\min}, K_{\max}) \tag{3.57}$$

with $p_{\min} = 0.05$, $p_{\max} = 0.95$, $K_{\max} = 10$ and $K_{\min} = 0.1$. From this probability distribution of transfer functions, 1 000 systems are sampled and disturbed with an SNR of 3 dB and 6 dB. Then the regularized FIR systems are identified. Afterwards, the NRMSE of the identified system is evaluated on noise free test data with $N_{\text{test}} = 1000$. The results for the NRMSE are shown in Fig. 3.9. As expected for this experiment, the first order IIRP penalty performs best, since the penalty matches the first order system exactly. It is interesting to notice that also the second order IIRP matrix is able to represent the data better than the TC kernel.

Second order system A similar experiment is done for second order systems. A second order system

$$G(z) = \frac{K(1-p)^2 z^2}{(z-p)^2} \tag{3.58}$$

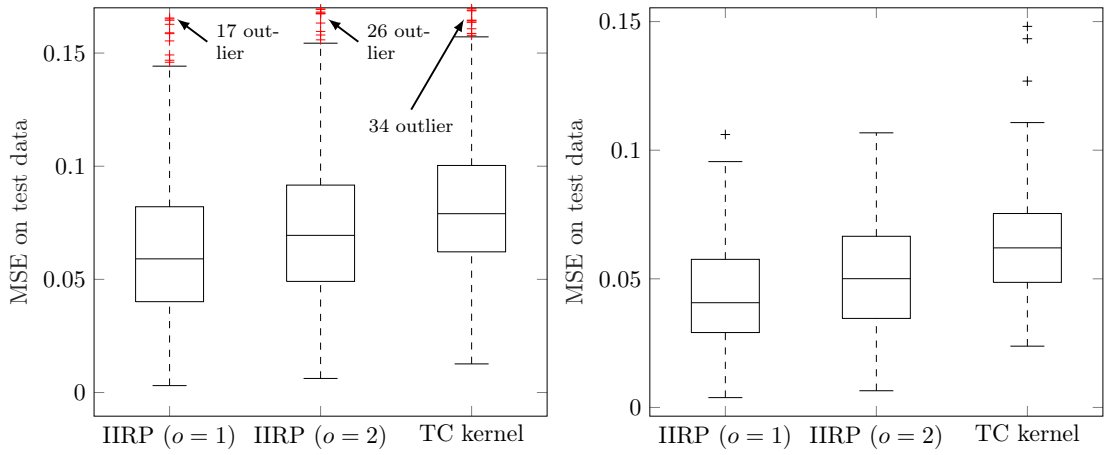

Figure 3.9: Results for 1000 realizations of the first order system. The left plot shows an SNR of 3 dB, while the right plot shows the result for an SNR of 6 dB.

is considered. The same excitation and disturbance as for the first order system are applied. Also, the identification procedure is kept the same, only an IIRP matrix of second order is employed. First, Fig. 3.10 shows the results for fixed pole and gain with different noise realizations at an SNR of 3 dB. It can be seen that the second order IIRP matrix method results in a significantly smoother behavior. In the figure, there is one system for which the identification is challenging due to the noise realization. By regularization, the behavior of the identified model improved significantly between $k = 40$ and $k = 50$ by showing a smoother response.

Now, again for the analysis of the effectiveness of prior knowledge incorporation, a Monte-Carlo experiment is conducted with randomly generated dynamic systems. Gain and double pole of the second order system are drawn from the same probability distribution as for the first order system. The results of the Monte-Carlo simulation, which are done under the same conditions as for the first order system, are shown in Fig. 3.11. It can be seen that for both investigated SNRs, the second order kernel performs significantly better than the TC kernel or the first order IRP matrix. The TC kernel and the first order IRP matrix method perform equally well. This result is as expected. The second order kernel contains the most prior knowledge for the sampled systems. From the two examples given here, it can be concluded that the IRP matrix method is able to incorporate sufficient prior knowledge very successfully, which is better tailored for the specific type of process.

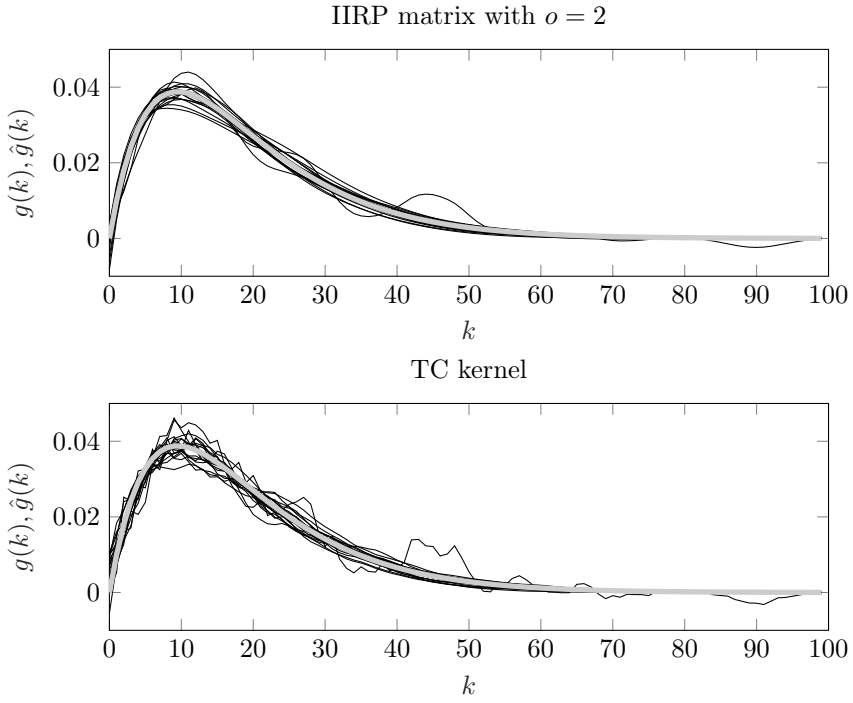

Figure 3.10: Results of the IRP matrix regularized FIR approach for the second order system

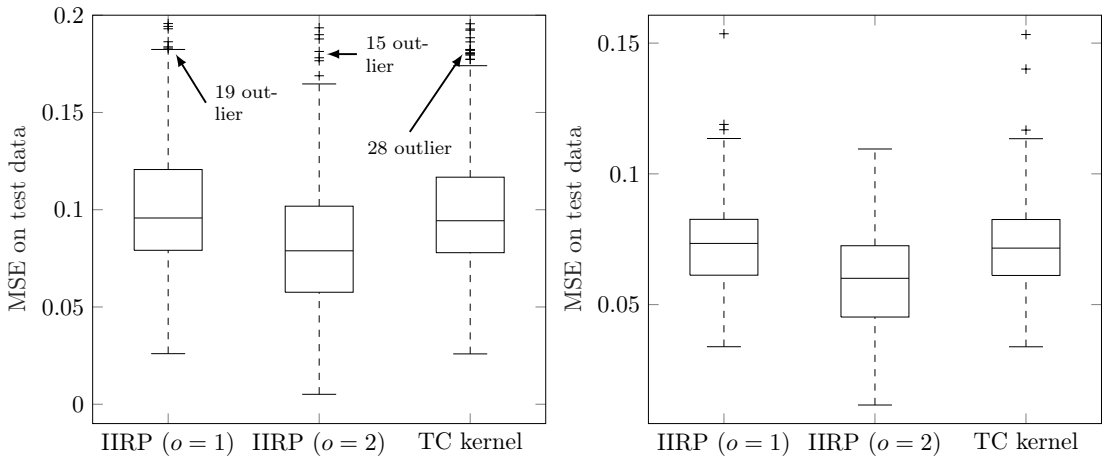

Figure 3.11: Results for 1000 realizations of the second order system. The left plot shows an SNR of 3 dB, while the right plot shows the result for an SNR of 6 dB.

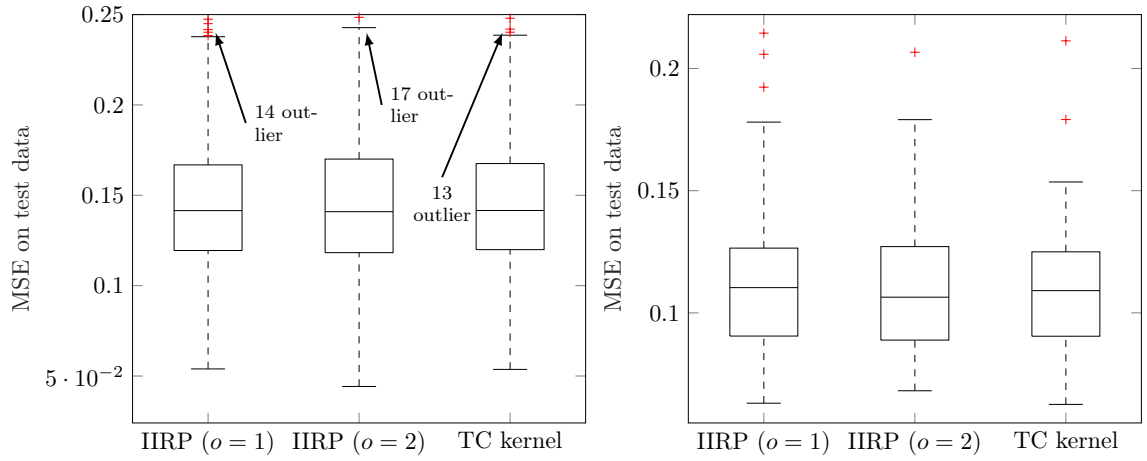

Figure 3.12: Results for 1000 realizations of the random dynamic system with random order. The left plot shows an SNR of 3 dB while the right plot shows the result for an SNR of 6 dB

3.4.2 Random Systems with Random Order

Finally, random systems with random order are investigated. The creation of the test datasets has been performed using the Matlab method rss. Here, random systems of order 20 are created. The sampling frequency is chosen to be three times the bandwidth of the systems. The slowest pole of the system is guaranteed to be $p < 0.95$. In Fig. 3.12, the results of the Monte-Carlo study are shown. Neither the TC kernel nor the IRP matrix has any advantage in the investigated systems. Nevertheless, all three ways of regularization perform similarly. This leads to the conclusion that IRP identification offers a very robust way to incorporate prior knowledge. For reasonably randomly generated systems, a loss in performance is hardly visible, although, for the second order IRP matrix, a system structure with two poles has been assumed, an assumption not met by most of the investigated systems. Note that the algorithm chooses itself how much prior knowledge is incorporated by determining the regularization strength with λ, which is chosen automatically by tuning the GCV error. This is the reason why these techniques are good choices for order selection and gray box identification, as will be elaborated on in the next sections.

3.5 Order Selection via Hyperparameter Tuning

Order selection is a challenging issue for traditional system identification algorithms. If the order is chosen poorly, the identification method can fail completely. Furthermore, the order and the location of poles of a system characterize its dynamic

properties. To generate reliable estimates of the pole locations for the investigated system provides some insight on its own. Classically, the model order is selected by complexity selection techniques, like AIC, described in detail in Sect. 2.3. Here, a novel option will be specified based on the addition of an elementary penalty term.

3.5.1 Robustness with Respect to Wrong IRP Matrices

An interesting property of the regularized FIR method with IRP matrices can be found if the behavior of the estimates is analyzed for wrong poles in the IRP matrix. The ability of a method to perform well, even if the model assumptions are not true, will be referred to as *robustness*. For the analysis, the transfer function

$$G(z) = \frac{0.5372z^2}{(z - 0.9e^{j\frac{\pi}{4}})(z - 0.9e^{-j\frac{\pi}{4}})} \tag{3.59}$$

is considered. The system has an oscillatory pole pair and unit gain. Now, four different IRP matrices, for identification, are considered. All matrices are weighted exponentially with $\alpha = 0.95$. The first and second ((a) and (b)) are first order IRP matrices with one pole at 5 and 0.95, respectively. Thus, IRP matrix (a) represents a first order unstable system and IRP matrix (b) a stable first order system. The third and fourth ((c) and (d)) are second order IRP matrices. The poles for the construction of (c) are $0.85e^{\pm j\frac{\pi}{4}}$, so that the oscillation of (c) is more damped than the oscillation of the true system. For (d), the poles are chosen to be equal to the true system as $0.9e^{\pm j\frac{\pi}{4}}$.

For the experiment, the system is excited with 1 000 samples of a PRBS signal. The output is disturbed by i.i.d. Gaussian noise such that the SNR is 5 dB. The identification results for the four different systems are shown in Fig. 3.13. All four identification results correspond quite appropriately to the true response. This behavior is explained by the automatic choice of the regularization strength λ. It is observed that the more inappropriate the prior knowledge, the lower λ is chosen. While for the IRP matrix with a pole at 5 (a), the GCV algorithm determines $\lambda = 0.226$, for the correct poles (d), $\lambda = 6.525 \cdot 10^4$ is found by GCV. This experiment explains the mechanism of the regularization. By the automatic tuning of λ, the algorithm itself determines how much prior knowledge is incorporated. Thus, the algorithm allows for two things, the incorporation of prior knowledge, and automatic conclusion whether this prior knowledge is appropriate. This explains its high robustness.

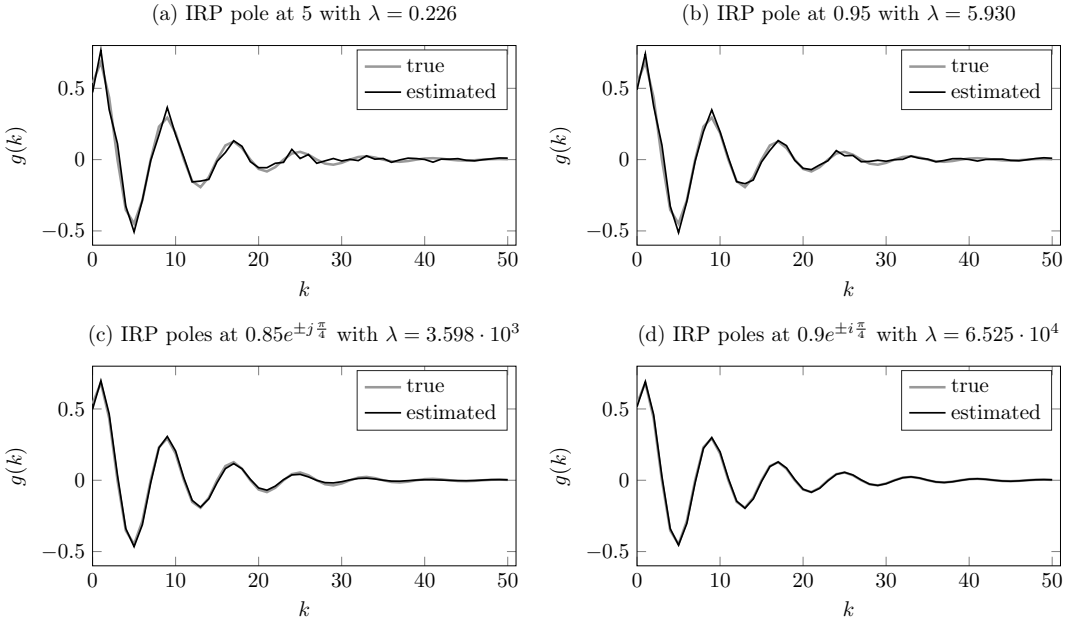

Figure 3.13: Results for regularized FIR impulse responses estimated with different IRP matrices. The true system has poles at $0.9e^{\pm j\frac{\pi}{4}}$.

3.5.2 Robustness in Comparison with OE

The practically most severe concern in system identification is an inappropriate choice of the model order. For other techniques, like OE identification, a wrong model order can deteriorate the identification result completely. Therefore, it is investigated how order-mismatch affects the performance of both the IRP matrices FIR approach and an OE model. To assess this difference in performance, a model with transfer function

$$G(z) = \frac{0.05372z^3}{(z-0.9)(z-0.9e^{i\frac{\pi}{4}})(z-0.9e^{-i\frac{\pi}{4}})} \tag{3.60}$$

is considered. This system has one real pole and an oscillatory pole pair. To identify this system, $N = 1000$ samples are generated. These samples are disturbed by i.i.d. Gaussian noise such that the SNR is 30 dB. This data is then used to identify an OE model and an FIR model with the IRP approach with prior of second and third order each. The procedure is repeated 200 times. For the demonstration of the robustness of the order selection, $\alpha = 1$ is chosen. The nonlinear optimization of the OE model is initialized with the identification result of an ARX model of the same order. The hyperparameter optimization of the regularized FIR model is initialized at $a_i = 0$, $i = 1, \ldots, o$ and $\lambda = 1000$, but is not very sensitive to the chosen initial values. The result of the optimization is shown in Fig. 3.14. On the

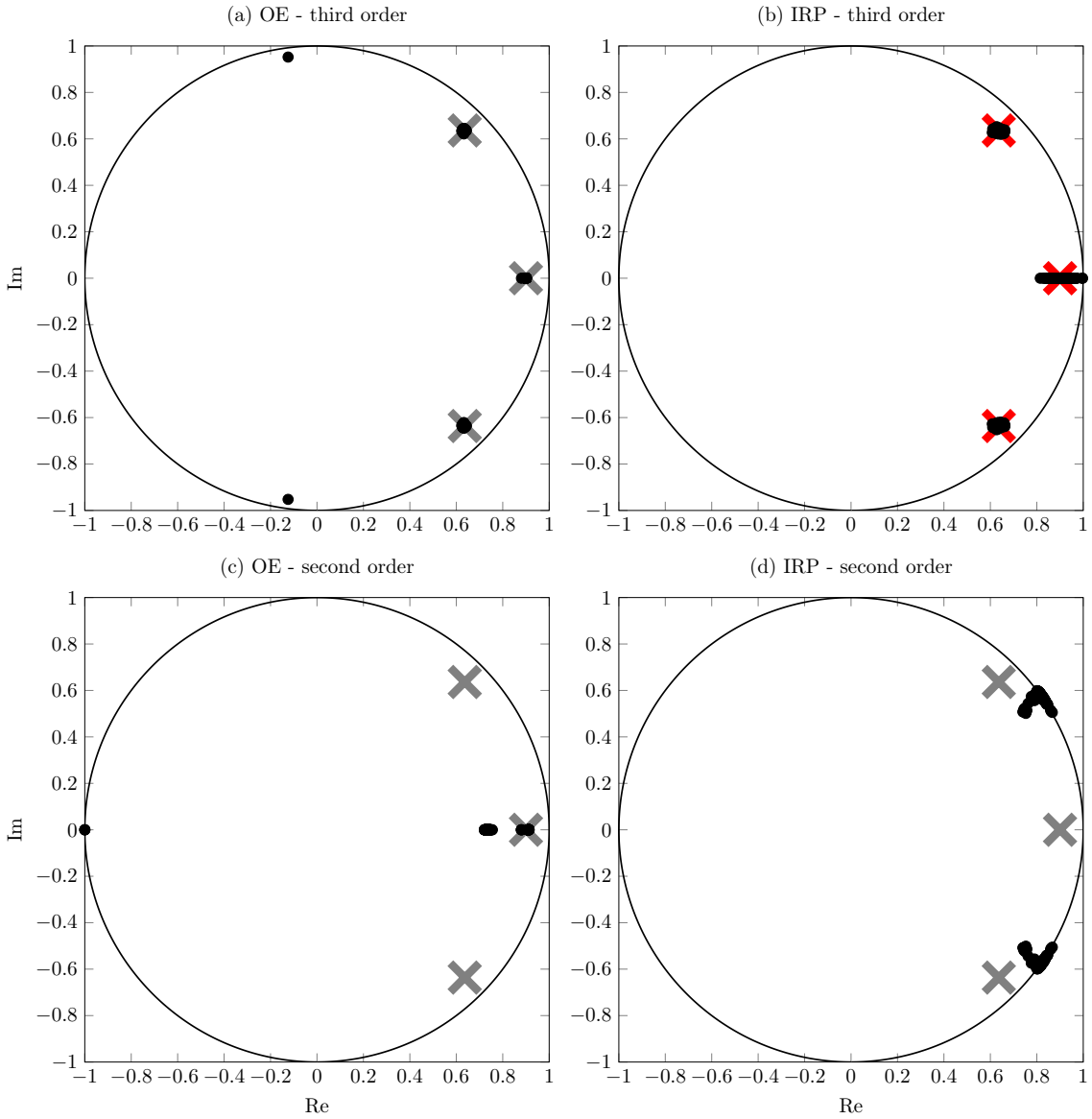

Figure 3.14: Comparison of pole location between OE and IRP FIR models of second and third order. In the order mismatch case, the IRP FIR model is able to represent oscillatory behavior while the OE model is not. crosses: true poles, dots: model poles

left side of the figure, subfigure (a) and (c), the true pole locations and the obtained poles for the OE system are shown. For the third order case, subfigure (a), the result shows that the true poles are matched very well. There is one exception. In one of the 200 cases, the poles are completely out of place. The reason for this is that OE solves a nonlinear optimization problem that can have local optima, which not necessarily correspond to the optimal global solution. As demonstrated in Sect. 3.5.1, the RFIR problem is significantly more robust due to the ability to assess the validity of the prior knowledge by choice of λ. For the order mismatch case shown in subfigure (c), it can be concluded that the OE model is unable to identify a system with oscillatory behavior. All poles identified have zero imaginary values. Also here solutions which correspond to local optima can be seen. For the FIR case with the correct model order, subfigure (b), the variance of the identified poles is more considerable. An advantage found empirically is that the algorithm does not converge to a local optimum for this case in the simulations. For the second order case, poles with imaginary parts only occur for the FIR case, subfigure (d). The fact that some of the poles found are not stable does not negatively influence the outcome of the regularized estimate. This is not problematic since the algorithm can choose λ small, if the prior is not appropriate, as discussed in Sect. 3.5.1.

It can now be concluded that IRP matrices are more robust to order mismatch, especially the choice of a too low order, than OE systems. This makes them a reliable alternative in practice.

