Linköping studies in science and technology. Dissertations.
No. 1351

Regularization for Sparseness and Smoothness

Applications in System Identification and Signal Processing

Henrik Ohlsson

Department of Electrical Engineering
Linköping University, SE–581 83 Linköping, Sweden

Linköping 2010

Linköping studies in science and technology. Dissertations.
No. 1351

Regularization for Sparseness and Smoothness – Applications in System Identification and Signal Processing

Henrik Ohlsson

ohlsson@isy.liu.se
www.control.isy.liu.se
Division of Automatic Control
Department of Electrical Engineering
Linköping University
SE–581 83 Linköping
Sweden

ISBN 978-91-7393-287-5 ISSN 0345-7524

Printed by LiU-Tryck, Linköping, Sweden 2010

To family and friends!

Abstract

In system identification, the *Akaike Information Criterion* (AIC) is a well known method to balance the model fit against model complexity. Regularization here acts as a price on model complexity. In statistics and machine learning, regularization has gained popularity due to modeling methods such as *Support Vector Machines* (SVM), *ridge regression* and *lasso*. But also when using a *Bayesian* approach to modeling, regularization often implicitly shows up and can be associated with the prior knowledge. Regularization has also had a great impact on many applications, and very much so in clinical imaging. In *e.g.,* breast cancer imaging, the number of sensors is physically restricted which leads to long scan times. Regularization and sparsity can be used to reduce that. In *Magnetic Resonance Imaging* (MRI), the number of scans is physically limited and to obtain high resolution images, regularization plays an important role.

Regularization shows-up in a variety of different situations and is a well known technique to handle ill-posed problems and to control for overfit. We focus on the use of regularization to obtain sparseness and smoothness and discuss novel developments relevant to system identification and signal processing.

In regularization for sparsity a quantity is forced to contain elements equal to zero, or to be sparse. The quantity could *e.g.,* be the regression parameter vector of a linear regression model and regularization would then result in a tool for variable selection. Sparsity has had a huge impact on neighboring disciplines, such as machine learning and signal processing, but rather limited effect on system identification. One of the major contributions of this thesis is therefore the new developments in system identification using sparsity. In particular, a novel method for the estimation of segmented ARX models using regularization for sparsity is presented. A technique for piecewise-affine system identification is also elaborated on as well as several novel applications in signal processing. Another property that regularization can be used to impose is smoothness. To require the relation between regressors and predictions to be a smooth function is a way to control for overfit. We are here particularly interested in regression problems with regressors constrained to limited regions in the regressor-space *e.g.,* a manifold. For this type of systems we develop a new regression technique, *Weight Determination by Manifold Regularization* (WDMR). WDMR is inspired by applications in biology and developments in manifold learning and uses regularization for smoothness to obtain smooth estimates. The use of regularization for smoothness in linear system identification is also discussed.

The thesis also presents a real-time *functional Magnetic Resonance Imaging* (fMRI) bio-feedback setup. The setup has served as proof of concept and been the foundation for several real-time fMRI studies.

Populärvetenskaplig sammanfattning

Modeller används inom de flesta områden för att efterlikna verkligheten. Anledningarna kan vara allt ifrån att det är fysikaliskt omöjligt till att det är kostsamt att utföra experimenten och därför utförs dessa på en modell istället. En modell kan också användas till att generalisera och förutse beteenden för nya situationer. Vi använder exempelvis en mental modell för cykling för att från tidigare erfarenheter kunna hantera nya situationer.

I denna avhandling studeras matematiska modeller. Framför allt diskuteras en teknik för att framkalla egenskaper så som *gleshet* och *glatthet* hos modellparametrar och skattningar. Denna teknik betecknas *regularisering*. Varför är man då intresserad av att framkalla dessa egenskaper? Gleshet kan vara av nytta för att välja ut mätstorheter som man bör fortsätta att mäta om man vill bibehålla goda skattningsresultat. Om det är kostsamt att mäta kan denna användning vara värdefull. Gleshet har också visats användbart vid medicinsk bildbehandling för till exempel minskning av röntgentider. I denna avhandling används regularisering för gleshet på problem inom områdena *systemidentifiering* och *signalbehandling*. Bland annat diskuteras hur regularisering för gleshet kan användas för att upptäcka plötsliga förändringar. Glatthet är i många fall motiverat av fysikaliska skäl. Många signaler som är intressanta att modellera och förutse beter sig på ett mjukt och kontinuerligt sätt. Det finns därför skäl till att modellen som används även har dessa egenskaper. Ett av resultaten i denna avhandling är en ny modelleringsmetod, *Weight Determination by Manifold Regularization* (WDMR). Ett specifikt användningsområde som diskuteras är skattning av vattentemperatur från mätningar av den kemiska sammansättningen i musselskal. Antagandet att det finns ett glatt samband mellan den kemiska koncentrationen i musselskalet och temperaturen är här viktigt för bra skattningar.