3.5.3 Penalized Order Selection Procedure

Nevertheless, in a real-world application, an appropriate order for the penalty matrix must be selected. Here, a novel heuristic for the selection of the order IRP penalty construction is proposed. To assess the robustness of the regularized identification, it is proposed to penalize unstable poles p in the IRP FIR hyperparameter optimization with the additional term

$$J^{pen} = (1 - |p|)^2 \qquad \text{if } |p| > 1 \tag{3.61}$$

and non-oscillatory poles with negative real part with

$$J^{pen} = Re\{p\}^2 \qquad \text{if } \mathrm{Im}\{p\} = 0, \tag{3.62}$$

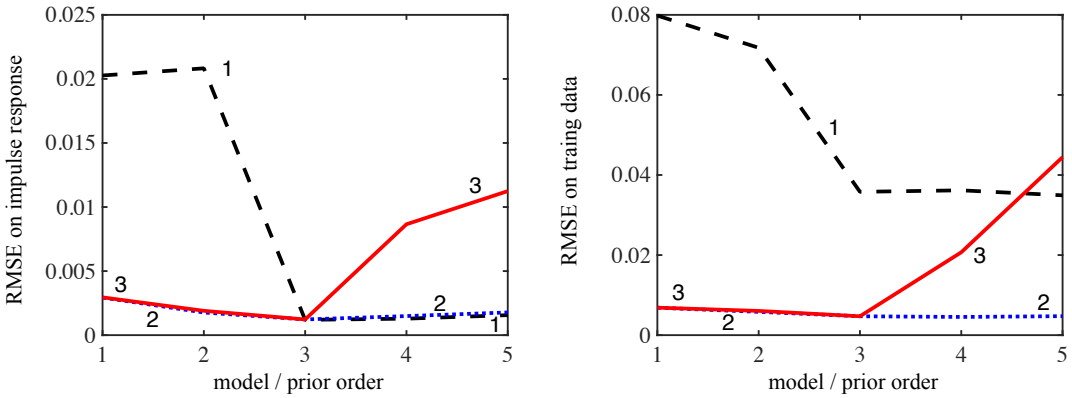

Figure 3.15: Results for the (penalized) error of different identified models for order selection. (1) OE model, (2) regularized FIR model, and (3) regularized FIR model with the proposed penalty

because they have no continuous time correspondence. This procedure is analyzed for a third order system. In Fig. 3.15, the NRMSE values of the output (left) and the sum of squared errors of the impulse response coefficients (right) for different identification with prior orders 1-5 are shown. The prior poles of the regularized FIR model were derived by an unconstrained optimization, while for the other (3), the penalties (3.61) and (3.62) were added. It can be seen that the performance of the OE model (1) deteriorates for low orders. The regularized FIR model (2), though, is, as discussed in the previous subsection, very robust to the correct choice of the model order. The penalized regularized FIR model (3) becomes significantly worse when the model order surpasses the process order 3. This behavior is due to the added penalties. None of the two other models, neither the OE nor the regularized FIR model, show such a significant performance decrease. The penalized regularized FIR identification technique with additional penalties can thus be robustly utilized for an accurate order selection procedure.

3.6 Regularized Identification of Gray Box Models

Gray box identification is encountered quite frequently in the system identification literature. In [13], it is argued that gray box identification always means to exploit a priori available knowledge about an object and experimental data. Under this perspective, all Bayesian methods employed for system identification are gray box techniques. Especially in a nonlinear setting, not considered in this chapter, the variety of modeling variants for systems is wide. Thus characterization between black and white models is involved, leading to a whole palette of gray models [70]. Nev-

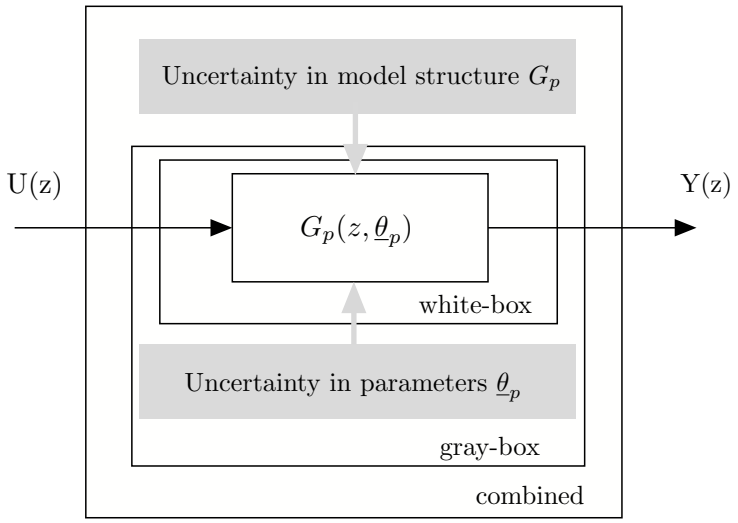

Figure 3.16: Structure of gray box models

ertheless, in this paragraph, gray box models refer to the specific situation when a physical linear model of the plant is available. Linearity is a restricting factor here, but these models often work well when a specific operation point is considered. The problem, as it is understood here, is illustrated in Fig. 3.16. If a white box model, this is a model generated by first-principles only, is considered, the parameters of this model will be fixed after a design phase. In this phase, it is possible to do measurements of parts of the system, like experimentally evaluating the weight of a component to determine a parameter of the equation of the system, but after that, the values remain set. Parameters or the structure of the model is not determined based on measured input or output values. Classical gray box identification works by estimation of the parameters of the system by estimating their values from input/output measurements. Statistically, this usually means that the parameters of the model are estimated by a maximum-likelihood approach with the assumption of i.i.d. Gaussian output noise. This leads to an output error formulation of the identification problem and results in a nonlinear optimization problem, for which it is not guaranteed that the global solution is found. A black box approach for linear systems is an unregularized FIR identification algorithm. It is, however, possible to take the uncertainty in the parameters of the system into account. For black box identification, the most flexible model is to be used. Since the IO-behavior of a linear system is uniquely characterized by its impulse response, identification of all impulse response coefficients is the most general approach. Whether a gray box or a black box model is preferable depends strongly on the data available from the system. If a large amount of informative data (sound system excitation, low noise

level) is available, the black box model has the advantage of being insensitive to inaccuracies made during the physical modeling process (e.g., neglection of higher dynamic modes). If there is only a low amount of data or less informative data available, the black box model is prone to overfitting. Impulse response preserving models can, as will be shown in this section, provide a loophole of this dilemma by combining classic gray box and the black box approach [84].

3.6.1 From State-Space to Impulse Response

We start by assuming that some physical model is available as a parameter depended state space model

$$\begin{aligned} \underline{x}_s(k+1) &= \underline{A}(\underline{\theta}_p)\, \underline{x}_s(k) + \underline{b}(\underline{\theta}_p)u(k) \\ y(k) &= \underline{c}^T(\underline{\theta}_p)\underline{x}_s(k) + d(\underline{\theta}_p)u(k). \end{aligned} \tag{3.63}$$

Here, the system parameters $\underline{A}(\underline{\theta}_p)$, $\underline{b}(\underline{\theta}_p)$, $\underline{c}(\underline{\theta}_p)$, $d(\underline{\theta}_p)$ depend (possibly nonlinearly) on the parameters $\underline{\theta}_p$. If a physical model of a mechatronic system is available this parameter $\underline{\theta}_p$ can be, for example, a mass, a stiffness, an inductivity, or any other unknown physical constant of the system. An overview of techniques for the construction of models from first principles can be found in [55]. It can be concluded from simple recursion that the impulse response of the system described by (3.63) is given by[67]

$$g_p(k) = \begin{cases} d(\underline{\theta}_p) & k = 0 \\ \underline{c}^T(\underline{\theta}_p)\underline{A}(\underline{\theta}_p)^{k-1}\underline{b}(\underline{\theta}_p) & k > 0. \end{cases} \tag{3.64}$$

This equation allows for each value of the physical parameters $\underline{\theta}_p$ to compute the corresponding impulse response. Furthermore, the transfer function of the input/output behavior of the state space system is given by

$$G(z) = \underline{c}^T(\underline{\theta}_p)\left(z\underline{I} - \underline{A}(\underline{\theta}_p)\right)^{-1}\underline{b}(\underline{\theta}_p) \tag{3.65}$$

Thus, a transfer function is assigned to each possible value of $\underline{\theta}_p$. This transfer function can then be employed for the construction of an impulse response preserving matrix of the same order as the state space representation. The computation of a value of the GCV function is summarized in Algorithm 3. Usually, this calculation of

Algorithm 3 Calculation of the GCV error for RFIR identification

1: Construct a linear state space system depending on physical parameters $\underline{\theta}_p$.
2: Construct the corresponding transfer function $G_p(z)$ by (3.65).
3: Build the impulse response preserving matrix with (3.33)
4: Calculate the GCV error, according to (2.94)

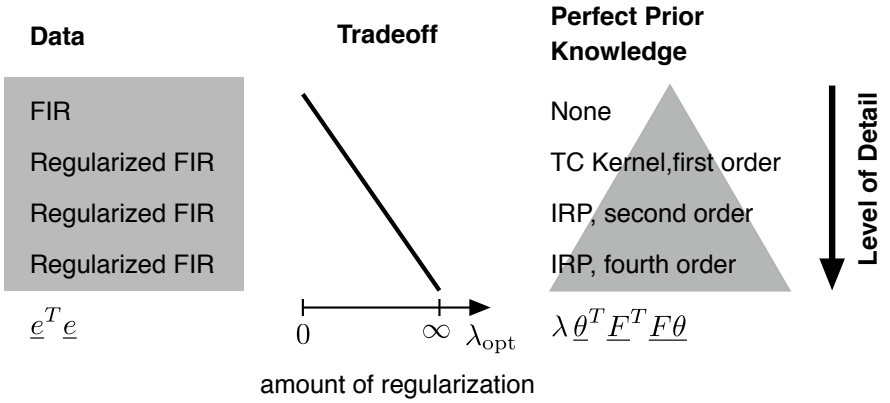

Figure 3.17: Possible trade-off between prior knowledge and available data for regularized system identification.

the GCV error is embedded in an optimization loop to find the best hyperparameters for the problem. If the optimal hyperparameters are found, the optimal model parameters are found by solving the regularized estimation problem.

3.6.2 The Appropriate Amount of Prior Knowledge

With the gray box approach, there are three types of parameters estimated by the GCV. These are the strength of the regularization λ, a parameter of the weighting matrix α, and the physically relevant parameters $\underline{\theta}_p$. The effect of the incorporation of prior knowledge is schematically depicted in Fig. 3.17. Here, it is shown that by using the GCV for hyperparameter estimation, the trade-off between data and a priori available knowledge is done using λ. The most extreme case is the consideration of an FIR model solely without any additional information. In this case, the estimation of the parameters of the impulse response is only determined by data, and prior knowledge has no influence. If prior knowledge is to be included, then the most basic option is to construct the penalty matrix with the TC kernel. In this case, λ cannot grow arbitrarily large since the full rank for $\underline{F}$ would cause all impulse response coefficients to be equal to zero, which is usually of no practical value. By using IRP matrices, it is possible to increase the level of detail of the prior knowledge. If this prior knowledge is, as will be the case for most physical models,

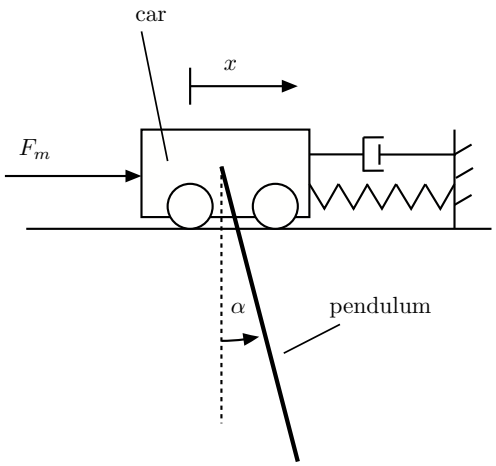

Figure 3.18: Structure of the experimental inverse pendulum problem.

accurate, the value of λ can be increased, and the hyperparameter tuning technique will allow for more regularization and so more prior knowledge for the solution of the identification problem. It is interesting to note that by this procedure, an automatic data-based trade-off between prior knowledge and information available in the data is done.

3.6.3 Pendulum Example

For the investigation of the applicability of the algorithm to a real-world system, a pendulum example is investigated. The structure of the experiment is shown in Fig. 3.18. A force $F(k)$ is applied to a car with the mass m_c. The movement of the car results in both, a change in the car's position $x(k)$, and a rotation of the pendulum with an angle $\alpha(k)$. The moment of inertia of the pendulum corresponding to its rotation point is denoted by J_p. The car is further attached to a spring with spring constant c_s and damping constant d. If the sampling time is chosen as $T_s = 20$ ms, then the following transfer function can be derived [84]

$$\frac{B(z)}{z^4 - 3.73z^3 + 5.27z^2 - 3.35z + 0.8}. \tag{3.66}$$

The IRP matrix is constructed using the denominator coefficients of the transfer function.

In an experiment, $N = 1000$ samples are collected while the system is being exited by a PRBS signal. The results obtained by the gray box method are shown in Fig. 3.19. To demonstrate the effect of the chosen IRP matrix, the value of λ is varied. The first figure shows the identification result for $\lambda = 0$. This means that

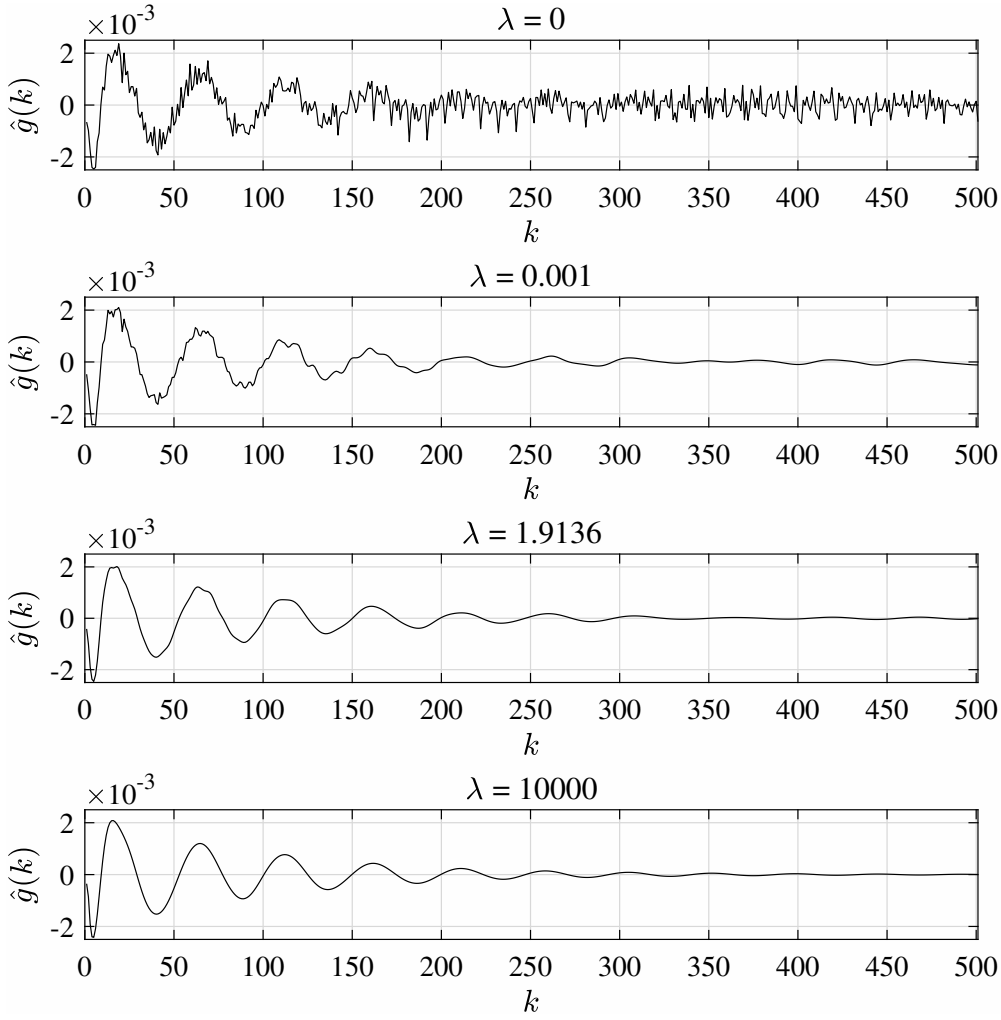

Figure 3.19: Obtained impulse responses for different values of λ.

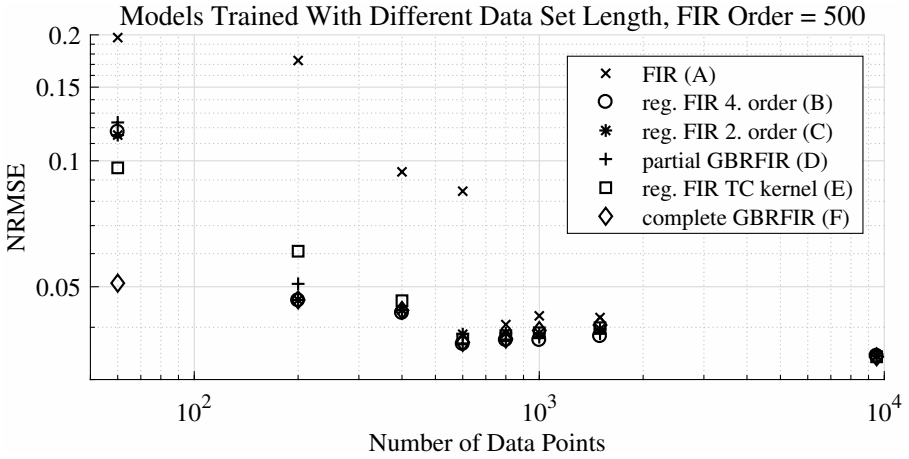

Figure 3.20: NRMSE for different penalty terms and different number of samples.

no prior knowledge is induced. Since the term $\lambda \underline{F}^T \underline{\theta F}$ is zero by definition, a classic FIR system is identified. The spiky response of this FIR model is clearly visible. This behavior does not only occur for the pendulum system but is typical for linear system responses identified by an FIR model. If λ is increased, which is shown in the next subfigure, the spikiness of the response starts to decrease. In this case, prior knowledge is induced in the system, and the response is fitted to be more like the model based on physical principles. The subfigure thereafter shows the value of λ optimized by GCV. The spikiness is almost gone. However, compared to the last subfigure, where λ has been chosen to be very high so that the response will correspond to a model with the given denominator coefficients, it can be seen that some coefficients of the impulse response are allowed to differ.

For a more detailed analysis of the trade-off between incorporated prior knowledge and model accuracy, both the length of the data and the used kernel are varied. To compare the state of the art methods an unregularized FIR, regularized IRP matrices of second and fourth order, the TC kernel, gray box kernel where only dampings are estimated (partial gray box), and a complete gray box where all 4 parameters are known from (3.66) are considered. The results are shown in Fig. 3.20. One observation that can be made is that for a high number of informative data, e.g., $N = 10\,000$, all methods perform almost similar. However, for less informative data, this picture changes. If very little data is available, the pure gray box model (F) performs best. This is relatively easy to explain. The number of hyperparameters to be identified for this approach is two, since only regularization strength and penalty parameter λ have to be estimated. For the partial gray box model (D), this number grows to four, and for the reg. IRP FIR (B) it grows to six. For a larger amount of data, the IRP matrices of second and fourth order become comparable. However,

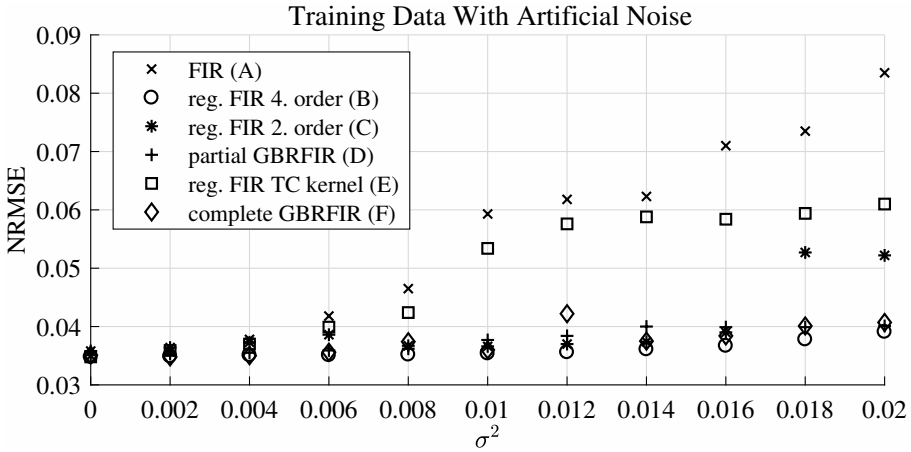

Figure 3.21: Performance for different penalty matrix approaches and varying noise levels.

it is remarkable that, for this example, IRP matrices perform better than the TC kernel for cases with an average number of data.

To evaluate the robustness of the method to noise, the obtained output data is corrupted by Gaussian white i.i.d noise with variance σ^2. The results are shown in Fig. 3.21. For low noise levels, the performance of the identification techniques is equal. If the noise is increased, the FIR models variance error increase, and so the performance on test data. Also, the performance of the TC kernel starts to deteriorate. For high noise levels, the performance of the IRP matrix of fourth order, the partial gray box model, and the complete gray box model are nearly equal. The proposed method is thus able to infer the gray box parameters accordingly with the GCV approach.

Concluding, it has been shown that IRP matrices combined with the regularized FIR identification approach allow for systematic integration of prior knowledge to linear system identification. A significant benefit compared to classical gray box identification techniques is the robustness of the method. Irrespective of the applied construction techniques for the penalty term in the regularization, if sufficient data is provided, all approaches are able to perform well. In the case where data is disturbed, an excellent gray box model can make a substantial difference.

4 Regularized Local FIR Model Networks

The ideas described in Chap. 3 apply for linear system identification only. Variance errors can, however, be much higher for nonlinear system identification. This is caused by the significant increase in the number of parameters, especially for the identification of nonlinear FIR systems. For the identification of linear FIR systems, each delayed input requires the estimation of one *parameter*. For NFIR systems, each delayed input increases the *dimension* of the input space by one, leading to a substantial increase in the number of parameters required for each input and its coupling with the other inputs.

The application of local model networks offers a loophole to this dilemma. Due to the separation of $\underline{z}$ and $\underline{x}$ variables described in Sect. 2.1.4, it is possible to keep the number of nonlinear dimensions small, while the local models are FIR models with many parameters. However, if the local models are of FIR type, the variance error remains an issue. In [80], initially, a possible solution to this problem is proposed. As described in [79], this can also be extended to MIMO systems. Furthermore, the approach has been applied to a simulated Diesel engine process [82].

In this chapter first, the structure of the proposed LMN with local FIR models is introduced. Then, the role of regularization, including the estimation of hyper-parameters, is addressed. The straightforward extension of the approach to MIMO systems is treated afterward. Finally, the feasibility of the procedure is demonstrated on two numerical examples, and the method is applied to a simulated Diesel engine process.

4.1 Structure of Local Linear Model Networks

The structure of the network corresponds to the classical LMN, as described in Sect. 2.1.4. In this approach, M local FIR models $L_i(\underline{x}(k))$ are weighted with validity

functions $\Phi_i(\underline{z}(k))$ to calculate the output

$$y(k) = \sum_{i=1}^{M} L_i(\underline{x}(k))\Phi_i(\underline{z}(k)). \tag{4.1}$$

To identify NFIR models, the local models $L_i(\underline{x}(k))$ are chosen as FIR models with an additional offset. The scheduling variables $\underline{z}(k)$ can be chosen differently for the generation of the nonlinear partitioning with the validity functions $\Phi_i(\underline{z}(k))$. In contrast to $\underline{x}(k)$, the variables $\underline{z}(k)$ are chosen to also contain delayed values of the output $y(k)$. This enables a better representation of the nonlinear behavior.

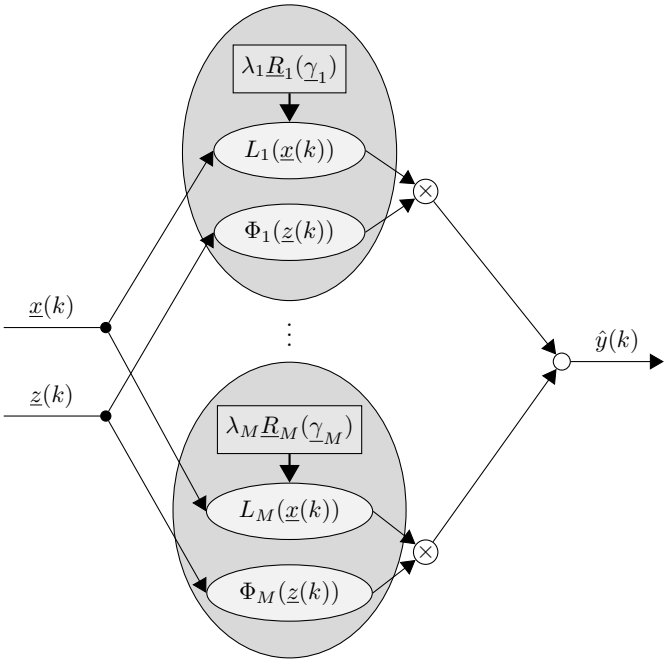

Figure 4.1: Structure of a regularized local FIR model network with M local models.

The proposed structure is summarized Fig. 4.1. Each neuron contains one validity function and a local FIR model. These FIR models are identified using a regularized approach. Therefore, the penalty matrix $R_i(\underline{\gamma}_i)$ regularizes the estimate of each local FIR model. The penalty term depends on the regularization strength λ_i and other hyperparameters $\underline{\gamma}_i$. This connection is indicated in the figure by the arrow pointing to the local models. The other hyperparameters contained in $\underline{\gamma}_i$ depend on the chosen construction mechanism for the penalty matrix. In principle, every approach described in Chap. 3 can be applied. Usually, the TC kernel is employed for its simplicity. So $\underline{\gamma}_i$ will be a scalar and contains only the exponential decay factor of the TC kernel.

4.1.1 Local FIR Models

To construct the local FIR models, the $n+1$-dimensional regressor variable $\underline{x}(k)$ contains only delayed input values and reads as

$$\underline{x}(k) = [u(k), u(k-1), \ldots, u(k-n)]^T . \tag{4.2}$$

The i-th local FIR model with offset can be written as

$$L_i(\underline{x}(k)) = \underline{b}_i^T \underline{x}(k) + c_i, \tag{4.3}$$

with the $n+1$-dimensional vector of FIR coefficients $\underline{b}_i$ for the i-th local model and the local offset c_i. The parameters of the local models are estimated by a local estimation approach, see Sect. 2.1.4 for the probabilistic interpretation. For the local estimation approach without regularization, the estimate is the solution to a weighted least squares problem. This is similar in the regularized case. Here, the local estimation approach leads to a weighted regularized least squares problem, which also has an analytical solution. The weighted regressor and the weighted output without offset for the i-th local model are calculated as

$$\underline{\tilde{X}}_i = \underline{W}_i^{\frac{1}{2}} \underline{X} \qquad\qquad \underline{\tilde{y}}_i = \underline{W}_i^{\frac{1}{2}} (\underline{y} - c_i). \tag{4.4}$$

To calculate $\underline{\tilde{y}}_i$, the offset c_i has to be available. The procedure to identify the offset is described in Sect. 4.2.3. The weighting matrices $\underline{W}_i$ are formed in the usual way for local model networks [90] according to

$$\underline{W}_i = \operatorname{diag}\left(\Phi_i(\underline{z}(1)), \Phi_i(\underline{z}(2)), \ldots, \Phi_i(\underline{z}(N))\right) \tag{4.5}$$

and the validity functions Φ_i are calculated according to (2.35). The transformation of the regressor matrix and the output vector given by (4.4) enables computation of the estimate for the parameters of the local models according to

$$\underline{\hat{b}}_i = \left(\underline{\tilde{X}}_i^T \underline{\tilde{X}}_i + \lambda_i \underline{R}_i\right)^{-1} \underline{\tilde{X}}_i^T \underline{\tilde{y}}_i. \tag{4.6}$$

Here, the matrix $\underline{R}_i$ serves as a penalty matrix for the regularization of the i-th local model. Consequently, local model networks can be employed to transfer favorable identification methods for linear systems to the nonlinear world by using the local estimation technique. This also has been demonstrated for OBFs [53] and for instrumental variable methods [90].

4.1.2 Choice of the Partitioning Space in Local Model Networks

A characteristic of LMNs is that they blend different models depending on the scheduling variables $\underline{z}$. Blending local FIR models is unique compared to the peculiar blending behavior of local ARX models. First, this difference is illustrated. Afterwards, different possible choices for the scheduling variables and the consequences for the LMN will be analyzed.

Blending characteristics To explain the advantages of the blending behavior of local FIR models, their blending behavior is compared with local ARX models. The blending behavior of local ARX models has been investigated in [90]. There, it has been shown that blending can cause undesired oscillatory and even unstable behavior in the blending regime. To illustrate these issues and compare them to the FIR case, we consider the two systems

$$G_1(z) = \frac{z^4}{(z-0.9)^2(z-0.8)^2} = \frac{z^4}{A_1(z)} \tag{4.7}$$

and

$$G_2(z) = \frac{z^4}{(z-0.25)^2(z-0.35)^2} = \frac{z^4}{A_2(z)}. \tag{4.8}$$

The first subsystem has relatively slow poles $p_{1,2} = 0.9$ and $p_{3,4} = 0.8$. In contrast, the second subsystem has relatively fast poles at $p_{1,2} = 0.25$ and $p_{3,4} = 0.35$. The denominator polynomial

$$A(z) = \zeta A_1(z) + (1-\zeta)A_2(z) \tag{4.9}$$

describes the blending behavior for the ARX case with $\zeta = 0 \ldots 1$ being the blending factor. Here, the coefficients of the denominator polynomial are blended, which changes the poles of the system during blending. In contrast to the FIR case, the coefficients of the impulse response are simply blended. Thus the impulse response reads as

$$g(k) = \zeta g_1(k) + (1-\zeta)g_2(k) \tag{4.10}$$

with $g_1(k)$ and $g_2(k)$ denoting the impulse responses of $G_1(z)$ and $G_2(z)$.

In Fig. 4.2 the impulse responses of the blended ARX systems are shown in the left subfigure, while in the right subfigure the blended FIR impulse responses are shown. The value of ζ is subsequently decreased. For $\zeta = 1$, and for $\zeta = 0$, the impulse responses for $G_1(z)$ and $G_2(z)$ are obtained for both the ARX and the FIR case.

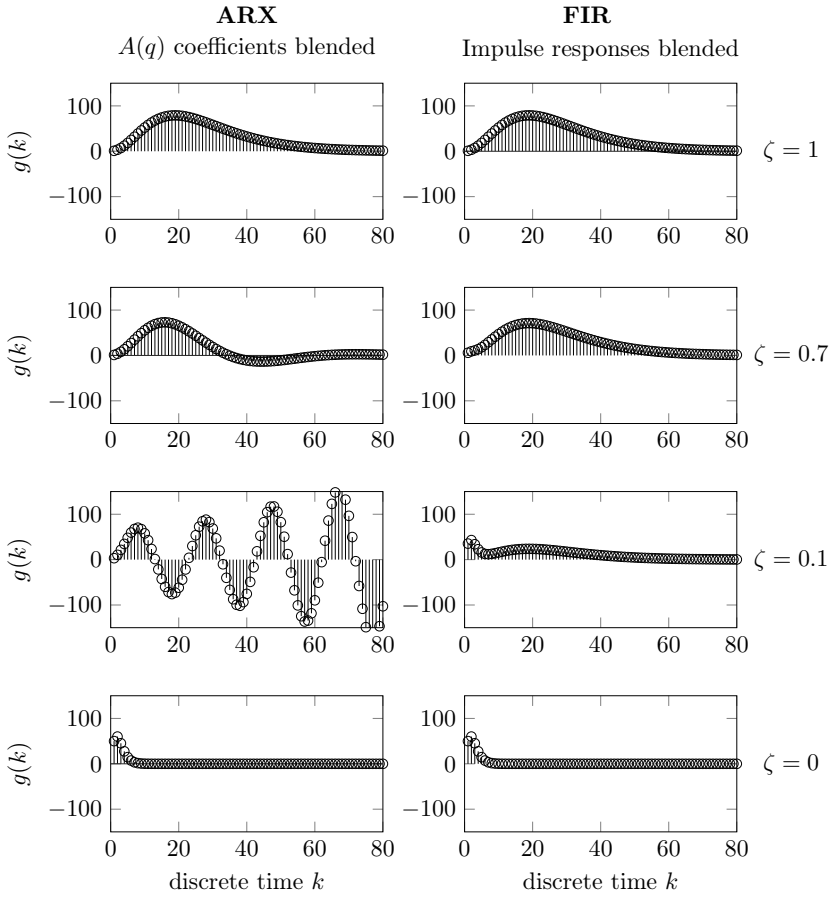

Figure 4.2: Blending behavior of a fourth order system with $p_{1,2} = 0.25$, $p_{3,4} = 0.35$ blended to $p_{1,2} = 0.8$ and $p_{3,4} = 0.9$ for different blending factors ζ.

The difference between blending of local ARX and local FIR models can be seen for the values of ζ in between. While for the FIR case, the blending is done smoothly, the impulse responses for blended ARX systems behave awkwardly. For $\zeta = 0.7$, the ARX impulse response becomes negative for values around $k = 40$, although both the impulse response of $G_1(z)$ and $G_2(z)$ are positive. But in particular, for $\zeta = 0.1$, the impulse response of the blended ARX system becomes oscillatory and unstable.

To explain the blending behavior of the ARX system, in Fig. 4.3 different pole locations for ζ between 0 and 1 are depicted. It can be seen that for all values of ζ except for the cases when ζ is exactly 0 or 1, the poles of the blended systems have an imaginary part. Furthermore, it can be seen that the fastest poles are blended to the slowest poles. This creates two ellipsoidal routes in the z-plane for blending, which are symmetric to the imaginary axis.

The choice of the splitting variables $\underline{z}$ significantly influences the ability to represent

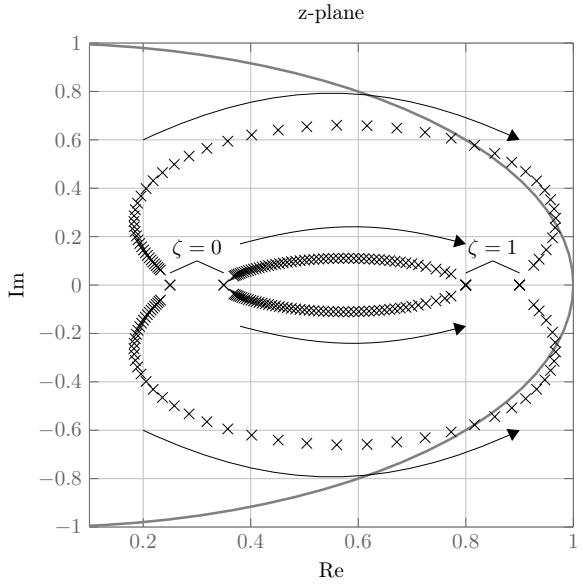

Figure 4.3: Pole locations of the blended local ARX models. For values of ζ between 0 and 1, each cross represents the location of one pole. The directions of the arrows indicate increasing values of ζ.

nonlinearities with the local model network. Several options are possible. The alternative, which is most straightforward for FIR models, is to consider only delayed input values for the vector $\underline{z}$. Another approach is to consider also delayed output values. These two cases are discussed in detail.

Delayed values of the input If only delayed input values are contained within $\underline{z}$, it is guaranteed that for a constant input $u(k) = u_0$, the output will be constant as well. If this case is considered, the splitting variable is equal to a constant $\underline{z}(k) = \underline{z}_0$ which contains only u_0. So, for each validity function, $\Phi_i(\underline{z}(k)) = \Phi_i(\underline{z}_0)$ holds. Thus, the validity functions will be constant as well. Using the constant input in the equation for the local models leads to

$$L_i(\underline{x}(k)) = L_i(\underline{x}_0) = \sum_{i=1}^{M} \sum_{j=0}^{N} b_i(j) u_0 + c_i, \tag{4.11}$$

which is also a constant. So the total output of all local models will be constant, too.

Delayed values of the output For delayed output values, this guarantee is lost. It can happen that the system is in a limit cycle when the input is constant. However, the output will remain bounded for an arbitrarily bounded input. First, assume that

the input is bounded by $u(k) \leq u_0$, then it holds that

$$y(k) \leq \sum_{i=1}^{M} \left(\sum_{j=0}^{n} b_i(j) u_0 + c_i \right) \Phi_i(\underline{z}(k)) \leq \max_i \left(\sum_{j=0}^{n} b_i(j) u_0 + c_i \right) = \text{const.} \quad (4.12)$$

The second inequality holds, due to the fact that all validity functions sum to one. Besides guaranteeing that the input is bounded, it is not guaranteed that it remains constant. It is possible that the system is in a limit cycle, but remain bounded. It is often advantageous to accept this slight risk and increase the flexibility of the model by including delayed output values.

4.2 Role of the Regularization Matrix

As described in detail in Chap. 3, identification of linear FIR systems is prone to a high variance error. If local FIR models are considered, this problem becomes significantly worse. The variance error increases since instead of the $n+1$ parameters of one n-th order FIR system $M(n+2)$ parameters of the FIR models with offset have to be identified. Controlling this variance error by a suitable regularization is thus the key for enabling well performing nonlinear identification with LMNs.

4.2.1 Choice of the Kernel or Penalty Matrix

To perform the identification routine described in Sect. 4.1.1, the local FIR models are affected by the penalty matrices in the same way as one global regularized FIR model. The major difference is that not one single regularization matrix, but as many regularization matrices as local models have to be chosen. In principle, the penalty matrix could be constructed by hand following an individual scheme for each local model. However, to enable the usage of regularized identification in an automatic local model construction algorithm (like LOLIMOT), an automatic procedure to choose the penalty matrices is required. Usually, the penalty matrices for the local models can be constructed from the same kernel or filter matrix. So, the penalty matrix for each local model is, e.g., constructed by the TC kernel. If such a scheme is employed, there are as many hyperparameters for each local model to choose, as there are hyperparameters in the construction scheme for the chosen penalty matrix. Here, different choices are possible, as well. One viable option is to estimate the local models with the same hyperparameters specified a priori, stemming, e.g., from a global regularized FIR model. For the TC kernel, this means that each local model has the same decay rate. This has the drawback that time constants which differ

between local models cannot be represented well. Alternatively, the local models can be estimated with individually tuned hyperparameters. This increases the number of unregularized parameters within the model but allows for more flexibility to capture varying dynamic behavior at different operation points.

It is, in principle, possible to integrate more complex prior knowledge, like the IRP kernel described in Chap. 3. This increases the number of hyperparameters for each local model. The risk here is, however, that if a scheme with very many hyperparameters is used, there is also a risk to overfit the hyperparameters. For the avoidance of too many parameters, the TC kernel is, therefore, a viable choice. For each local model with marginal likelihood optimization, three additional unregularized parameters are estimated. For the i-th local model, these are the measurement noise σ_i, the decay parameter α_i of the TC kernel, and the regularization strength λ_i.

4.2.2 The Marginal Likelihood

Tuning of the hyperparameters of the kernel for the local FIR models is critical for a high accuracy of the identification result. In principle, the marginal likelihood, as in the linear case, can be employed. There are, however, two differences. The first difference is that instead of a global model, the likelihood is only estimated for the output of the local model. The second difference is that an offset does not occur for linear identification. For the local FIR models, this offset is not penalized. This has the same consequence as for other unpenalized parameters, as described in Sect. 3.1.2. The influence of such non-Bayesian parameters is eliminated by the subtraction of their influence from the output. For the offset, this simply means that the offset c_i is subtracted from the local output. For a mathematical treatment of the likelihood, this means that instead of the pdf $p(\underline{y})$ for the global output, the pdf for the local offset corrected output $p(\underline{\tilde{y}}_i)$ is calculated. This pdf can be calculated according to

$$p(\underline{\tilde{y}}_i) = \frac{1}{\sqrt{(2\pi)^N \det \underline{Z}_i}} \exp\left(-\frac{1}{2}\underline{\tilde{y}}_i^T \underline{Z}_i^{-1} \underline{\tilde{y}}_i\right) \tag{4.13}$$

with

$$\underline{Z}_i = \frac{1}{\lambda_i} \underline{\tilde{X}}_i \underline{P}_i \underline{\tilde{X}}_i^T + \sigma_i^2 \underline{I}_N. \tag{4.14}$$

This is the likelihood function. Its logarithm is optimized. Thus the hyperparameter optimization problem can be formulated as

$$\underset{\underline{\gamma}_i, \lambda_i, \sigma_i}{\text{minimize}} \; \underline{\tilde{y}}_i^T \underline{Z}_i^{-1} \underline{\tilde{y}}_i + \log\det \underline{Z}_i. \tag{4.15}$$

The computation of the marginal likelihood is described in detail in Sect. 4.3.1.

4.2.3 Estimation of Offset and Noise

In contrast to the other parameters of the local impulse responses $\underline{b}_i$, the offset c_i is not penalized by the regularization matrix. In principle, a bias space, as introduced in Sect. 3.1.2, has to be considered. This requires for each computation of the likelihood to estimate the parameter c_i from the solution of a regularized FIR problem. Then this offset is subtracted from the output. Afterwards, with this offset corrected output, the marginal likelihood is estimated.

In [100], it has been described that a two-step procedure for the estimation of the noise is favorable for the linear FIR identification problem. For this procedure, in the first step, a high order ARX or FIR model is identified, and the variance of the error serves as an estimate for the noise variance. The same approach can be applied for the estimation of the noise variance in the nonlinear case. Instead of a global ARX or FIR model, a local FIR or ARX model, equipped with an additional offset term c_i, is estimated. This allows for both the identification of the offset c_i and the noise variance σ_i^2. Also in the nonlinear case, this procedure has been observed to be sufficient since the found offset and noise do not differ significantly from those computed by a regularized estimation of the bias space.

The error made by a local model is different from the error in the linear case. The error of the local model is the sum of two terms. The first term stems from the noise itself, and the second is due to the nonlinearity which cannot be described by the linear structure in the validity region. This second contribution is also the cause of the LOLIMOT algorithm to split local models with the highest local error greedily. For regularized identification, a larger estimated error will via σ_i^2 influence the result of the identification such that the penalty is increased. The higher the penalty term, the more emphasis is put on prior knowledge, and less attention is given to obtain a fit from the data. This, in turn, causes an even higher error on training data in the areas of strong nonlinearity. The LOLIMOT algorithm is thus rewarded for splitting this region further. So a noise estimate tuned by this procedure is not expected to and has not been observed to interfere with the LOLIMOT algorithm negatively.

4.3 Algorithmic Implementation

In Algorithm 4, the method for the identification of a regularized FIR LMN is summarized. To control the overall complexity of the local model, an information criterion

Algorithm 4 Identification of a regularized local FIR LMN

1: Select variables and model order for the $\underline{z}$ regressor.
2: **loop**
3: Select the worst local model for splitting according to the LOLIMOT algorithm
4: **for** each possible splitting direction **do**
5: Identify an unregularized high order FIR model and estimate the offset c_i and the noise variance σ_i^2.
6: Tune the hyperparameters λ_i and $\underline{\gamma}_i$ for the penalty matrix $\underline{P}_i(\underline{\gamma}_i)$ by nonlinear optimization of (4.15).
7: Estimate the local FIR parameters $\hat{\underline{b}}_i$ by computing (4.6) with the optimized hyperparameters.
8: Compute the effective number of parameters df_i utilized for the local model estimation and calculate an information criterion.
9: **end for**
10: Split in the direction which results in the highest improvement of the information criterion.
11: **end loop**

is applied. The $\mathrm{AIC_c}$ with the effective number of degrees-of-freedom for the number of parameters will be employed in the subsequent analysis.

There are two sources of model complexity for LMNs with regularized local FIR models. The first source stems from the parameters of the local FIR model. This source of complexity is controlled by the described regularization, which is tuned by an appropriate hyperparameter tuning method. The second source of complexity stems from the number of local models. The more local models are within the LMN, the better the nonlinear behavior of the system is described. As, due to splitting, the number of data points lying within a local model is decreased, it can be expected that the amount of regularization for this model will be higher due to less available information in the data. So decreasing complexity caused by regularization and increasing complexity caused by the number of local models interact. An appropriate way to assess this interaction is the utilization of the effective number of degrees of freedom for the calculation of the $\mathrm{AIC_c}$.

4.3.1 Efficient Implementation of Hyperparameter Tuning

During the estimation of the local models, the hyperparameters of the kernel matrix $\underline{\gamma}_i$ and the regularization strength λ_i have to be tuned. For the linear case, estimation of the likelihood function poses a computational challenge since, in its fundamental form, it contains the inverse of an $N \times N$ matrix, see Sect. 2.2.2. For the unweighted linear case, an algorithm avoiding this explicit inversion has been proposed [26]. It will be shown that this algorithm generalizes to the weighted and nonlinear case as well. For the hyperparameter tuning, the weighted version of the hyperparameter function is considered. First, as for the linear case [26], the Cholesky factorization of the covariance matrix $\underline{P}_i = \underline{L}_i^T \underline{L}_i$ is computed. The matrix $\underline{L}_i$ is a lower triangular matrix. Then, the skinny QR factorization of

$$\begin{pmatrix} \underline{\tilde{X}}_i \underline{L}_i & \underline{\tilde{y}} \\ \underline{I}_n \sigma^2 & \underline{0} \end{pmatrix} = \begin{pmatrix} \underline{Q}_1 & \underline{q}_2 \end{pmatrix} \begin{pmatrix} \underline{R}_1 & \underline{r}_2 \\ \underline{0} & r \end{pmatrix} \tag{4.16}$$

is calculated. The matrix $\underline{Q}_1$ is an orthogonal $N + n \times n + 1$ and $\underline{R}_1$ an upper triangular $n+1 \times n+1$ matrix. It is then possible, see [26], to calculate the negative log likelihood function according to

$$J = \frac{r^2}{\sigma^2} + (N - n) \log \sigma^2 + 2 \log \det \underline{R}_1. \tag{4.17}$$

Since $\underline{R}_1$ is an upper triangular matrix, the determinant is simply the product of its diagonal elements. Thus, its computation is not demanding. The function J is optimized by a nonlinear optimization algorithm. In this thesis, the constrained optimization algorithm `fmincon` from MATLAB is employed. For the TC kernel, the value of α_i is constrained to lie between 0.85 and 0.98 and λ_i to be positive.