Ett annat område som berörs i avhandlingen är mätning av hjärnaktivitet. Mer specifikt presenteras en praktisk uppställning för att mäta och tolka hjärnaktivitet i realtid.

Acknowledgments

I have enjoyed my PhD studies a lot! There are a number of reasons for that. First of all, I have had a great supervisor. My supervisor, Professor Lennart Ljung, has guided and inspired me through out the years of my PhD. Thank you Lennart, you been outstanding! Dr. Jacob Roll, my assistant supervisor, has also been of great importance. I am very grateful for all our discussions and for all the help you given me. Ulla Salaneck and Åsa Karmelind have also been invaluable. Thank you!

Secondly, the automatic control group at Linköping University is beyond ordinary. It is a creative, friendly, environment, and an excellent place for PhD studies. In particular I would like to thank Dr. Umut Orguner for all your help, interesting discussions and for being so kind! Also, thank you Dr. Tianshi Chen for interesting discussions and a nice collaboration. Thank you Lic Christian Lundquist, Dr. Ragnar Wallin, Zoran Sjanic, Lic Christian Lyzell, Daniel Petersson, Karl Granström, Lic Daniel Ankelhed and Patrik Axelsson for all the time away, moules frites, early mornings, balconies and Kalles kaviar. Windsurfing and kite people, Dr. Henrik Tidefelt, Dr. Johan Sjöberg, Dr. David Törnqvist, Tohid Ardeshiri, André Carvalho Bittencourt, Fredrik Lindsten, Dr. Emre Özkan and Sina Khoshfetrat Pakazad, it has been lots of fun! Office mates, Lic Johanna Wallén and Jonas Callmer, thanks a lot for your company and happy Fridays! Thank you Dr. Thomas Schön and Dr. Gustaf Hendeby for your company at Campushallen. Professor Fredrik Gustafsson and Professor Torkel Glad, thank you for great courses and collaborations!

External collaborators, thank you to Professor Anders Ynnerman, Professor Hans Knutsson, Dr. Mats Andersson, Dr. Joakim Rydell, Dr. Anders Brun, Lic Anders Eklund and Tan Khoa Nguyen for the collaboration within the MOVIII project. Thank you Dr. Carl Edward Rasmussen at University of Cambridge for a very nice stay in Cambridge. Professor Stephen Boyd, Maite Bauwens, Dr. Marc Deisenroth and Tillmann Falck, thank you for interesting discussions and good collaborations.

This thesis has been proofread by Professor Lennart Ljung, Patrik Axelsson, Daniel Petersson, Dr. David Törnqvist, Dr. Umut Orguner, Dr. Mehmet Guldogan, sister Pernilla Ohlsson and Dr. Thomas Schön. Thank you for your comments! Also thanks to Dr. Gustaf Hendeby, Dr. Henrik Tidefelt and Dr. David Törnqvist for LaTeX support.

Noelia! You been and are my love. You made me laugh and you made me happy. Thanks a lot for your patience! My family gets a lot of love and gratefulness too. You have always been there for me, even though I have been away. Thank you! Also thank you to friends from Uppsala, Amherst, Linköping, Y2000d, Cambridge for many happy memories!

I am also very grateful for the support from the Strategic Research Center MOVIII and from the Swedish Research Council in the Linnaeus center CADICS. It has been very motivating and I am very glad to have gotten the opportunity to be a part of these research centers.

Linköping, October 2010
Henrik Ohlsson

Contents

II Publications

Notation

Mathematical Symbols

Notation	Meaning
$\mathcal{R}$	set of real numbers
$\mathcal{R}^+$	set of positive real numbers
$\mathcal{Z}$	set of integers
$\mathcal{N}$	set of natural numbers
$\|\cdot\|_p$	ℓ_p-norm
$\|\cdot\|$	absolute value for a scalar and the determinant for a matrix
$dim(x)$	dimension of the vector x
$card(\mathcal{X})$	cardinality of the set $\mathcal{X}$
$rank(X)$	column rank of the matrix X
$sign(\cdot)$	sign function
$tr(\cdot)$	trace
I_n	$n \times n$-dimensional identity matrix
$0_{n\times m}$	$n \times m$-dimensional zero matrix
$1_{n\times m}$	$n \times m$-dimensional matrix of ones
$N(\mu, \sigma^2)$	Gaussian distribution with mean μ and variance σ^2
$N(x; \mu, \sigma^2)$	Gaussian distribution in x with mean μ and variance σ^2
$U(a, b)$	uniform distribution between a and b
$p_e(\cdot)$	probability distribution for e
$\{x_t\}_{t=1}^N$	set containing $x_1, x_2, \ldots, x_N$
E_x	expectation with respect to the random variable x
$\triangleq$	equal by definition
$\in$	belongs to
X^T	transpose of the matrix X
$\dot{x}$	time derivative of x
∇_θ	gradient with respect to θ
$\emptyset$	empty set