4.3.2 Efficient Computation of the Number of Parameters

For calculation of the $\mathrm{AIC_c}$ of the LMN, the effective degrees of freedom for each local model have to be calculated. Here, the QR factorization described by (4.16) can be reused. Let $\mathrm{tr}(\cdot)$ denote the trace of a matrix. Then, for the i-th local model,

the number of effective degrees of freedom df_i is calculated according to

$$\begin{aligned} df_i &= \mathrm{tr}\left(\underline{\tilde{X}}_i(\underline{P}_i\,\underline{\tilde{X}}_i^T\,\underline{\tilde{X}}_i + \sigma^2\underline{I}_n)^{-1}\underline{P}_i\underline{\tilde{X}}_i{}^T\right) \\ &= \mathrm{tr}\left(\underline{\tilde{X}}_i(\underline{L}_i\underline{L}_i^T\,\underline{\tilde{X}}_i{}^T\,\underline{\tilde{X}}_i + \sigma^2\underline{I}_n)^{-1}\underline{L}_i^T\underline{L}_i\,\underline{\tilde{X}}_i{}^T\right) \\ &= \mathrm{tr}\left(\underline{\tilde{X}}_i\underline{L}_i(\underline{R}_1^T\underline{R}_1)^{-1}\underline{L}_i^{-1}\underline{L}_i\,\underline{L}_i^T\underline{\tilde{X}}_i{}^T\right) \\ &= \mathrm{tr}\left(\underline{Q}_1\underline{R}_1(\underline{R}_1^T\underline{R}_1)^{-1}(\underline{Q}_1\underline{R}_1)^T\right) \\ &= \mathrm{tr}(\underline{Q}_1\underline{Q}_1^T) \end{aligned} \tag{4.18}$$

This form is significantly simpler than a direct calculation of the trace of the smoothing matrix. The matrix $\underline{Q}_1$ has already been computed for the calculation of the marginal likelihood, and thus for the computation of the information criterion, only the matrix multiplication of $\underline{Q}_1\underline{Q}_1^T$ has to be performed.

4.4 Multiple Inputs and Multiple Outputs

Up to now, only the SISO case for identification has been considered. The algorithm proposed here extends naturally to MISO and MIMO systems. The regressor variable has to contain the delayed values of all inputs if the system has multiple input signals.

4.4.1 Consideration of Multiple Inputs

If multiple inputs are considered for the RFIR LMN, in principle, the structure of the model remains the same. The only difference is that the local models become MISO FIR instead of SISO FIR models. This change is illustrated in Fig. 4.4. Here, for $m = 2$ inputs, it is shown that both inputs are fed to a tap delay line before being applied to the LMN as a nonlinear function approximator. The delayed output is, as in the SISO case, contained only in $\underline{z}$. For estimation of the local MISO FIR models, the regressors for the individual inputs are denoted as $\underline{X}_{u_1}, \ldots, \underline{X}_{u_m}$ with m indicating the number of inputs to the system and $\underline{X}_{u_i}$ denoting the regression matrix to model the FIR system of input $u_i(k)$. These regressors are then stacked to a common regressor

$$\underline{X} = \left[\underline{X}_{u_1} \quad \ldots \quad \underline{X}_{u_m}.\right]. \tag{4.19}$$

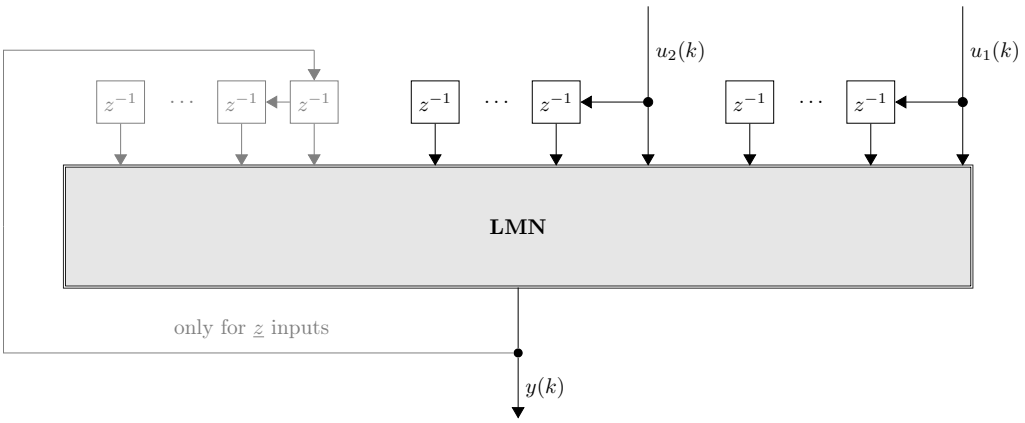

Figure 4.4: Structure of a regularized FIR LOLIMOT model with two inputs

Besides the regressor, also the local regularization matrix has to be changed. Therefore, individual regularization matrices are concatenated diagonally into a matrix

$$\underline{P} = \begin{bmatrix} \underline{P}_1 & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{P}_2 & \dots & \underline{0} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0} & \underline{0} & \dots & \underline{P}_m \end{bmatrix}. \tag{4.20}$$

The matrices $\underline{P}_1$, $\underline{P}_2, \dots, \underline{P}_m$ are individual kernel matrices for the inputs $u_1(k)$, $u_2(k), \dots, u_m(k)$. For the estimation procedure, nothing changes. The regressors obtained by stacking the individual regressors are weighted according to (4.4). Then the estimation of the parameters is processed as usual. A difference is that the number of hyperparameters for the local model changes, too. Now, instead of the hyperparameters for one regularization matrix, the hyperparameters of m regularization matrices have to be tuned. Both marginal likelihood and GCV can be employed to do this. In principle, it is possible to assign identical hyperparameters to the penalty matrices of different inputs. This would cause, e.g., in the case of the TC kernel, the dynamics of the individual FIR models for the inputs to be alike. Setting the parameters to be equal can be useful if it is known a priori that the inputs affect the output with a similar dynamic. If this is not known, different regularization parameters for each input are advantageous.

4.4.2 Consideration of Multiple Outputs

For the consideration of multiple outputs, two approaches are possible. The difference between these approaches is the partitioning of the input space. The first

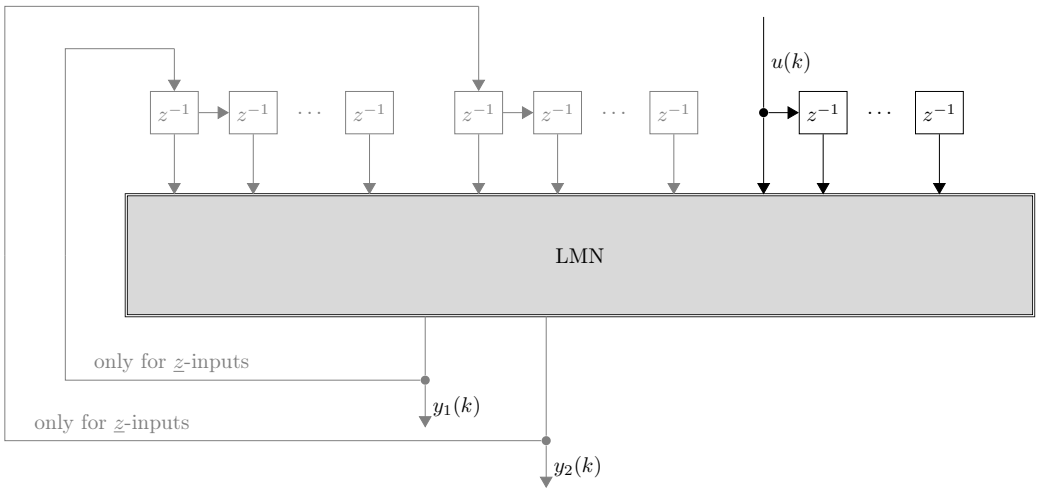

Figure 4.5: Structure of a regularized FIR LOLIMOT model with two outputs and shared $\underline{z}$-input spaces.

possibility is to train an individual RFIR LMN for each output. Here, nothing changes compared to the MISO or SISO case. Indeed, no knowledge of the nonlinear behavior of the different outputs is shared. The second option allows for a simple sharing of nonlinear behavior between the models for the different outputs. For this method, the nonlinear partitioning of the matrix $\underline{z}$ is shared. This approach is illustrated in Fig. 4.5. It can be seen that both output values $y_1(k)$ and $y_2(k)$ are fed back over a tap delay line into the nonlinear approximator. These are both colored in gray to underline that only $\underline{z}$ contains the delayed output values. The input space for the local linear models is the FIR input space which is only spanned by delayed values of $u(k)$.

4.5 Numerical Examples

For an illustration of the behavior and the advantages of the RFIR LMN, two nonlinear test systems are investigated.

4.5.1 SISO Wiener System

To illustrate the advantages of the regularized estimation, a Wiener system, described in [90], is investigated. The difference equation

$$\begin{aligned} y(k) = \arctan(&0.01867u(k-1) + 0.01746u(k-2)) \\ &+ 1.7826\tan(y(k-1)) - 0.8187\tan(y(k-2))) \end{aligned} \tag{4.21}$$

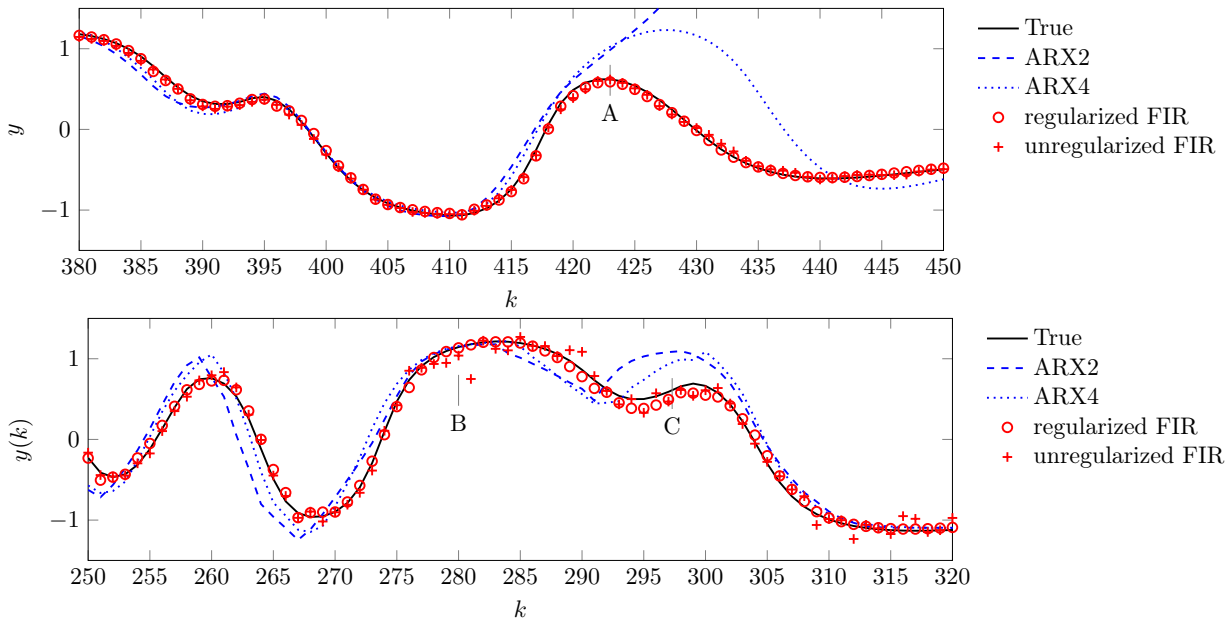

Figure 4.6: Excerpt of the obtained responses of the system for the low (upper plot) and the high noise (lower plot) case. ARX2 and ARX4 correspond to the local ARX model of second and forth order. The ARX2 model becomes unstable at A and performs poorly at C. The variance error of the non-regularized local FIR model becomes visible at B.

describes its behavior. For this system, $N = 1\,000$ samples have been simulated and disturbed by i.i.d. Gaussian noise. Two noise cases are investigated. The high noise case has an SNR of 40 dB and the low noise case one of 20 dB. The chosen input signal is an APRBS signal with a holding time of five samples. The regularized FIR LMN is trained by Algorithm 4 for both cases. Therein, the complexity of the model is determined by the $\mathrm{AIC_c}$. For comparison, LMNs trained by LOLIMOT with local unregularized FIR models, as well as local ARX models of second and fourth order, have also been identified. An excerpt from the time domain behavior of the identified models is shown in Fig. 4.6. It can be seen that the LMN with regularized local FIR local models shows the best responses for both the low and the high noise case. The local ARX model suffers from the inconsistency of the identification procedure. In the low noise case, the ARX model with two delayed input values becomes unstable (letter A in the figure), while the ARX model with four models performs highly undesirable (letter C). In the high noise case, it can be seen that local non-regularized FIR models suffer from high variance errors (letter B in the figure). The response becomes inaccurate for some samples, and the error changes between the samples at a high rate. The cause for this error is the variance error of the FIR coefficients of the local models. For the local regularized local FIR models, this does not occur.

Here, a change in the subsequent impulse response coefficients of the local models is penalized by the regularization. Thus, rapid changes between these coefficients occur less frequently. Another important reason for the improved performance of the regularized FIR compared to the unregularized one is the fact that per local model, fewer parameters are needed. Thus the regularized FIR LMN can split the input space more frequently, allowing for a better representation of the nonlinearity of the process. In Tab. 4.1 the results for different SNRs for all investigated model types

SNR	property	ARX2	ARX4	FIR unreg.	FIR reg.
20 dB	NMSE	0.185	0.069	0.013	**0.0029**
	M	22	16	4	7
	n_{eff}	38	57	239	72
27 dB	NMSE	0.153	1.75	0.0073	**0.0023**
	M	49	21	6	8
	n_{eff}	66	89	241	98
30 dB	NMSE	0.122	0.034	0.0050	**0.0020**
	M	30	18	6	8
	n_{eff}	52	76	283	101
37 dB	NMSE	0.0618	1.02	0.0041	**0.0021**
	M	24	35	6	9
	n_{eff}	40	102	285	107
40 dB	NMSE	0.798	1.51	0.0039	**0.0019**
	M	36	42	6	9
	n_{eff}	57	134	284	107

Table 4.1: NMSE values on test data, number of local models M, and effective number of parameters n_{eff} for different local models and SNRs in the nonlinear Wiener case. Best performing values are marked bold. The effective number of parameters is rounded to the next integer value.

are shown. All investigated models with NMSE values greater than $2 \cdot 10^{-1}$ have been observed to be unstable. This does not occur for the unregularized local FIR models. Nevertheless, regularization leads to substantial improvement. For all investigated SNRs, the regularized models were able to achieve the best performance.

4.5.2 SISO Hammerstein System

Another nonlinear system is a Hammerstein nonlinear system. Here, in contrast to the Wiener system, the input of the system is fed to a nonlinearity and then the signal is applied to a dynamic system. A Hammerstein system of second order is

given by [90]

$$\begin{aligned} y(k) =& 0.01867 \arctan(u(k-1)) + 0.01746 \arctan(u(k-2)) \\ &+ 1.7826 \tan(y(k-1)) + 0.8187 \tan(y(k-2)). \end{aligned} \tag{4.22}$$

An APRBS between -3 and 3 is used for excitation. In Tab. 4.2 the results of the system are shown. It can be seen that in case of low noise the ARX model

SNR	property	ARX2	ARX4	FIR unreg.	FIR reg.
20 dB	NMSE	0.395	0.206	0.161	**0.143**
	M	53	24	5	8
	n_{eff}	99	97	249	90
27 dB	NMSE	0.270	**0.081**	0.138	0.116
	M	31	6	7	11
	n_{eff}	58	36	264	138
30 dB	NMSE	0.229	**0.0592**	0.144	0.124
	M	46	17	7	11
	n_{eff}	80	64	269	148
37 dB	NMSE	0.142	**0.0198**	0.141	0.126
	M	21	13	7	10
	n_{eff}	48	59	269	139
40 dB	NMSE	0.0746	**0.0232**	0.139	0.122
	M	28	33	7	10
	n_{eff}	48	107	269	128

Table 4.2: NMSE values on test data, number of local models M, and effective number of parameters n_{eff} for different local models and SNRs in the Hammerstein case. Best performing values are marked bold. The effective number of parameters is rounded to the next integer value.

achieves better peformance than the regularized FIR model, but for higher noise the regularized FIR model performs better. This is due to the inconsistency of the ARX estimation. The regularized FIR model does not have this problem. It is intereseting to note that although the true process is of second order, only the ARX model of fourth order performs better than the second order ARX model. It seems that the higher order of the ARX model compensates the incosistency of the ARX model.

4.5.3 MISO Wiener System

To illustrate the behavior of the identification routine for a system with multiple inputs, a Wiener system with two inputs is considered. Two second order systems

which are Euler-forward discretized versions of a continuous time second order system, with double poles at 0.8 for the first and 0.2 for the second input, are considered. These two systems are then fed to a nonlinearity of arctan type. The difference equations of the system are

$$\begin{aligned} s_1(k) &= 0.004679u_1(k-1) + 0.004377u_1(k-2) \\ &\quad +1.81s_1(k-1) + 0.8187s_1(k-2) \\ s_2(k) &= 0.1912u_2(k-1) + 0.112u_2(k-2) \\ &\quad +0.8987s_2(k-1) + 0.2019s_2(k-2) \\ y(k) &= \arctan(s_1(k) + s_2(k)). \end{aligned} \tag{4.23}$$

The time constants of the second order systems have been chosen such that the G_1 function represents a fast response while the G_2 function is representing a slow one. The system is identified for $N = 1\,000$ samples, and the output is corrupted by i.i.d. Gaussian noise such that the SNR is 27 dB. For testing $N_t = 50\,000$ noise free samples are generated. Three different local model types, a regularized local FIR model, an unregularized local FIR model, and a local ARX model of fourth order are identified. These are referred to as *unregularized FIR*, *regularized FIR*, and *ARX*. The order of the local FIR models is chosen as $n = 80$. The results of the identification, however, are not sensitive to the chosen FIR model order due to the appropriate regularization.

In Sect. 3.2.4, it has been demonstrated that regularized identification of impulse responses improves the variance error. This is confirmed for the nonlinear case, too. The identified local impulse responses are shown in Fig. 4.7.

The left two subfigures show the regularized behavior, while the right two subfigures show the unregularized case. For the slow time constant, the behavior can hardly be recognized for the unregularized case due to the sharp fluctuations from one value of the impulse response to the next. This is not the case for the regularized response. Here, it can also be seen that the gain of the different operating regimes identified by the LMN can be distinguished. The same effect is visible for the fast time constant. Especially parameters which are almost zero in the regularized case vary in the unregularized case due to the variance error. It is important to notice that the number of local models and thus the ability to represent the nonlinear behavior of the system is lower in the unregularized case too. In the unregularized case,

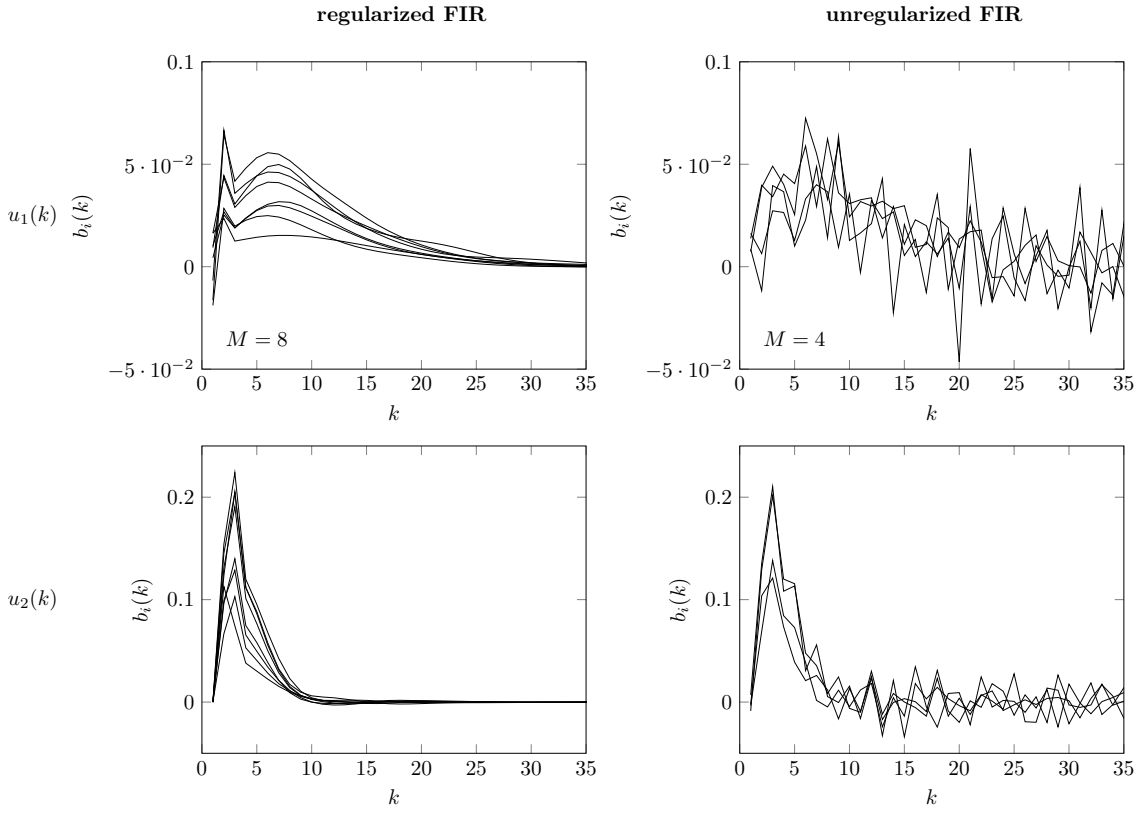

Figure 4.7: Local impulse responses of the identified local models. The left half of the figure shows the impulse response coefficients $b_i(k)$ of the local regularized FIR model, while the right half shows the impulse responses of the unregularized impulse responses with $i = 1 \dots M$. The upper plot corresponds to impulse responses from $u_1(k)$ and the lower to impulse responses from $u_2(k)$ to the output.

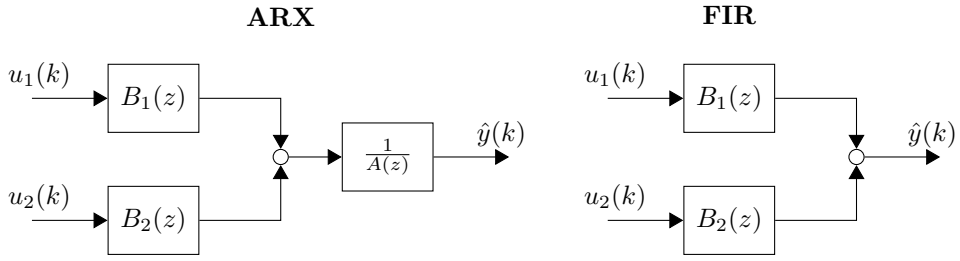

Figure 4.8: Block diagram of an ARX (left) and an FIR (right) model with two inputs. In contrast to the ARX model the FIR model has no common denominator dynamic.

$M = 4$ local models are chosen by the $\mathrm{AIC_c}$, while in the regularized case $M = 8$ models are selected. This is caused by the lower number of effective parameters for the regularized case, which is considered in the $\mathrm{AIC_c}$ to terminate the LOLIMOT training procedure.

Besides the biased estimate, the ARX model has another structural drawback for MISO systems. This is illustrated in Fig. 4.8. Here, block diagrams for the linear ARX and linear FIR case are shown. Both inputs in the ARX case share the same denominator dynamic $A(z)$. In consequence, if the two inputs have different dominating time constants, e.g., one input has a very rapid influence on the output, while the influence of the other is slow, the denominator $A(z)$ will have to contain both the fast and the slow pole and the nominator polynomials $B_1(z)$ and $B_2(z)$ have to cancel this dynamic. The structural behavior carries over to the nonlinear LMN case. Each local ARX model allows only for consideration of the influence of the previous global output. In contrast, for the FIR case, the dynamics are represented solely by the nominator dynamics $B_1(z)$ and $B_2(z)$, which have significantly more coefficients than in the ARX case. The only drawback is the higher amount of coefficients. This is diminished though by regularization leading to the demonstrated favorable behavior.

An excerpt of the identified model answer on test data is displayed in Fig. 4.9. Also, it can be seen that shortcomings of the different model types occurring in the linear case, generalize to the nonlinear case as well. The variance error made for the estimate of the FIR model parameters, which can also be seen in the impulse responses, becomes visible in the output signal, too. Furthermore, for the local ARX models, the dynamic representation of the process shows less variation for neighboring coefficients than for the unregularized FIR, but the overall system dynamics are not represented accurately. This is caused by the wrongly estimated time constants of

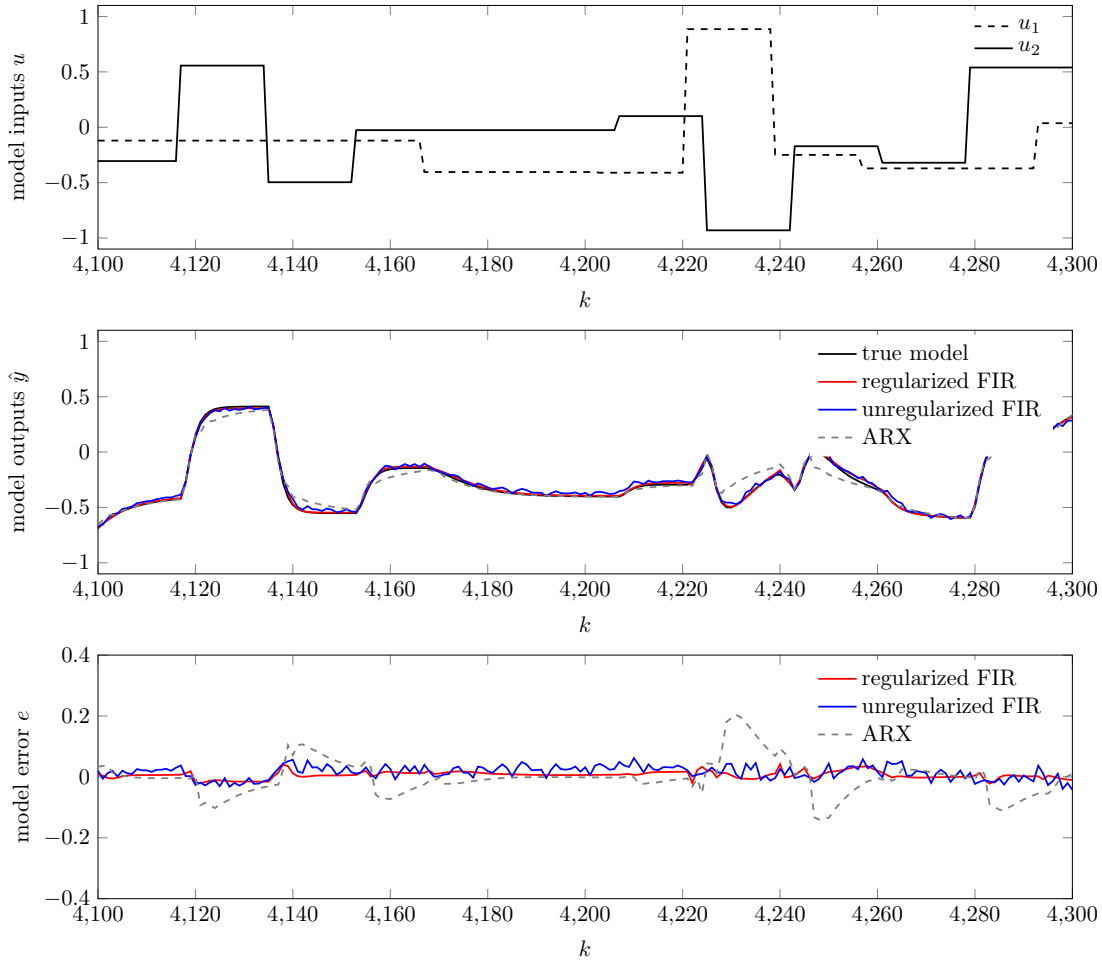

Figure 4.9: Excerpt of time response for test data. Results for local ARX, unregularized, and regularized local FIR models for LOLIMOT are shown. The regularized local models show the best performance on test data.

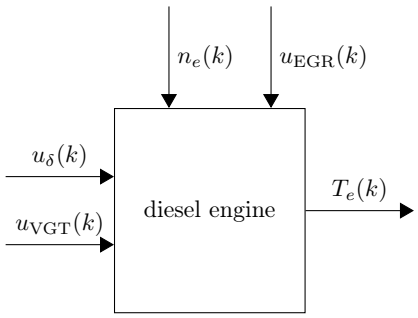

Figure 4.10: Block diagram of the investigated engine process.

the ARX model due to inconsistency issues. In contrast, the regularized FIR is able to represent the process accurately. The different normalized root mean squared error (NRMSE) values on test data are shown in Tab. 4.3. It is interesting to notice that regularized FIR and unregularized FIR are able to find comparably good representations on training data, while on test data, the performance of both methods differs significantly. The regularized FIR model is substantially better, here. Thus, it can be concluded that for regularized local FIR models, the applied regularization strategy is able to reduce the variance error significantly while keeping the bias error in an appropriate range.

Local model	NRMSE for training data	NRMSE for test data
regularized FIR	10.7%	**2.98**%
unregularized FIR	10.0%	4.65%
ARX	12.16%	9.37%

Table 4.3: Normalized root mean squared errors for the investigated local models

4.6 Application to a Diesel Engine Process

The simulative application of the method to a diesel engine process has been described in [82]. The Diesel engine model is described in [129] and models the torque $T_e(k)$ of a Diesel engine, which has a variable geometry turbocharger (VGT) and exhaust gas recirculation (EGR). There are two input signals that can be applied to the model: The mass of the injected fuel $u_\delta(k)$ and the position of the VGT actuator $u_{\mathrm{VGT}}(k)$. Both are scaled between 0 and 100, with 100 describing the maximal actuator position. To simplify the identification problem, the position of the EGR valve and the speed of the motor are kept constant. Furthermore, the nonlinear influence of the remaining two input signals is investigated.

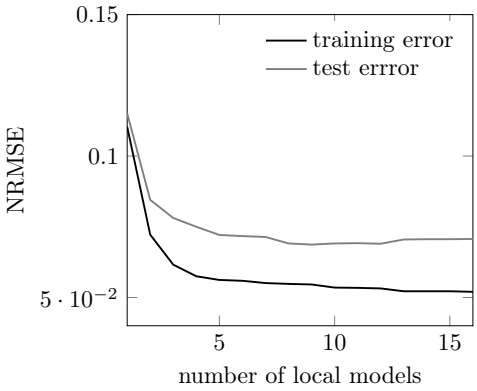

Figure 4.11: Training and test error of the regularized FIR local model network in dependence of the number of splits.

4.6.1 Data Generation

For the generation of samples of the system, uncorrelated APRBS signals with a minimum holding time of 0.5 s have been applied to u_δ and u_{VGT}. This is a reasonable trade-off between static and dynamic accuracy. The speed of the engine has been kept constant at a rate of $n_e = 2000$ rpm and the EGR valve is kept closed ($u_{\mathrm{EGR}} = 0$). The sampling time is chosen to be $T_s = 25$ ms. The simulated output is corrupted by different noise levels, yielding an SNR of 10000, 1000, and 100. These different cases will be called the *low*, *medium*, and *high* noise case. For the evaluation of the performance of the algorithm, the test data is kept noise free. Since no noise term is present in the NRMSE, it can be used for a comparison of the model performance over the different noise cases. The input u_δ is varied between 1 and 250 mg/cycle and the variable turbine geometry u_{VGT} between 1 % and 100 %.

4.6.2 Training of the Regularized Local Models

The local models are trained with the LOLIMOT algorithm with regularized local models, as described in Sect. 4.4.1. In Fig. 4.11, the convergence behavior of the algorithm is shown. Both the training and the test error are depicted. The best model has 8 local models since the generalization error monotonically decreases up to here. Due to the variance error, the training error continues to decrease, although the test error increases. This bias-variance behavior is very valuable. As will be seen in Chap. 5, this bias-variance trade-off is often tough to achieve for other model structures. Especially for neural networks, structure selection is a tough problem. For these several hyperparameters describing the structure have to be tuned. This

tuning requires several runs (usually around 100) of complete neural network training. Furthermore, a validation dataset is required to compare the performance of the resulting NN structures. For LMNs trained by the described method requires tuning of only one hyperparameter (the number of local models), which is determined reliably by AIC_c. This is a distinct advantage compared to deep neural networks, especially for small and medium-sized datasets where due to limited availability of data, no validation dataset can be provided.

4.6.3 Discussion of the Local Impulse Responses

To gain some insight into the behavior of the local models for the LMN, the identified local impulse responses of the engine model are shown in Fig. 4.12. There are two impulse responses for each local model, one for the injected fuel mass (upper part) and one for the variable turbine geometry. It can be seen from the regularized responses that the fuel mass has a very rapid influence on the engine torque $T_e(k)$, while for the turbine geometry, the time constant is significantly longer in several operation regimes. The unregularized FIR responses show such a strong change from one time step to the next. This hinders a reliable assessment of properties like gain or time constant from the depicted impulse responses. Another fact which can be seen from the responses of the variable turbine geometry input is that the time constant changes depending on the local model considered. Likewise, this fact is not visible in the unregularized responses due to the substantial variation from one time step to the next.

4.6.4 Comparison to Local ARX Models

To compare the behavior of the proposed method, also an LMN with local ARX models is identified with the LOLIMOT algorithm. In Fig. 4.13, the behavior of the models on train and test data is shown. It can be seen that both methods with local ARX and with local FIR models work fairly well for this example. In Tab. 4.4, the NRMSE values of the investigated models are shown. For high SNR (low noise

SNR	regularized FIR	ARX
10 000	0.079	**0.078**
1 000	**0.071**	0.08
100	**0.072**	0.076

Table 4.4: NRMSE values on test data for the corresponding model structures.

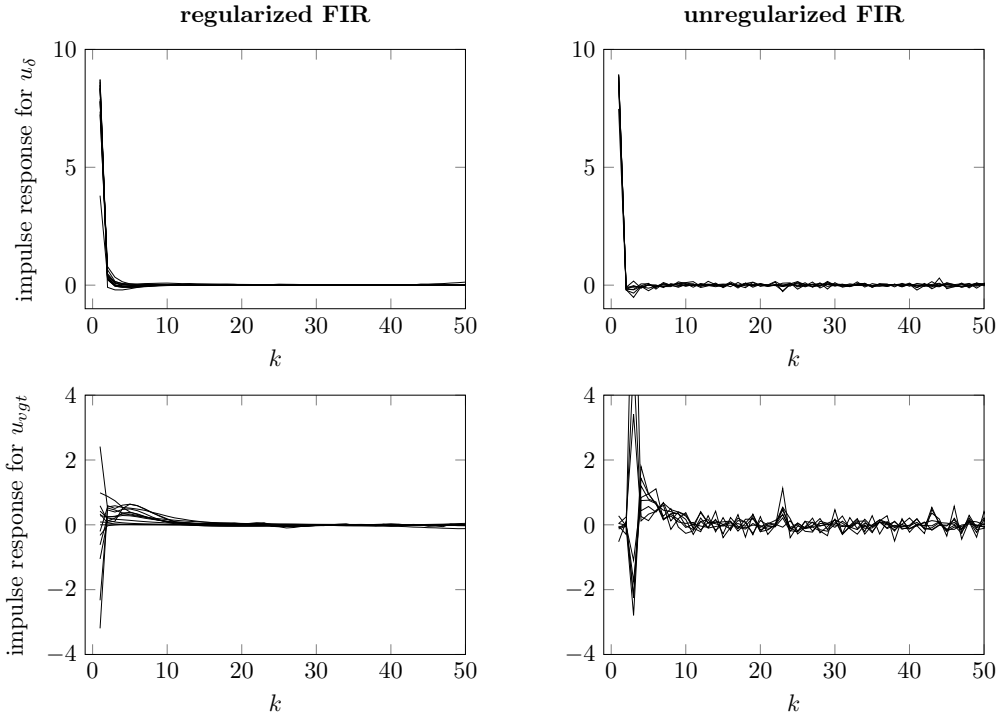

Figure 4.12: Local impulse responses for both the unregularized and the regularized local FIR models of the identified engine model.

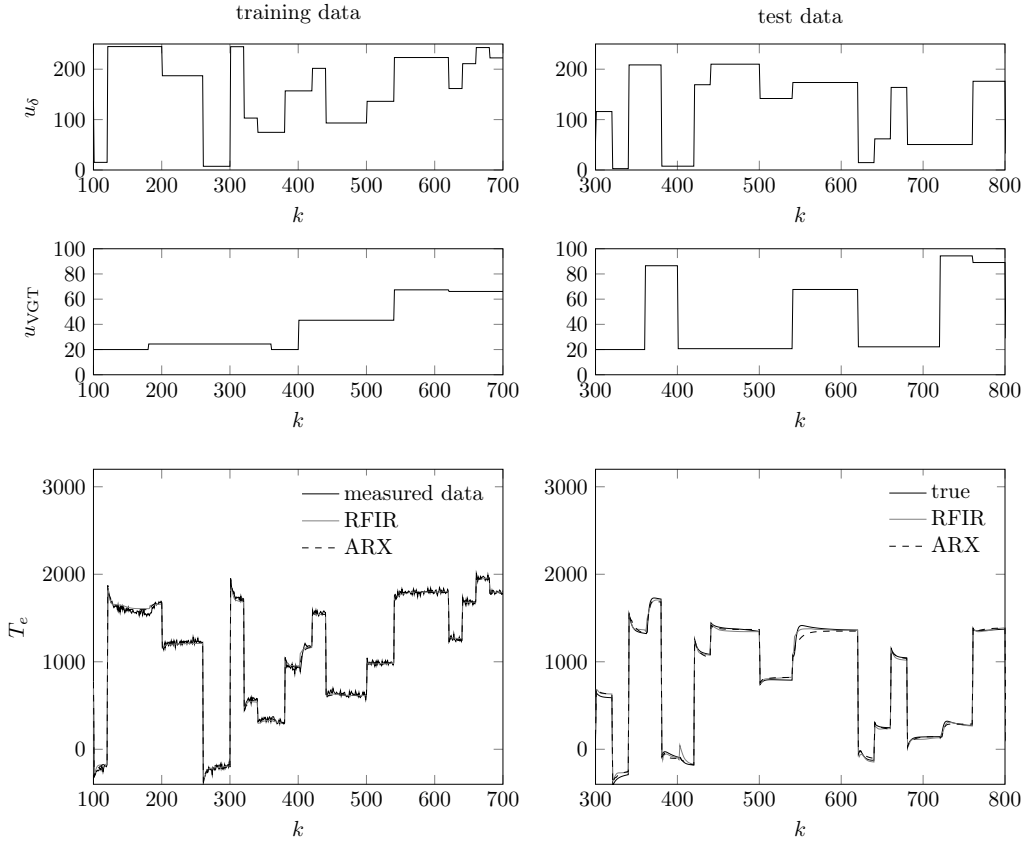

Figure 4.13: Excerpt of the responses of the models on training and test data.

case), ARX and regularized FIR work comparably well, with the ARX model working slightly better. For lower SNRs (high noise case), however, the regularized FIR model performs better than the ARX model.

Concluding, it has been shown that regularized local FIR models offer advantages compared to local ARX or local unregularized FIR models. The newly proposed models are guaranteed to be stable, while for all examples, the performance was at least comparable to established approaches. The stability guarantee thus comes without a cost. Furthermore, the performance has always been better than unregularized FIR models for all examples. This allows for a better representation of nonlinearities (due to a lower number of parameters) and a lower variance error for the estimated coefficients of the local FIR models.

5 Regularized Deep FIR Neural Networks

In the previous chapter, it has been shown that LMNs allow for an extension of the regularized FIR method from the linear domain to nonlinear problems. Due to the specific structure of LMNs, the estimation problem can be formulated as a weighted least squares problem with an analytical solution. Recently, neural networks with many layers have shown tremendous success in several machine learning problems [108, 63], including some applications to system identification [114]. Convolutional networks have also been applied for system identification [42, 7]. The goal of this chapter is to investigate whether the regularized FIR approach from the linear domain can be applied to identify a deep convolutional neural network structure, too. Usually, stochastic gradient descent (SGD) is applied as an optimization algorithm to train neural networks. This algorithm and its specific properties, compared with other optimization algorithms, are analyzed with the example of a linear FIR system first. Afterwards, the structure for a deep neural network which uses regularized FIR systems as building blocks is described. Finally, training this structure on the Bouc-Wen system, a standard benchmark for system identification, is investigated.

5.1 Properties of SGD – A Review for FIR Identification

As described in Sect. 2.1.3 for the training of a neural network, a non-convex optimization problem is to be solved. A commonly employed characteristic is that Neural Networks (NNs) are trained by stochastic gradient descend (SGD). In principle, this algorithm is straightforward. It calculates the gradient of the loss function on a subset of data (the mini-batch) and takes a step downwards this negative gradient. This algorithm has the appealing advantage that its computational complexity grows only

linearly with the number of parameters of the NN and also linearly with the number of samples within the mini-batch [51].

5.1.1 Unregularized FIR

Stochastic gradient descend (SGD) is considered in the context of unregularized linear FIR models first. The principle of SGD is to update the parameter in the descending direction of the negative gradient of the loss function. This gradient is calculated only on a subset of the data. The data subset onto which the gradient is calculated is called a *mini-batch* [51]. As described in Sect. 2.2.1, the calculated output of a linear FIR model can be written as

$$\underline{\hat{y}_b} = \underline{X}_b \, \underline{\theta}. \tag{5.1}$$

The process output vector $\underline{y}_b$ and the modeled output vector $\underline{\hat{y}}_b$ contain only a subset of N_b samples of the output. Here, it is assumed that at each step of the SGD algorithm, these samples are randomly picked (without replacement) from the complete output $\underline{y}$. The rows from the complete regressor $\underline{X}$ of the FIR problem are picked at the same locations and concatenated in the batched regressor $\underline{X}_b$. Then, the loss function for the mini-batch reads as

$$J_b = \frac{1}{N_b} \underline{e}_b^T \underline{e}_b = \frac{1}{N_b} (\underline{y}_b - \underline{\hat{y}}_b)^T (\underline{y}_b - \underline{\hat{y}}_b) \tag{5.2}$$

with the vector of errors for the mini-batch $\underline{e}_b$. It follows from the chain rule that the gradient of the loss function is

$$\frac{\partial J_b}{\partial \underline{\theta}} = \frac{1}{N_b} \frac{\partial \underline{e}_b^T \underline{e}_b}{\partial \underline{\theta}} = -\frac{2}{N_b} \left[\frac{\partial \underline{\hat{y}}_b}{\partial \underline{\theta}} \right]^T \underline{e}_b. \tag{5.3}$$

The SGD algorithm simply moves along the descending direction of the negative gradient with a learning rate η, and iteratively computes the updated value of the parameters $\underline{\theta}^{(n+1)}$ at the n-th step of the algorithm according to

$$\underline{\theta}^{(n+1)} = \underline{\theta}^{(n)} + \eta \frac{2}{N_b} \left[\frac{\partial \underline{\hat{y}}_b}{\partial \underline{\theta}} \right]^T \underline{e}_b. \tag{5.4}$$

For the linear FIR case, the gradient of the modeled output with respect to the parameter is

$$\frac{\partial \underline{\hat{y}}_b}{\partial \underline{\theta}} = \underline{X}_b. \tag{5.5}$$

Inserting in (5.4) results in

$$\underline{\theta}^{(n+1)} = \underline{\theta}^{(n)} + \eta \frac{2}{N_b} \underline{X}_b^T \underline{e}_b \tag{5.6}$$

for the SGD algorithm. To demonstrate the mechanism of SGD, the second order system

$$G(z) = \frac{0.49z^2 + 0.48z}{z^2 - 1.84z + 0.94} \tag{5.7}$$

is excited with $N = 1\,000$ samples of a PRBS input signal. The output is disturbed by i.i.d. Gaussian noise with a variance of $\sigma^2 = 0.1$.

The optimization problem is, from a computational perspective, simple. It has a unique, analytically calculable optimum. If a second order method, like Newton's method, is applied, it converges within one step to this optimal solution.

If the behavior of SGD is concerned, its behavior depends strongly on the chosen learning rate η. Convergence shows in all cases a non-monotonic behavior. For a too high learning rate, its convergence is poor locally, and for a too small learning rate its convergence is very slow. This is exemplified in Fig. 5.1.