$\bigcap$	intersection
$\subset$	proper subset
$\subseteq$	subset
∂_x	subdifferential with respect to x
$k(\cdot, \cdot)$	kernel
ℓ	length-scale of a squared exponential kernel
φ	regressor
y	system output
u	system input
x	state
λ	regularization parameter
$\mathcal{N}_{\mathsf{o}}$	index set associated with the observed data
$\mathcal{N}_{\mathsf{e}}$	index set associated with the estimation-data set
N_{e}	number of elements in the estimation-data set
$\mathcal{N}_{\mathsf{v}}$	index set associated with the validation-data set
N_{v}	number of elements in the validation-data set
$\mathcal{N}_{\mathsf{t}}$	index set associated with the test-data set
N_{t}	number of elements in the test-data set
$f(\varphi, \theta)$	model evaluated at the regressor φ and the regressor parameter θ
$f(\varphi)$	model evaluated at the regressor φ
f_0	system function
T_s	sample time

Abbreviations and Acronyms

Abbreviation	Meaning
AFMM	Adaptive Forgetting by Multiple Models
AIC	Akaike Information Criterion
ARX	Auto-Regressive with eXogenous variables
BCI	Brain Computer Interface
BLUE	Best Linear Unbiased Estimator
BOLD	Blood Oxygen Level Dependent
CCA	Canonical Correlation Analysis
CS	Compressed Sensing
CUSUM	CUmulative SUM
CV	Cross Validation
EKF	Extended Kalman Filter
FDI	Fault Detection and Isolation
FIR	Finite Impulse Response
fMRI	functional Magnetic Resonance Imaging
FOCUSS	FOCal Underdetermined System Solver
GLM	General Linear Modeling
GP	Gaussian Process
GPCA	General Principal Component Analysis

GPR	Gaussian Process Regression
HMM	Hidden Markov Model
i.i.d.	independent and identically distributed
IMM	Interacting Multiple Model
KF	Kalman Filter
KKT	Karush-Kuhn-Tucker
K-NN	K-Nearest Neighbor
LARS	Least Angle Regression
lasso	least absolute shrinkage and selection operator
LLE	Locally Linear Embedding
LQG	Linear-Quadratic-Gaussian
LS	Least-Squares
LS-SVM	Least-Squares Support Vector Machines
LS-SVR	Least-Squares Support Vector Regression
MAE	Mean Absolute Error
MAP	Maximum A Posteriori
MLE	Maximum Likelihood Estimate
MPC	Model Predictive Control
MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
OE	Output Error
PCA	Principal Component Analysis
PEM	Prediction Error Method
PLS	Partial Least Squares
PRBS	Pseudo-Random Binary Sequence
PWA	Piece-Wise Affine
PWARX	Piece-Wise Auto-Regressive with eXogenous variables
PWASON	Piece-Wise Affine system identification using Sum-Of-Norms regularization
RKHS	Reproducing Kernel Hilbert Space
SISO	Single-Input Single-Output
SNR	Signal-to-Noise Ratio
s.t.	subject to
STATESON	STATE estimation by Sum-Of-Norms regularization
SVM	Support Vector Machines
SVR	Support Vector Regression
UAV	Unmanned Aerial Vehicle
UTM	Universal Transverse Mercator
WDMR	Weight Determination by Manifold Regularization
w.p.	with probability
w.r.t.	with respect to

Part I

Background

1 Introduction

1.1 Models and Modeling

Models are used in most scientific disciplines as substitutes for reality. It can be that it is practically impossible to conduct experiments on the physical system and a model thereof is therefore used to replace it. Or it could be that the model is used to generalize to new situations not previously seen.

We humans use models every day, *mental models*. These models are built-up from past experiences and make it possible for us to, *e.g.,* ride our bikes. When we bike, we use our mental model for biking to not fall over. In particular, we need to use previous biking experience to generalize to new situations.

In this thesis, methods for computing models are discussed. Like for a human, most of the models will be based on gathered past observations. We do not summarize these in a mental model, but seek instead a *mathematical model* that can explain these observations. A mathematical model describes a system's behavior using mathematical language. Mathematical language could be a set of differential or difference equations, or it could be a rule for how to combine past observations.

Mental models are of particular use for us and our brain. Mathematical models are not useful for our brain (at least not in the same way as mental models) but of particular interest and use for engineering and science. The two next examples motivate the use of mathematical models. We will return to both of these examples at later phases of this thesis.