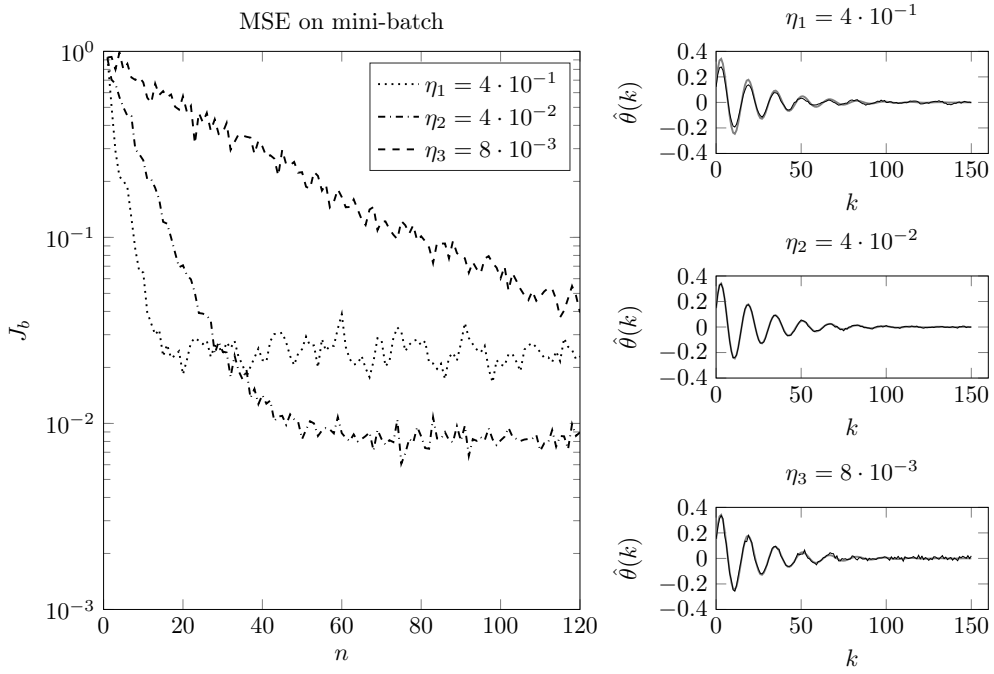

Figure 5.1: Comparison of different learning rates ($\eta_1 = 4 \cdot 10^{-1}$, $\eta_2 = 4 \cdot 10^{-2}$, $\eta_3 = 8 \cdot 10^{-3}$). On the right side, the estimated impulse response is shown in black and the true impulse response in gray. For high η, the local convergence is poor. For low η, convergence is slow.

If now the problem is changed from linear to nonlinear, the problem becomes even more pronounced. There is the possibility that the algorithm gets stuck within a locally optimal solution or a saddle point. Recently, there has been some work, which analyzes the escaping behavior of SGD from saddle points [57].

In contrast to all these disadvantages, SGD has one significant advantage: Its computational simplicity. The gradient of the loss function can be computed with the same amount of operations as a forward-pass through a neural network. Furthermore, its formulation allows for mini-batches and so the computation of a sample-wise gradient. For applications having a huge amount of data and for structures with many parameters, this advantage is dominant, despite the existing disadvantages.

The application of SGD makes training of NNs a time-consuming activity. Often, it is necessary to adjust the sizes of the mini-batches and the learning rate specifically for the problem. And, even worse, the chosen structure of the network can interact with the optimal learning rate. Usually, for the best training speed, the mini-batch size is chosen according to the available cache of the GPU as a power of two. If, for example, the number of neurons, e.g., is changed, it can be required to change the learning rate too. Some heuristics, like the Adaptive Moment (Adam) algorithm [60], which heuristically adjusts the learning rate automatically, are of help, but the fundamental issues remain.

5.1.2 Regularized FIR

In Sect. 2.2.2 and Chap. 3, techniques for regularized identification of FIR systems are described. These techniques work by adding a penalty term to the objective function of the optimization problem for the identification of the FIR parameters. For NNs, it is also common to apply a regularization term for the prevention of overfitting. Adding a quadratic penalty for the parameters, according to

$$J_b = \frac{1}{N_b} \sum_{i=1}^{N} \left(y(i) - \hat{y}(i) \right)^2 + \lambda \underline{\theta}^T \underline{\theta}, \tag{5.8}$$

is referred to as *weight decay* [51]. It is equivalent to the simplest form of Tikhonov regularization and its Bayesian interpretation is described in Sect. 2.1.5. Many additional and/or alternative regularization schemes, like drop-out or batch-normalization, exist, see [108, 51] for an overview, but are not elaborated on in this thesis. Often, the performance of the NN on test data can be influenced positively by regularization. This might have several reasons. One reason is that due to regularization,

the variance error of the estimate is reduced. The other is that the penalty term within the gradient guides the estimate of the parameters obtained by SGD to a useful region within the high dimensional parameter space. The prior distribution for $\underline{\theta}$, see (2.46), which is required to obtain (5.8), does not consider any correlation between entries of $\underline{\theta}$. For linear identification, as described in Sect. 2.2.2, it can be beneficial to incorporate correlations into the prior distribution. For the regularized FIR case, the objective function is slightly more complex with a penalty term that considers the correlation between the entries of $\underline{\theta}$. It is formulated as

$$J_b = \frac{1}{N_b} \sum_{i=1}^{N_b} \left(y_b(i) - \hat{y}_b(i) \right)^2 + \lambda \frac{N_b}{N} \underline{\theta}^T \underline{P}^{-1} \, \underline{\theta}. \tag{5.9}$$

The term $\frac{N_b}{N}$ occurs since the gradient is calculated only on a subset containing N_b samples. If the gradient step for SGD is calculated for this structure, one obtains

$$\underline{\theta}^{(n+1)} = \theta^{(n)} + \frac{2}{N_b} \eta \underline{X}_b^T \, \underline{e}_b - \eta \frac{2N_b}{N} \lambda \underline{P}^{-1} \, \underline{\theta}^{(n)}. \tag{5.10}$$

So, for a quadratic regularization, SGD steers the solution of the problem to a region within the parameter space, which has both a low error and a small penalty value.

5.2 Architecture and Training Procedure

The idea of employing neural networks for nonlinear FIR identification is not new. In [102], a special form of an NFIR neural network is proposed. To handle the dimensionality of the NFIR NN, the regressor $\underline{X}$ of an FIR model is decomposed into its principal components using principle component analysis (PCA). Then a neural network for the projection of the input onto the first singular vector of the PCA is trained such that the squared output error is minimized.

More recent approaches employ convolutional NNs of different types for nonlinear identification. In [7] a deep convolution NN is learned for several problems. There the relation to block-oriented models is also described. In [42], convolutional NN are applied to benchmark problems. Regularization is only utilized in the form of weight decay or as a dropout regularization [7]. The regularized FIR approach is not considered in the current literature.

5.2.1 Architecture

For the identification of NFIR systems, a novel architecture is introduced. The architecture consists of several *units*, where each unit consists of a temporal convolutional layer followed by two densely connected nonlinear layers. At the end of the NN, a linear layer is connected to describe the output of the network. In Fig. 5.2, the structure of the NN is depicted.

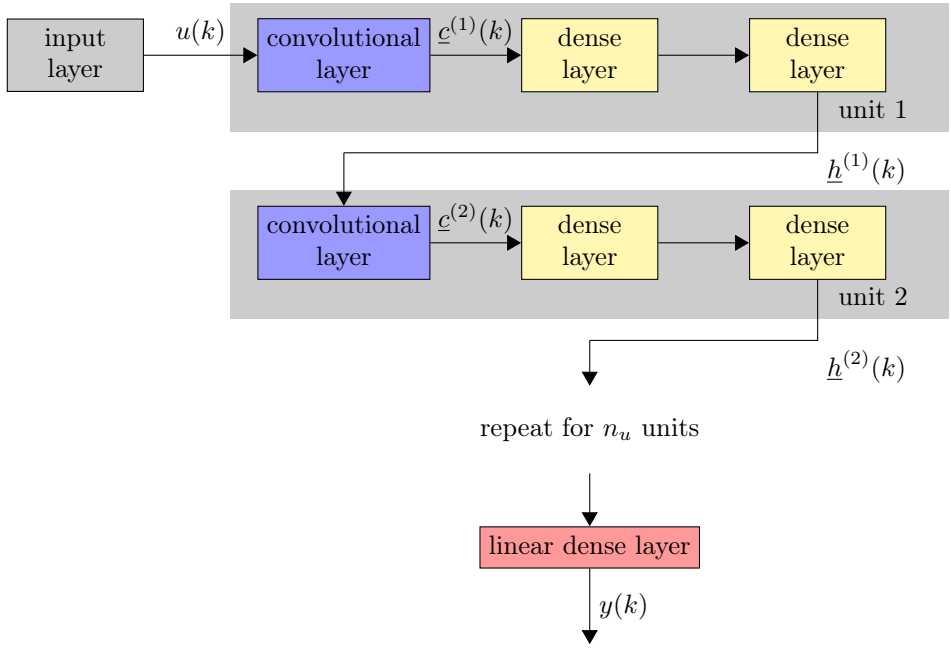

Figure 5.2: Structure of the Deep Regularized FIR Neural Network from the machine learning perspective.

Input layer The input layer is utilized to preprocess the data for the NN. For most NN structures, a rescaling is advantageous. For this architecture, the data is scaled to zero mean and unit variance. The same scaling is applied to the training output, too.

Index	Range	Description
i	$0 \dots n_u$	number of unit
j	$0 \dots n_n$	number of input
k	$0 \dots N_b$	discrete time
l	$0 \dots n_n$	number of output

Table 5.1: Indices for the notation of quantities contained in the Deep RFIR NN

Convolutional layers The input layer is followed by a convolutional layer. In contrast to image processing NNs, the convolutional units here are one dimensional along

the time dimension. The hidden values of the output of the convolutional layer are denoted by $c_l^{(i)}(k)$ for the output of the l-th convolutional neuron for the i-th unit at the discrete time step k. The notation $b_{jl}^{(i)}(k)$ denotes the k-th filter coefficient for the filter from the j-th input to the l-th hidden output for the i-th unit.

The scalar n_n is the number of neurons for each layer. In principle, n_n can be chosen differently for each layer. This has the advantage that a custom-tailored NN architecture can be found. The disadvantage is that instead of one value n_n, one parameter per layer has to be found. Usually, this increase in the search space for hyperparameter tuning is not advantageous, and thus the proposed architecture has the same number of neurons in each layer.

For the first convolutional layer, the equation describing the layer is

$$c_l^{(1)}(k) = \sigma\left(\sum_{m=0}^{n} b_l^{(1)}(m)u(k-m)\right) \tag{5.11}$$

for $l = 1, \ldots, n_n$. For activation, a Relu function $\sigma(x) = \max(0, x)$ is employed. Thus, each neuron of the first hidden layer is a convolution of the corresponding input. An extension to multiple inputs is straightforward.

In most software packages for neural networks, like Tensorflow [1], the convolution is not implemented in exactly this way. Instead of a true convolution, the cross-correlation is calculated. This, however, has no severe consequence. The value of $b_{jl}^{(i)}(k)$ can simply be found by flipping the entries along the time dimension k. The number of parameters for the first convolutional layer is $n_n(n+1)$.

For the following layers, the equations describing convolutional layers are

$$c_l^{(i)}(k) = \underline{\sigma}\left(\sum_{j=1}^{n_n}\sum_{m=0}^{n} b_{jl}^{(i)}(m)h_j^{(i-1)}(k-m)\right) \tag{5.12}$$

The final output of the units is denoted by $h_l^{(i)}(k)$. As a reminder, index i refers to the number of the unit, the index l refers to the number of the output, and k is the discrete time step.

The number of parameters for the convolutional layers are $n_n^2(n+1)$. These layers contain most of the parameters of the neural network due to the multiplication of the n_n^2 terms with the number of filter coefficients.

Nonlinear dense layers The output of the convolutional layer is fed to two nonlinear dense layers. These are described by the equations

$$\underline{h}^{(i)}(k) = \underline{\sigma}(\underline{A}^{(i,2)}\underline{\sigma}(\underline{A}^{(i,1)}\underline{c}^{(i)}(k) + \underline{d}^{(i,1)}) + \underline{d}^{(i,2)}). \tag{5.13}$$

for the i-th unit with the parameters $\underline{A}^{(i,1)}, \underline{A}^{(i,2)}$ for the description of the interactions and $\underline{d}^{(i,1)}$ and $\underline{d}^{(i,2)}$ for the offsets. The number of parameters of the two nonlinear dense layers combined per unit is $2(n_n^2 + n_n)$.

Linear dense layer The last computation of the network is a linear dense layer which combines the filtered responses to the system response according to

$$\hat{y}(k) = \sum_{l=1}^{n_n} h_l^{(n_u)}(k) p_l + q. \tag{5.14}$$

with an additional offset parameter q and linear parameters $\underline{p}$. The linear dense layer thus has $n_n + 1$ parameters.

In total, a network with n_u units and n_n neurons has

$$n_p = \underbrace{(n_u - 1)n_n^2(n+1) + n_n(n+1)}_{\text{convolutional layers}} + \underbrace{n_u\left(2n_n^2 + 2n_n\right)}_{\text{nonlinear dense layers}} + \underbrace{n_n + 1}_{\text{linear dense layer}} \tag{5.15}$$

parameters. Most of the parameters are contained within the convolutional units.

5.2.2 Regularization

To deal with the variance error due to the large number of parameters, a regularization term is applied. The loss function of the NN is

$$\begin{aligned} J = &\sum_{k=1}^{N_b} \left(y(k) - \hat{y}(k)\right)^2 \\ &+ \underbrace{\lambda \sum_{j=1}^{n_n} \sum_{m=0}^{n} \underline{b}_j^{(1)T} \underline{P}^{-1} \underline{b}_j^{(1)}}_{\text{1. layer}} + \underbrace{\lambda \sum_{i=2}^{n_u} \sum_{j=1}^{n_n} \sum_{l=1}^{n_n} \underline{b}_{jl}^{(i)T} \underline{P}^{-1} \underline{b}_{jl}^{(i)}}_{\text{2. ... } n_u\text{. layer}} + \lambda \underline{w}^T \underline{w}. \end{aligned} \tag{5.16}$$

The parameter vector $\underline{w}$ contains all parameters of the NN which are not filter coefficients for the convolutional layers. The matrix $\underline{P}$ is a kernel matrix obtained by a TC kernel as described in Sect. 2.2.2. Thus, $\underline{P}$ itself has only one hyperparameter - the decay factor α. This decay factor is chosen to be constant as $\alpha = 0.93$ in all the studies to simplify the search for optimal hyperparameters. A strategy to tune

this hyperparameter is left for future work. This a demanding task since no closed expressions for GCV or marginal likelihood are available.

5.2.3 Relation between FIR and Convolutional Models

There are two possible perspectives on the proposed structure. The first sees the structure as a convolutional neural network with a specific kind of regularization for the filters. The second perspective considers the system as a block-oriented nonlinear system. This has also been described in [7]. From this second perspective, for the first layer, there is only one input and the transfer functions are

$$B_l^{(1)}(z) = \sum_{m=0}^{n} b_l^{(1)}(m) z^{-m} \tag{5.17}$$

with $l = 1, \dots n_n$. In the first layer the input is fed to n_n transfer functions. From one time signal, n_n time signals are generated. These time signals are fed to individual nonlinearities $\sigma(\cdot)$ each. From the NN perspective, this nonlinear block usually belongs to the convolutional layer, whereby from the control perspective, two separate blocks are displayed. This signal is then fed to another nonlinearity. This block is a static nonlinearity. It corresponds to the densely connected layers of the NN, see (5.13) and reads as

$$\underline{h}^{(i)}(k) = \underline{N}^{(i)}(\underline{c}^{(i)}(k)) = \underline{\sigma}\left(\underline{A}^{(i,2)}\underline{\sigma}(\underline{A}^{(i,1)}\underline{c}^{(i)}(k) + \underline{d}^{(i,1)}) + \underline{d}^{(i,2)}\right). \tag{5.18}$$

Here, the control viewpoint offers a new perspective. The nonlinearity to which the signals are applied is static. Thus, the output of this block is not affected by the previous values of the input. Dynamic behavior happens solely in the transfer functions realized by the convolutional layer.

For the following layers, again, more indices are required to describe the transfer functions, but the principle remains the same. So, for the i-th unit, the transfer functions of the convolutional unit are

$$B_{jl}^{(i)}(z) = \sum_{m=0}^{n} b_{jl}^{(i)}(m) z^{-m}. \tag{5.19}$$

The block diagram of the proposed NN architecture is shown in Fig. 5.3. The diagram starts with the input entering the first unit. This unit consists of the blue block diagram part, equivalent to the convolutional unit. Interestingly, the block diagram which describes the convolutional unit contains a nonlinear block at the end. This

nonlinear block is due to the nonlinearity, usually part of the convolutional layer in a NN. The two dense layers following the convolutional unit are represented by a nonlinearity block (yellow) within the block diagram. The output of each unit is a multidimensional signal $\underline{h}^{(i)}(k)$ with the dimension equal to the number of neurons n_n. This output is then the input for the subsequent layer. This procedure is carried on until n_u units have been passed. At the end of the NN, the n_n dimensional output of the last layer is linearly weighted and summed up. Finally, the offset is added to obtain the output $\hat{y}(k)$. Even though stemming from fundamentally different disciplines, convolutional NNs and block-oriented systems share strong similarities. Therefore it makes perfect sense to apply techniques from the machine learning community to identify block-oriented systems.

5.2.4 Training Procedure

Since it is not guaranteed that the globally optimal solution for the optimization problem is found, several properties of the training procedure matter. A significant influence can be attributed to the size of the mini-batches, kind of initialization of the parameters, and the applied optimization algorithm.

Mini-batches As usual for SGD methods, training is performed with mini-batches. The training data thus has to be subdivided into mini-batches of appropriate size. This has the advantage that the model can be trained efficiently on a GPU. Usually, there is one input signal $u(k)$ and one output signal $y(k)$ which is available for values of k between 1 and N. This signal has to be subdivided into several mini-batches. For the identification of FIR systems, the input signal has to be longer than the output signal, since n delayed input values are required for the computation of the first output for $y(k)$. For an NN with n_u convolutional layers, with n filter coefficients each, this means that

$$n_t = nn_u \tag{5.20}$$

additional input values are required for the computation of the first output. For the division of the data, these n_t samples have to be skipped only once. For the following batches, some values of $u(k)$ will be contained in both the actual and the previous batch to avoid lost values of the output. The procedure is illustrated in Fig. 5.4. The total number of batches can be calculated according to

$$n_b = \frac{N - n_t}{N_b}. \tag{5.21}$$

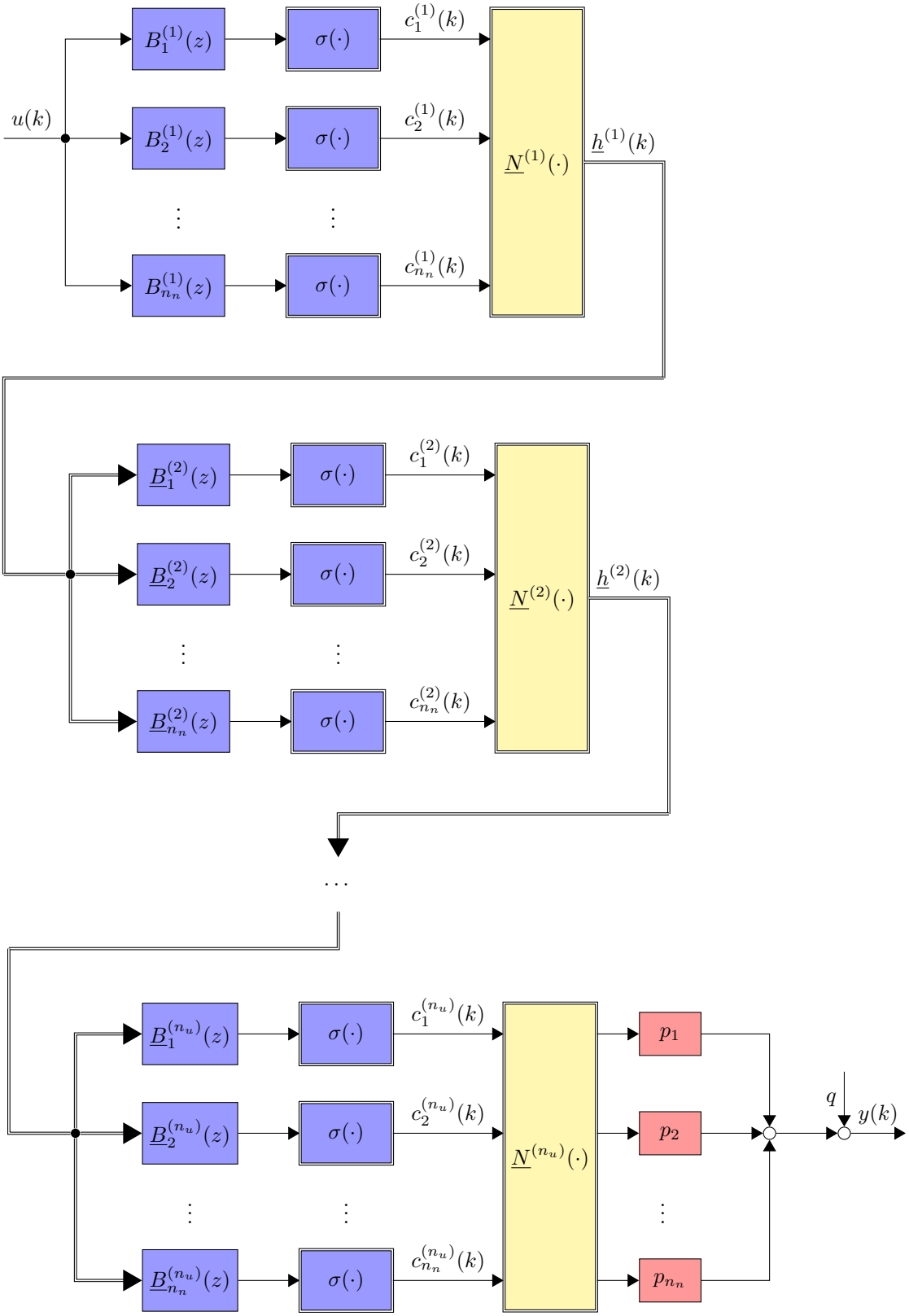

Figure 5.3: Block diagram for the block-oriented perspective on the proposed NN structure. The parts which correspond to convolutional units are colored blue, parts which correspond to the nonlinear layers which consist of two dense layers are shown in yellow, and the linear output unit is shown in red.

If n_b is not an integer, this means that the last batch will contain less than N_b samples if rounded up. In this case, either a batch with a lower number of samples is used, or the samples contained in this batch are discarded. For datasets containing a large number of samples, the latter can be reasonable, since the variance of the gradient will be higher, the fewer samples are contained within one batch. If not

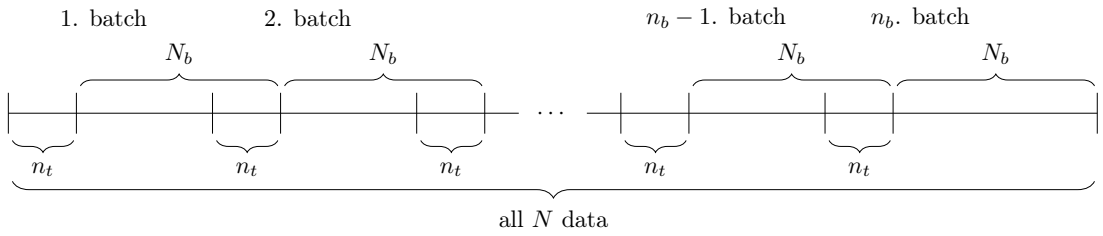

Figure 5.4: Generation of mini-batches for sequential input and output data of a neural network.

noted otherwise, the number of samples in a batch has been chosen as $N_b = 128$.

Initialization Several initialization schemes for NNs are available. For an overview, see [51]. For the studies described here, the parameters are initialized by a Glorot initialization scheme [49]. The initialization of the weights works according to

$$\underline{b}_{\text{ini}} \sim \mathcal{U}\left(-\sqrt{\frac{3}{n_n}}, \sqrt{\frac{3}{n_n}}\right), \tag{5.22}$$

with $\underline{b}_{\text{ini}}$ denoting the initial values of the parameters for the layer.

Optimizer For optimization, the Adam optimization algorithm [60] is applied. There are several types of heuristics for the optimization algorithms available [51]. The difference to SGD is that a so-called momentum is defined. This means that the gradient is calculated as an exponentially decaying average of the gradients of previous mini-batches and the gradient of the current mini-batch. From a systems perspective, the gradient is low-pass filtered. Adam furthermore scales the gradient with respect to each parameter according to the variance of the previous gradients. The default learning rate is chosen, if not mentioned otherwise, as the recommended default value of $1 \cdot 10^{-3}$ [60].

5.3 Results on the Bouc-Wen Benchmark Example

The proposed neural network structure is evaluated on a benchmark example. An example, which is known to be notoriously difficult due to its internal states describing

the friction, is the Bouc-Wen benchmark [112]. The performance of the proposed method on the benchmark will show that it is possible to represent systems with nonlinear internal states by methods without any feedback appropriately.

5.3.1 Training Data Generation

The Bouc-Wen benchmark has been proposed in [112]. It describes the movement of a mass attached to a damper and spring with a hysteretic behavior. The differential equations describing the behavior are

$$m_L\ddot{y}(t) + r(y,\dot{y}) + z(y,\dot{y}) = u(t), \tag{5.23}$$

with

$$r(y,\dot{y}) = k_L y + c_L \dot{y}. \tag{5.24}$$

The function $z(\cdot)$ fulfills the first-order differential equation

$$\dot{z}(y,\dot{y}) = \alpha_h \dot{y} - \beta_b(\gamma_b|\dot{y}||z|^{\nu_b-1}z + \delta_b\dot{y}|z|^{\nu}). \tag{5.25}$$

For the benchmark problem, the parameters are $m_L = 2$, $c_L = 10$, $k_L = 5 \cdot 10^4$, $\alpha_b = 5 \cdot 10^4$, $\beta_b = 10^3$, $\gamma_b = 0.8$, $\delta_b = -1.1$ and $\nu_b = 1$ [112]. To numerically integrate the differential equation, a Newmark integration scheme, according to the benchmark description [112], is employed.

The excitation signal is a random phase multisine signal. This signal is described by [111]

$$u(k) = \sum_{n=n_{\min}}^{n_{\max}} \sin\left(2\pi\frac{n}{N}k + \phi_i\right) \tag{5.26}$$

with $n_{\min/\max} = \frac{f_{\min/\max}}{T_s}$ and $f_{\min/\max}$ denoting the lowest and highest frequency of interest, respectively. If $n_{\min/\max}$ is not an integer, it is usually rounded to the next higher discrete frequency value. The phase ϕ_i of the different sinusoidal components is randomly sampled from a uniform distribution between 0 and 2π. If not noted otherwise, the performance on validation data is obtained by a validation signal with 10^5 samples generated by the aforementioned procedure.

5.3.2 Influence of the Regularization Strength

To analyze the effect of regularization, two different experiments with the same NN structure and training setup are performed. The goal of regularization is to prevent

the NN from overfitting. A relatively complex NN with 4 units, 40 neurons per layer, and 50 filter coefficients per convolutional layer (398 151 parameters in total) is considered.

The dataset contains 12 800 output samples (leading to 100 batches with a length of 128 samples) for training. The learning rate is chosen as $5 \cdot 10^{-4}$. Now, a completely unregularized version and a regularized version with $\lambda = 10^{-4}$ are compared. The convergence behavior for training data with a high SNR of the NN structures is shown in Fig. 5.5. By default, the output data generated for the benchmark is disturbed by i.i.d. Gaussian noise such that the SNR is 40 dB [94]. This is referred to as low noise or high SNR case. Training these NNs is performed on two Tesla P100 GPUs on the CfaDS cluster of FH Bielefeld. The training of each NN takes approximatively 4 h with this infrastructure.

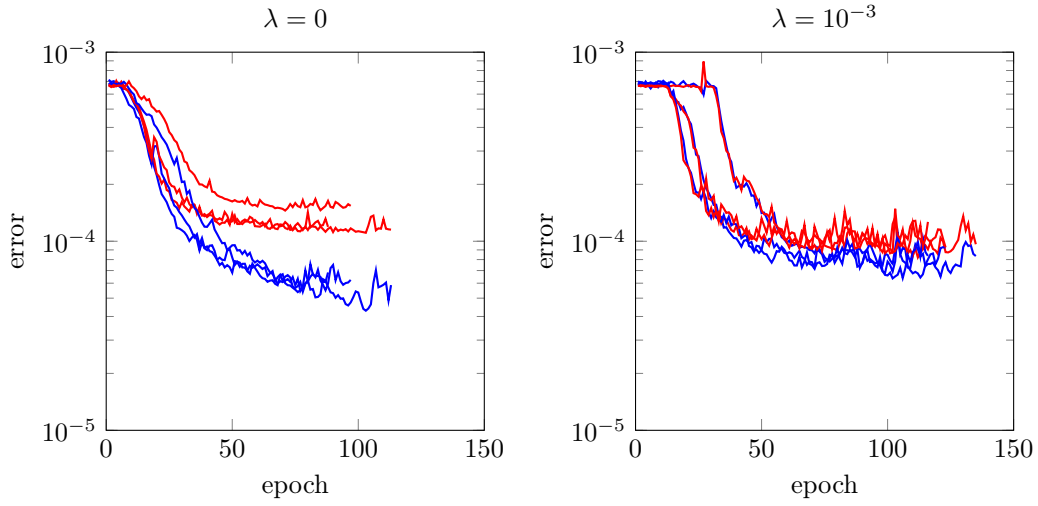

Figure 5.5: Convergence for low noise case of training (blue) and validation (read) error for a deep RFIR network with 4 layers and 40 neurons per layer for a regularized and an unregularized case.

It can be seen that for the low noise case (high SNR), the regularized version performs only slightly better than the unregularized one. This holds, although it is visible that the unregularized version overfits the training data, while the regularized version reaches comparable performance on training and validation data. This is a remarkable and highly surprising result since the number of parameters is more than 30 times higher than the number of samples for training. This ability of NNs to generalize, although the training data is overfitted, has been observed for other machine learning problems, especially image recognition, too [134]. The mechanism of this generalization is not completely understood, although several hints indicate

that a favorable interpolation behavior of the NN contributes to the successful generalization. There are also some classical interpolating methods, e.g., nearest neighbor methods [43], which perform reasonably on unseen test data, although the training data is fit perfectly.

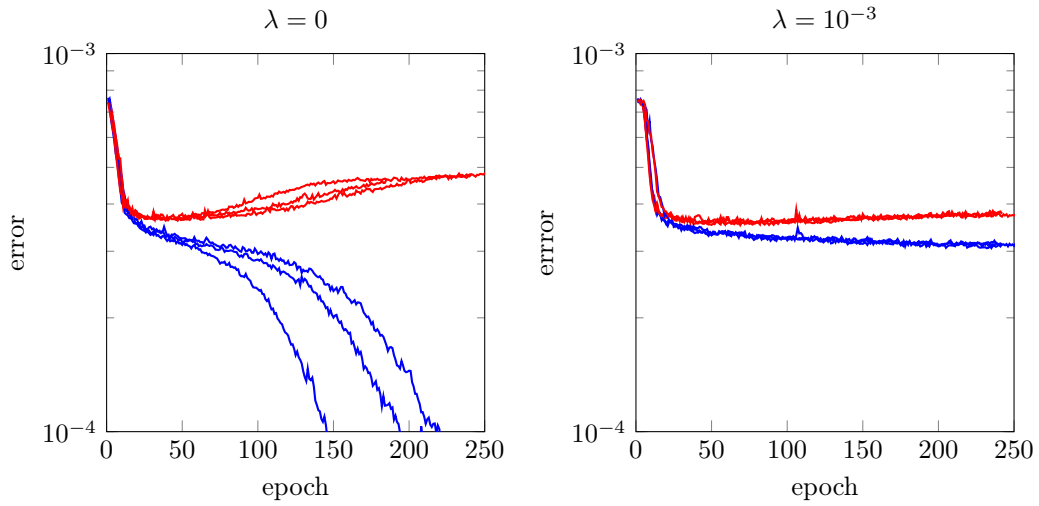

Figure 5.6: Convergence for high noise case of training (blue) and validation (red) error for a deep RFIR network with 4 layers and 40 neurons per layer for two different regularization strengths.

In the second scenario, the information contained in the data is reduced. To accomplish this, the output is disturbed by i.i.d. Gaussian noise, such that the SNR of the data becomes 6 dB (high noise case). The other hyperparameters and the structure of the NN are chosen in the same way as in the first scenario. Here, the overfitting of the NN structure becomes much more pronounced in the unregularized case. The training error is further reduced after 80 epochs, but the validation error starts to increase. For the regularized case, this problem is significantly reduced. Also, here, a small amount of overfitting can be observed, but it is not as substantial as for the unregularized case. This shows that the proposed regularization scheme is able to control the capacity of the NN. Though, also in the high noise example, it is remarkable that the unregularized deep NN is able to achieve a model that is able to represent the behavior of the system at all due to the high number of unregularized parameters.

5.3.3 Depth

For the analysis of the influence of the depth of the deep RFIR NN, the number of units is changed systematically. In Fig. 5.7, it is shown that the error on validation

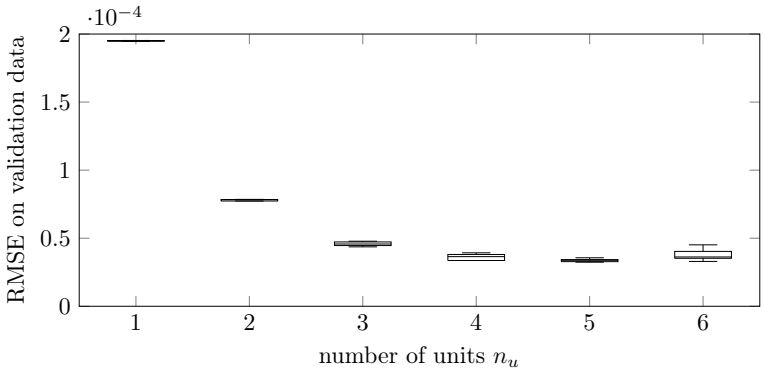

Figure 5.7: Dependence of the RMSE on validation data a deep regularized FIR NN with 50 neurons per layer on the number of units. For each number of units 5 experiments have been conducted.

data decreases significantly with the number of units connected in series. The same observation has been made for many other NN network structures. It is, however, not fully understood. It has been derived that for deep fully connected ReLU NNs, the number of linear regions of the output of the NN increases exponentially with the number of layers [51]. But it is not clear why the best of these, exponentially many, solutions is learnable and if SGD is able to recover that solution.

5.3.4 Number of Neurons

As another architectural relevant part, the number of neurons n_n of the NN has to be chosen. It is important to notice that the hyperparameter n_n affects the total number of parameters quadratically. In Fig. 5.8 the number of neurons of a $n_u = 4$ units deep RFIR NN has been varied. For each number of neurons, 3 runs have

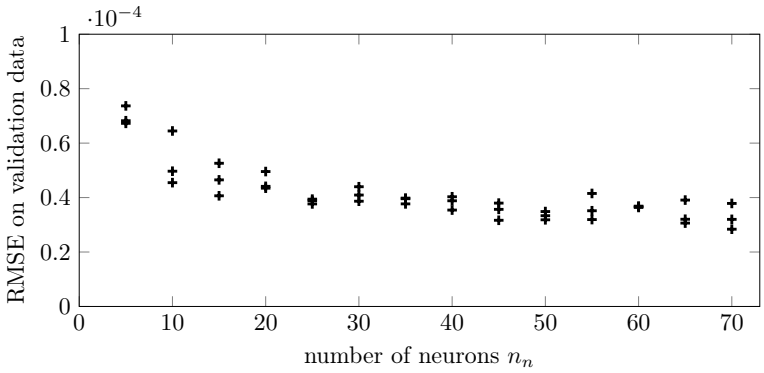

Figure 5.8: Dependence of the RMSE on validation data for a 4 layer deep regularized FIR NN on the number of neurons per layer.

been performed, to reduce effects resulting from different initializations. The same

training setup as for the variation of the number of units is employed.

It can be seen that also the number of neurons has an effect on the performance of the NN. However, the effect is not as significant as for the number of layers. For $n_n < 50$, an increasing number of neurons affects the error positively. Afterwards, the error remains relatively constant. The variations are more due to different initializations of the NN training than due to the number of neurons.

5.3.5 Best Performing Network

The proposed neural network structure is evaluated on the Bouc-Wen benchmark [94, 112]. Each configuration of a NN corresponds a unique nonlinear optimization problem. This optimization problem is then solved by an algorithm for which the solution is known to depend strongly on the initialization of the parameters. It is the goal to adapt the hyperparameters of the neural network in such a form, that the best generalization performance of the network is obtained. Some guidelines for tuning of these parameters can be found, e.g., in [51]. A summary of the used hyperparameters for the proposed NN with an informal description of the influence of higher values for some hyperparameters is given in Tab. 5.2. The choice of the learning rate is

Hyperparameter	**Influence of a higher value**	**Applied value**
Learning rate (Adam)	faster convergence, poor local convergence and a higher chance of global divergence	$1 \cdot 10^{-4}$
Decay of learning rate	lower learning rate for later epochs with better local, but weaker global convergence	0.15
Bath size	lower variance of the gradient	32
Number of filter coefficients	higher capacity, longer history	50
Number of layers	higher capacity, longer history	5
Number of neurons	higher capacity	80
Regularization strength	lower capacity	$1 \cdot 10^{-4}$
Decay rate (TC kernel)	faster local responses, lower capacity	0.93

Table 5.2: Applied values of hyperparameters and influence of higher values for important hyperparameters.

critical for the determination of the best performing network structure. A significant problem for the tuning of the learning rate is divergence of the model. The higher the learning rate is chosen, the higher is the risk that the model will not converge.

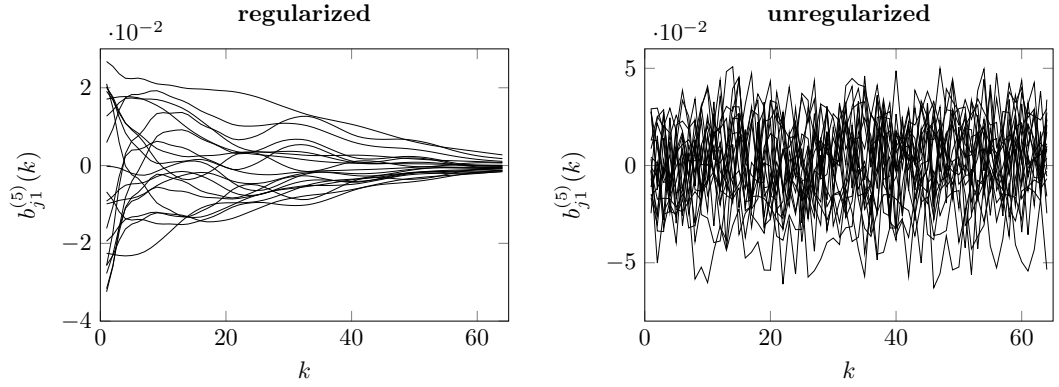

Figure 5.9: Filter coefficients of the last convolutional ($n_u = 5$) for the first output from the inputs $j = 1, \dots 20$. The left plot shows a regularized NN with $n_n = 80$ and the right side shows an unregularized NN with $n_n = 60$.

However, there seems to be a relation that the model performance becomes better, the higher the learning rate is chosen. This relation is problematic since, for an investigation of the hyperparameters, it is advantageous to try out as many possibilities as possible without the need to redo the computation, if the NN has not converged. This behavior has also been observed for other types of problems [51] and occurs in nonlinear system identification too. Another good trick for the learning rate is to scale it sequentially according to

$$\eta^{(n_e)} = \frac{\eta_0}{\sqrt{1 + \kappa^{n_e}}} \tag{5.27}$$

with n_e denoting the number of the epoch, $\eta^{(n_e)}$ the learning rate for this epoch, η_0 denoting the initial learning rate, and κ the decay factor. This results in an exponentially decaying learning rate [60]. For the NN described here, $\kappa = 0.15$, and $\eta_0 = 1 \cdot 10^{-3}$ has been used.

In Fig. 5.9 some filter coefficients of the last layer for the regularized Deep FIR NN and the last layer for an unregularized Deep FIR NN are shown. It can be clearly seen that regularization changes the chain of filter coefficients significantly. It is surprising to see that also for the unregularized case a high performance on the test data is achieved. One explanation for this is that the spiky behavior of the filter coefficients is avoided in the mean.

In Tab. 5.3, the results for the official test datasets of the Bouc-Wen benchmark are summarized. Other methods with strong performance, especially the polynomial nonlinear state space models (PNLSS) [93, 38], local model state space networks (LMSSN) [114], or recurrent NN structures [115] all contain feedback. There is, for

e_{RMS} multisine $[\times 10^{-5}\text{m}]$	e_{RMS} sinesweep $[\times 10^{-5}\text{m}]$	Description (Num. of parameters)	Ref.
PNLSS Models			
1.34	1.12	Decoupled (51)	[38]
1.87	1.20	MIMO / linear $[2-3]$ (90)	[38]
5.42	-	MIMO / linear [2] (34)	[93]
3.15	-	MIMO / linear $[2-4]$ (109)	[93]
1.27	-	MIMO / linear $[2-7]$ (364)	[93]
1.21	-	MIMO / linear [3 5 7] (217)	[93]
Other Models			
17.0	13.8	LMN with NARX	[11]
16.4	17.2	regularized LMN FIR	[11]
31.2	24.9	LMN with OBF	[11]
7.9	11	Stochastic Subspace	[9]
5.3	1.5	NARX Sigmoidal (1571)	[131]
5.7	1.9	Decoupled NARX (151)	[131]
8.76	6.39	Volterra feedback	[113]
468	18.6	Nelder-Mead	[18]
468	19.0	NOMAD	[18]
Recurrent NN			
2.8	5.98	LSTM (3 layers)	[114]
7.6	4.1	ReLU RNN	[114]
LMSSN Models			
4.70	2.45	MISO / MISO, 18 Splits (110)	[115]
3.32	1.76	MISO / affine, 21 Splits (125)	[115]
2.66	2.36	MIMO / MISO, 9 Splits (125)	[115]
2.83	2.30	MIMO / affine, 7 Splits (135)	[115]
Convolutional NN			
2.43	1.73	Deep RFIR NN	
3.21	2.97	Deep FIR (without regularization) NN	

Table 5.3: Comparison of selected methods on Bouc-Wen benchmark [115]

these types of systems, no stability guarantee. It is often the case that the stability of these models depends on the applied input. Especially for polynomial systems, instabilities are the reason that no values can be given for the performance of several PNLSS models on the sinesweep test system.

Convolutional NNs do not suffer from instability issues. In Tab. 5.3, the performance values for the best deep RFIR NN is described as well. For completeness, the deep FIR (without regularization) is listed as well. The RFIR NN is able to achieve a comparable test error on both the multisine and the sinesweep test dataset. The deep RFIR NN achieves comparable performance on the dataset. It is remarkable that, in contrast to the polynomial state space systems which are unstable in many scenarios, stability is guaranteed by the structure of the RFIR NN.

There are, however, drawbacks. The need for validation data is difficult to avoid, and finding the best performing structure is often more an *art* than science. Despite the fact that the validation error offers a justification of the found structure, the process of obtaining this structure is not straightforward. Finding an appropriate strategy to automatize this process is an interesting question for further research. Another drawback is the relatively high number of data required to train the network. The RFIR NN is thus only recommended as an option for nonlinear modeling if a high number of samples ($> 100\,000$) are available.

6 Conclusion

The goal of this thesis is the introduction of novel techniques for regularization of both linear and nonlinear FIR models.

6.1 Summary

To develop these techniques, the foundations of system identification are described in Chap. 2. The statistical perspective of maximum likelihood and Bayesian methods are introduced. Then, the state-of-the-art approach for the regularized identification of FIR models is discussed. The choice of appropriate criteria for model complexity selection, including information criteria like Akaike's information criterion or bounds based on the VC-dimension, are discussed. Techniques for dynamic system identification with FIR models offer the compelling advantage that stability is guaranteed. Their greatest weakness is the high number of parameters required.

In the following chapters of the thesis, three different approaches for the identification of dynamic systems are developed. All of these methods mitigate the variance error due to the high number of parameters by application of an appropriate regularization term.

The first technique, described in Chap. 3, which uses impulse response preserving (IRP) matrices to regularize the impulse response of linear systems, offers the ability to integrate prior knowledge from a signals and systems perspective. In this approach, the impulse response is forced to be similar an a priori assumed behavior. The method has been applied to numerical benchmark studies and a laboratory pendulum example.

The second procedure, described in Chap. 4, extends the regularized identification approach of impulse responses to nonlinear systems within the framework of local model networks (LMNs). Due to the separation of scheduling variables and variables for the local models, the Regularized FIR Local Model Network (RFIR LMN) is able

to fit high dimensional and locally regularized FIR models as local models, while the dimensionality of the scheduling variables is kept low. This identification algorithm is applied to a challenging nonlinear numerical process and a simulated Diesel engine. Especially in the high noise case, the RFIR LMN can improve performance considerably.

The third method, the deep Regularized FIR Neural Network (RFIR NN), described in Chap. 5, combines convolutional NNs with the regularization approach for linear systems. It allows for building a deep neural network for the identification of nonlinear dynamic systems. This approach is applied to the Bouc-Wen benchmark example and achieves state-of-the-art results while additionally guaranteeing stability for arbitrary input signals. The advantages and disadvantages of the three methods, introduced in Chaps. 3-5, are summarized in Tab. 6.1.

Property	**IRP RFIR**	**RFIR LMN**	**Deep RFIR NN**
Complexity	−	o	+
Computational Effort (Training)	++	+	−−
Computational Effort (Evaluation)	++	+	−
Consideration of Prior Knowledge	++	+	+
Extension to MIMO	+	+	+
Extrapolation Behavior	+	+	+
Noise Tolerance	++	+	+
Required Amount of Data	++	+	o
Stability	++	+	++
Suitability for Real-Time Systems	++	+	−
Tuning Effort	++	+	−
++: very favorable model properties, −−: very undesirable model properties			

Table 6.1: Comparison of different properties of the proposed methods.

The comparison shows that the simplest model, the IRP RFIR model, posses advantageous properties in several areas. The major drawback is that the model complexity, and thus its capability is limited to linear systems. In consequence, performance can be limited compared to nonlinear methods. The comparison of the two nonlinear approaches, the RFIR LMN and the Deep RFIR NN, shows that the complexity of the RFIR LMN is smaller than the complexity of the Deep RFIR NN. Thus, the performance of the Deep RFIR NN is better for highly nonlinear and dynamically complex processes as the Bouc-Wen example from Sect. 5.3. The drawbacks of the Deep RFIR NN are its computation effort required for training and for evaluation (due to the high number of parameters and high dynamic order due to its deep structure).

This also hinders its suitability for real-time systems, which require a fast model. The largest disadvantage is the tuning effort required. While the RFIR LMN works out of the box and estimates both the regularization strength and model complexity automatically, the Deep RFIR NN requires substantial tuning of hyperparameters for architecture, optimization algorithm, and regularization strength. The usage of the Deep RFIR NN is recommended only if a large amount of samples ($> 100\,000$) is available, and high accuracy is needed.

6.2 Outlook

Impulse Response Preserving FIR Identification: The linear case for kernel and penalty based regularization is well understood. The main topics lie in the extension of these methods to nonlinear problems. A possible line of research can be online learning approaches for both the linear parameters and the hyperparameters. This would allow tracking of time varying systems.

Another interesting research direction is the investigation, whether non-causal systems can be employed to identify unstable causal systems. For this procedure, the identification could benefit from regularization, too. Representation of systems with pure integration are a concern for this approach, so a procedure to deal with this type of systems is to be developed.

Regularized Local FIR Networks: For LMNs with local FIR models, there are also several possible lines of further research. In this contribution, axis-orthogonal splitting schemes have been investigated. The flexibility of LMNs can be increased by allowing axis-oblique splits. To enable this kind of network to be learned, one challenge is the learning method for the parameter estimation applied. With a change in the orientation of the split, the validity function changes too. This change requires a re-estimation of the linear parameters and in the case of FIR for the hyperparameters as well. The way this estimation algorithm is to be designed is an exciting line of further research.

The separation of nonlinear scheduling variables and linear parameters for the local models is a unique feature of LMNs. The optimal choice of these variables is often problem specific. Thus, the identification algorithm could benefit from a novel automatic procedure for the selection of the variables for these input spaces. It is further possible to select filtered versions of the input (e.g. by orthogonal basis functions)

as scheduling variables. It remains to be investigated whether such a choice can influence the identification result positively.

Another appealing topic is the further integration of physically available prior knowledge, similar to the linear case with IRP matrices. It could be possible to linearize an a priori available nonlinear system at the operation point corresponding to the center of the local model and use this linearized model as a means for regularization.

Deep Regularized FIR Neural Networks: Deep learning offers increased performance in the mid- and big data regimes. It remains, however, still dubious why very deep methods allow for such good generalization performance. This will be a topic for future research, not only for system identification but in context of deep neural networks in general.

Extrapolation behavior of models is an issue of great concern for real-world problems. Thus, analysis and comparison of extrapolation behavior is a possible research direction. Neural networks based on an FIR structure are guaranteed to have stable extrapolation behavior. However, this hold for other structures, like the LMNs, too. Comparing their extrapolation behavior could reveal unknown advantages of either of the two methods.

The choice of optimal hyperparameters for neural networks is still an open topic. Thus, for the regularized deep FIR networks, advanced techniques for tuning of the regularization parameters are of significant importance. Using advanced methods of regularization often requires an increasing number of fiddle parameters. Systematic approaches from neural architecture search can provide ways to simplify the training and extend the capability of the presented approach.

Finally, it is expected that system identification can significantly benefit from the progress being made in the machine learning domain. A thorough understanding of the properties of dynamic systems, especially stability, will remain a substantial requirement for the application of the methods in reality.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[2] Janos Abonyi, Robert Babuska, and Ferenc Szeifert. Modified gath-geva fuzzy clustering for identification of takagi-sugeno fuzzy models. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 32(5):612–621, 2002.

[3] Yaser S Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning from data*, volume 4. AMLBook New York, NY, USA:, 2012.

[4] Hirotugu Akaike. Fitting autoregressive models for prediction. *Annals of the institute of Statistical Mathematics*, 21(1):243–247, 1969.

[5] Hirotugu Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

[6] Hirotuguike Akaike. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, 1973.

[7] Carl Andersson, Antônio H Ribeiro, Koen Tiels, Niklas Wahlström, and Thomas B Schön. Deep convolutional networks in system identification. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 3670–3676. IEEE, 2019.

[8] Carl Andersson, Niklas Wahlström, and Thomas B Schön. Data-driven impulse response regularization via deep learning. *IFAC-PapersOnLine*, 51(15):1–6, 2018.

[9] Anela Bajric. System identication of a linearized hysteretic system using covariance driven stochastic subspace identication. In *Workshop on Nonlinear System Identification Benchmarks*, 2016.

[10] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[11] Julian Belz, Tobias Münker, Tim O Heinz, Geritt Kampmann, and Oliver Nelles. Automatic modeling with local model networks for benchmark processes. *IFAC-PapersOnLine*, 50(1):470–475, 2017.

[12] Yoshua Bengio and Yves Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *Journal of machine learning research*, 5(Sep):1089–1105, 2004.

[13] Torsten Bohlin. A case study of grey box identification. *Automatica*, 30(2):307–318, 1994.

[14] Giulio Bottegal and Gianluigi Pillonetto. Regularized spectrum estimation using stable spline kernels. *Automatica*, 49(11):3199–3209, 2013.

[15] Stephen Boyd and L. Vandenberghe. *Introduction to Applied Linear Algebra*. Stanford University, 2017.

[16] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[17] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and Regression Trees*. CRC Press, New York, 1999.

[18] Mathieu Brunot, Alexandre Janot, and Francisco Carrillo. Continuous-time nonlinear systems identification with output error method based on derivative-free optimisation. *IFAC-PapersOnLine*, 50(1):464–469, 2017.

[19] Kenneth P Burnham and David R Anderson. *Model selection and multimodel inference: a practical information-theoretic approach*. Springer Science & Business Media, 2003.

[20] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2007.

[21] Francesca Paola Carli, Tianshi Chen, and Lennart Ljung. Maximum entropy kernels for system identification. *Automatic Control, IEEE Transactions on*, 62(3):1471–1477, 2017.

[22] S Chen and SA Billings. Representations of non-linear systems: the narmax model. *International Journal of Control*, 49(3):1013–1032, 1989.

[23] Tianshi Chen. On kernel design for regularized LTI system identification. *arXiv preprint arXiv:1612.03542*, 2016.

[24] Tianshi Chen, Martin S Andersen, Lennart Ljung, Alessandro Chiuso, and Gianluigi Pillonetto. System identification via sparse multiple kernel-based regularization using sequential convex optimization techniques. *Automatic Control, IEEE Transactions on*, 59(11):2933–2945, 2014.

[25] Tianshi Chen, Tohid Ardeshiri, Francesca P Carli, Alessandro Chiuso, Lennart Ljung, and Gianluigi Pillonetto. Maximum entropy properties of discrete-time first-order stable spline kernel. *Automatica*, 66:34–38, 2016.

[26] Tianshi Chen and Lennart Ljung. Implementation of algorithms for tuning parameters in regularized least squares problems in system identification. *Automatica*, 49(7):2213–2220, 2013.

[27] Tianshi Chen and Lennart Ljung. Regularized system identification using orthonormal basis functions. *arXiv preprint arXiv:1504.02872*, 2015.

[28] Tianshi Chen, Henrik Ohlsson, and Lennart Ljung. On the estimation of transfer functions, regularizations and Gaussian processes-revisited. *Automatica*, 48(8):1525–1535, 2012.

[29] Vladimir Cherkassky, Xuhui Shao, Filip M Mulier, and Vladimir N Vapnik. Model complexity control for regression using VC generalization bounds. *IEEE transactions on Neural Networks*, 10(5):1075–1089, 1999.

[30] A. Chiuso and G. Pillonetto. System identification: A machine learning perspective. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):281–304, 2019.

[31] Alessandro Chiuso and Gianluigi Pillonetto. A Bayesian approach to sparse dynamic network identification. *Automatica*, 48(8):1553–1565, 2012.

[32] Harald Cramér. *Mathematical methods of statistics*, volume 1. Princeton university press, 1946.

[33] Mohamed Darwishy, Gianluigi Pillonetto, and Roland Toth. Perspectives of orthonormal basis functions based kernels in Bayesian system identification. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 2713–2718, Dec 2015.

[34] Enno de Boer, Helena Leurent, and Adrian Widmer. Lighthouse' manufacturers lead the way—can the rest of the world keep up? *McKinsey Quarterly*, 1:1–8, January 2019.

[35] G De Nicolao and G Ferrari Trecate. Consistent identification of NARX models via regularization networks. *Automatic Control, IEEE Transactions on*, 44(11):2045–2049, 1999.

[36] Francesco Dinuzzo. Kernels for linear time invariant system identification. *SIAM Journal on Control and Optimization*, 53(5):3299–3317, 2015.

[37] Diego Eckhard, Alexandre S Bazanella, Cristian R Rojas, and Håkan Hjalmarsson. On the convergence of the prediction error method to its global minimum. *IFAC Proceedings Volumes*, 45(16):698–703, 2012.

[38] Alireza Fakhrizadeh Esfahani, Philippe Dreesen, Koen Tiels, Jean-Philippe Noël, and Johan Schoukens. Parameter reduction in nonlinear state-space identification of hysteresis. *Mechanical Systems and Signal Processing*, 104:884–895, 2018.

[39] Ronald A Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 222(594-604):309–368, 1922.

[40] Ronald Aylmer Fisher. Theory of statistical estimation. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 22, pages 700–725. Cambridge University Press, 1925.

[41] Ronald Aylmer Fisher. Two new properties of mathematical likelihood. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 144(852):285–307, 1934.

[42] Marco Forgione and Dario Piga. dynoNet: a neural network architecture for learning dynamical systems. *arXiv preprint arXiv:2006.02250*, 2020.

[43] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

[44] Roger Frigola, Yutian Chen, and Carl Edward Rasmussen. Variational gaussian process state-space models. In *Advances in neural information processing systems*, pages 3680–3688, 2014.

[45] Roger Frigola, Fredrik Lindsten, Thomas B Schön, and Carl Edward Rasmussen. Bayesian inference and learning in Gaussian process state-space models with particle MCMC. In *Advances in Neural Information Processing Systems*, pages 3156–3164, 2013.

[46] Carl Friedrich Gauss. *Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections: A Translation of Gauss Theoria Motus. With an Appendix.* Little, Brown and Company, 1857.

[47] Zoubin Ghahramani and Sam T Roweis. Learning nonlinear dynamical systems using an EM algorithm. *Advances in neural information processing systems*, pages 431–437, 1999.

[48] Torkel Glad and Lennart Ljung. *Control theory.* CRC press, 2014.

[49] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[50] Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[51] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT Press, 2016.

[52] Benjamin Hartmann. *Lokale Modellnetze zur Identifikation und Versuchsplanung nichtlinearer Systeme.* PhD thesis, Universitätsbibliothek der Universität Siegen, 2014.

[53] Tim Oliver Heinz and Oliver Nelles. Efficient pole optimization of nonlinear laguerre filter models. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7. IEEE, 2016.

[54] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[55] Rolf Isermann. *Mechatronic systems: fundamentals*. Springer Science & Business Media, 2007.

[56] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[57] Chi Jin, Praneeth Netrapalli, Rong Ge, Sham M Kakade, and Michael I Jordan. On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. *arXiv preprint arXiv:1902.04811*, 2019.

[58] Michael I. Jordan. Artificial intelligence—the revolution hasn't happened yet. *Harvard Data Science Review*, 6 2019. https://hdsr.mitpress.mit.edu/pub/wot7mkc1.

[59] George S Kimeldorf and Grace Wahba. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41(2):495–502, 1970.

[60] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[61] Juš Kocijan, Agathe Girard, Blaž Banko, and Roderick Murray-Smith. Dynamic systems identification with Gaussian processes. *Mathematical and Computer Modelling of Dynamical Systems*, 11(4):411–424, 2005.

[62] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[63] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[64] Fredrik Lindsten, Thomas B Schön, et al. Backward simulation methods for monte carlo statistical inference. *Foundations and Trends® in Machine Learning*, 6(1):1–143, 2013.

[65] Thomas Lipp and Stephen Boyd. Variations and extension of the convex–concave procedure. *Optimization and Engineering*, 17(2):263–287, 2016.

[66] Lennart Ljung. Convergence analysis of parametric identification methods. *IEEE transactions on automatic control*, 23(5):770–783, 1978.

[67] Lennart Ljung. *System identification*, volume 2. Prentice-Hall, 1999.

[68] Lennart Ljung. *Control theory: multivariable and nonlinear methods*. Taylor & Francis, 2000.

[69] Lennart Ljung. Approaches to identification of nonlinear systems. 2010.

[70] Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.

[71] Lennart Ljung and Tianshi Chen. Convexity issues in system identification. In *Control and Automation (ICCA), 2013 10th IEEE International Conference on*, pages 1–9. IEEE, 2013.

[72] Lennart Ljung and Tianshi Chen. What can regularization offer for estimation of dynamical systems? In *Adaptation and Learning in Control and Signal Processing*, volume 11, pages 1–8, 2013.

[73] Anna Marconato and Maarten Schoukens. Tuning the hyperparameters of the filter-based regularization method for impulse response estimation. *IFAC-PapersOnLine*, 50(1):12841–12846, 2017.

[74] Anna Marconato, Maarten Schoukens, and Johan Schoukens. Filter-based regularisation for impulse response modelling. *IET Control Theory & Applications*, 11(2):194–204, 2016.

[75] David A McAllester. Some pac-bayesian theorems. *Machine Learning*, 37(3):355–363, 1999.

[76] Biqiang Mu, Tianshi Chen, and Lennart Ljung. Asymptotic properties of generalized cross validation estimators for regularized system identification. *IFAC-PapersOnLine*, 51(15):203–208, 2018.

[77] Tobias Münker, Julian Belz, and Oliver Nelles. Improved incorporation of prior knowledge for regularized FIR model identification. In *American Control Conference (ACC)*, pages 1090–1095. IEEE, 2018.

[78] Tobias Münker, Geritt Kampmann, Max Schüssler, and Oliver Nelles. System identification and control of a polymer reactor. In *IFAC 20th World Congress, Berlin*, 2020.

[79] Tobias Münker and Oliver Nelles. Local model network with regularized MISO finite impulse response models. In *IEEE World Congress on Computational Intelligence*, 2016.

[80] Tobias Münker and Oliver Nelles. Nonlinear system identification with regularized local FIR model networks. In *4th IFAC International Conference on Intelligent Control and Automation Science*, 2016.

[81] Tobias Münker and Oliver Nelles. Generalizing piecewise affine system identification to local model networks. In *IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, 2017.

[82] Tobias Münker and Oliver Nelles. Nonlinear system identification with regularized local fir model networks. *Engineering Applications of Artificial Intelligence*, 67:345–354, 2018.

[83] Tobias Münker and Oliver Nelles. Sensitive order selection via identification of regularized fir models with impulse response preservation. *IFAC-PapersOnLine*, 51(15):197–202, 2018.

[84] Tobias Münker, Timm J Peter, and Oliver Nelles. Gray-box identification with regularized FIR models. *at-Automatisierungstechnik*, 66(9):704–713, 2018.

[85] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[86] Roderick Murray-Smith. *A local model network approach to nonlinear modelling*. PhD thesis, University of Strathclyde, 1994.

[87] Roderick Murray-Smith and T Johansen. *Multiple model approaches to nonlinear modelling and control*. CRC press, 1997.

[88] O Nelles. Orthonormal basis functions for nonlinear system identification with local linear model trees (lolimot). In *Proc. IFAC Symposium on System Identification, Kitakyushu, Fukuoka, Japan*, 1997.

[89] Oliver Nelles. LOLIMOT-Lokale, lineare Modelle zur Identifikation nichtlinearer, dynamischer Systeme. *at-Automatisierungstechnik*, 45(4):163–174, 1997.

[90] Oliver Nelles. *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer Science & Business Media, 2001.

[91] Oliver Nelles. Axes-oblique partitioning strategies for local model networks. In *IEEE International Symposium on Intelligent Control*, pages 2378–2383, 2006.

[92] Brett Ninness and Fredrik Gustafsson. A unifying construction of orthonormal bases for system identification. *Automatic Control, IEEE Transactions on*, 42(4):515–521, 1997.

[93] Jean-Philippe Noël, A Fakhrizadeh Esfahani, Gaetan Kerschen, and Johan Schoukens. A nonlinear state-space approach to hysteresis identification. *Mechanical Systems and Signal Processing*, 84:171–184, 2017.

[94] JP Noël and M Schoukens. Hysteretic benchmark with a dynamic nonlinearity. In *Workshop on nonlinear system identification benchmarks*, pages 7–14, 2016.

[95] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

[96] Gianluigi Pillonetto and Alessandro Chiuso. Tuning complexity in regularized kernel-based regression and linear system identification: The robustness of the marginal likelihood estimator. *Automatica*, (58):106–117, 2015.

[97] Gianluigi Pillonetto, Alessandro Chiuso, and Giuseppe De Nicolao. Regularized estimation of sums of exponentials in spaces generated by stable spline kernels. In *American Control Conference (ACC)*, pages 498–503. IEEE, 2010.

[98] Gianluigi Pillonetto, Alessandro Chiuso, and Giuseppe De Nicolao. Prediction error identification of linear systems: a nonparametric Gaussian regression approach. *Automatica*, 47(2):291–305, 2011.

[99] Gianluigi Pillonetto and Giuseppe De Nicolao. A new kernel-based approach for linear system identification. *Automatica*, 46(1):81–93, 2010.

[100] Gianluigi Pillonetto, Francesco Dinuzzo, Tianshi Chen, Giuseppe De Nicolao, and Lennart Ljung. Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, 50(3):657–682, 2014.

[101] Gianluigi Pillonetto, Minh Ha Quang, and Alessandro Chiuso. A new kernel-based approach for nonlinear system identification. *Automatic Control, IEEE Transactions on*, 56(12):2825–2840, 2011.

[102] S Joe Qin and TJ McAvoy. Nonlinear FIR modeling via a neural net PLS approach. *Computers & chemical engineering*, 20(2):147–159, 1996.

[103] Carl Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2005.

[104] Cristian R Rojas and Håkan Hjalmarsson. Sparse estimation based on a validation criterion. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 2825–2830. IEEE, 2011.

[105] Cristian R Rojas, Patricio E Valenzuela, and Ricardo A Rojas. A critical view on benchmarks based on randomly generated systems. *IFAC-PapersOnLine*, 48(28):1471–1476, 2015.

[106] Cristian R Rojas, Bo Wahlberg, and Hakan Hjalmarsson. A sparse estimation technique for general model structures. In *Control Conference (ECC), 2013 European*, pages 2410–2414. IEEE, 2013.

[107] Jürgen Schmidhuber. New millennium AI and the convergence of history. In *Challenges for computational intelligence*, pages 15–35. Springer, 2007.

[108] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[109] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[110] Thomas B Schön, Adrian Wills, and Brett Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

[111] Johan Schoukens, Rik Pintelon, T Dobrowiecki, and Yves Rolain. Identification of linear systems with nonlinear distortions. *Automatica*, 41(3):491–504, 2005.

[112] Maarten Schoukens and Jean Philippe Noël. Three benchmarks addressing open challenges in nonlinear system identification. *IFAC-PapersOnLine*, 50(1):446–451, 2017.

[113] Maarten Schoukens and F Griesing Scheiwe. Modeling nonlinear systems using a volterra feedback model. In *Workshop on Nonlinear System Identification Benchmarks*, 2016.

[114] Max Schüssler, Tobias Münker, and Oliver Nelles. Deep recurrent neural networks for nonlinear system identification. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 448–454. IEEE, 2019.

[115] Max Schüssler, Tobias Münker, and Oliver Nelles. Local model networks for the identification of nonlinear state space models. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 6437–6442. IEEE, 2019.

[116] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[117] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[118] Stephen M Stigler et al. The epic story of maximum likelihood. *Statistical Science*, 22(4):598–620, 2007.

[119] Petre Stoica and Yngve Selen. Model-order selection: a review of information criterion rules. *Signal Processing Magazine, IEEE*, 21(4):36–47, 2004.

[120] Freek Stulp and Olivier Sigaud. Many regression algorithms, one unified model - a review. *Neural Networks*, 2015.

[121] Nariaki Sugiura. Further analysts of the data by Akaike's information criterion and the finite corrections. *Communications in Statistics-Theory and Methods*, 7(1):13–26, 1978.

[122] Andreas Svensson and Thomas B Schön. A flexible state–space model for learning nonlinear dynamical systems. *Automatica*, 80:189–199, 2017.

[123] Tomohiro Takagi and Michio Sugeno. Fuzzy identification of systems and its applications to modeling and control. *Systems, Man and Cybernetics, IEEE Transactions on*, (1):116–132, 1985.

[124] Harry Trentelman, Anton A Stoorvogel, and Malo Hautus. *Control theory for linear systems*. Springer Science & Business Media, 2012.

[125] Patricio E Valenzuela, Thomas B Schön, and Cristian R Rojas. On model order priors for Bayesian identification of SISO linear systems. *International Journal of Control*, pages 1–17, 2017.

[126] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2000.

[127] Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[128] Grace Wahba et al. A comparison of GCV and GML for choosing the smoothing parameter in the generalized spline smoothing problem. *The Annals of Statistics*, 13(4):1378–1402, 1985.

[129] Johan Wahlström and Lars Eriksson. Modelling diesel engines with a variable-geometry turbocharger and exhaust gas recirculation by optimization of model parameters for capturing non-linear system dynamics. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 225(7):960–986, 2011.

[130] Larry Wasserman. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media, 2013.

[131] David T Westwick, Gabriel Hollander, Kiana Karami, and Johan Schoukens. Using decoupling methods to reduce polynomial NARX models. *IFAC-PapersOnLine*, 51(15):796–801, 2018.

[132] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[133] Hao Ying. General siso takagi-sugeno fuzzy systems with linear rule consequent are universal approximators. *Fuzzy Systems, IEEE Transactions on*, 6(4):582–587, 1998.

[134] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[135] Mattia Zorzi and Alessandro Chiuso. Sparse plus low rank network identification: A nonparametric approach. *Automatica*, 76:355–366, 2017.