Example 1.1: Climate Reconstruction

There exist a number of climate recorders in nature from which the past temperature can be extracted. However, only a few natural archives are able to record climate fluctuations with high enough resolution so that the seasonal variations can be reconstructed. One such archive is a bivalve shell, see Figure 1.1. The chemical composition of a shell of a bivalve depends on a number of chemical and physical parameters of the water in which the shell was composed. Of these parameters, the water temperature is probably the most important one. It should therefore be possible to estimate the water temperature for the time the shell was built, from measurements of the shell's chemical composition. This would *e.g.,* give climatologists the ability to estimate past water temperatures by analyzing ancient shells. To do this, a model for how the chemical composition relates to water temperature would be needed.

Figure 1.1: *Bivalve shell.*

Example 1.2: Model-Based Reference Generation

Flight planning is essential for safety when flying. It makes sure that, on the flight route, the airplane does not get to close to other airplanes, takes into account weather forecasts, fuel consumption and time constraints, and makes sure that the airplane reaches its final destination. A route, in its simplest form, is a set of ordered coordinates, *waypoints*. In an autopilot of a commercial airplane or in the computer of an *Unmanned Aerial Vehicle* (UAV), waypoints are used to generate reference trajectories which the controllers then use to navigate between the waypoints. The most primitive *reference generator* does not take into account limitations and the dynamics of the airplane. It gives a reference which is simply a sequence of line segments connecting the waypoints. The airplane will not be able to follow this reference very well and it is obvious that fuel could have been saved and the comfort of the passengers could have been improved if instead a smooth trajectory would have been generated. However, any smooth trajectory does not suffice. The airplane may *e.g.,* be too large to follow the turns which may cause a not so smooth behavior after all. Therefore, a better approach would be to include a model of the airplane in the reference generator and do a *model-based reference generation*.

Model-based reference generation is a particular type of *trajectory generation* and of interest for *e.g.,* industrial robotics and planning for unmanned vehicles. Trajectory generation is further discussed in Paper D in Part II.

Since mathematical models are used and of importance in so many different fields, there are of course a huge variety of different types of models and modeling techniques. There are also several fields studying the act of modeling, each with its own nomenclature. In *system identification e.g.,* the act of modeling is referred to as *identification* and in the closely related field of *machine learning*, the

term *learning* or *inference* is used. Since this is a thesis in system identification, we will most of the time stick to the nomenclature used there.

Mathematical modeling can be divided into two categories. Modeling either belongs to *regression* or *classification*. In this thesis we are only concerned with regression. There is further a focus on different types of regularizations. This is also reflected in the name of the thesis.

1.2 Regularization

Regularization is a methodology for making an *ill-posed* problem *well-posed* (Poggio et al., 1985; Neumaier, 1998). A problem is ill-posed (Hadamard (1902), see also Tikhonov and Arsenin (1977, p. 7)) if its solution

- does not exist,
- is not unique or
- does not depend continuously on the input data.

If a problem is not ill-posed, it is well-posed. An example of an ill-posed problem could be the task of finding *the* $x \in \mathcal{R}^{n_x}$, given $y \in \mathcal{R}^{n_y}$ and $A \in \mathcal{R}^{n_y \times n_x}$, that solves

$$\min_x \|y - Ax\|_2^2. \tag{1.1}$$

If $rank(A) < n_x$ the minimizing x is not unique and the problem hence ill-posed. A well-posed regularized version of the problem is given by the regularized least squares problem

$$\min_x \|y - Ax\|_2^2 + \lambda \|x\|_2^2, \quad \lambda \in \mathcal{R}^+. \tag{1.2}$$

The added term $\lambda\|x\|_2^2$ conveys the desire that $\|x\|_2^2$ should be small. It also makes the solution unique and the problem well-posed. Regularization can also be used to communicate other prior thoughts concerning a parameter, signal or model. Common properties imposed by regularization are smoothness or sparseness, as we will see later. We will return to the regularized least squares problem in later chapters and leave the details for then.

Regularization is also a way to control for *overfitting*. Overfitting is a problem that can occur in the estimation process of a model and in particular when a stochastic noise process is modeled as a deterministic signal. The most common way to avoid overfitting is to limit the model's ability to pick up rapid variations in the data, often associated with the noise. One technique for doing this is regularization. By controlling for overfitting a *bias* is usually introduced. The *variance* is however decreased. Regularization is therefore also a way to deal with the *bias-variance trade-off* for a model.

In statistics and machine learning, regularization has gained popularity due to modeling methods such as *Support Vector Machines* (SVM, Vapnik (1979, 1995)), *ridge regression* (Hoerl and Kennard (1970), see also Hastie et al. (2001), p. 59)

and *lasso* (least absolute shrinkage and selection operator, Tibsharani (1996), see also Hastie et al. (2001), p. 64). When the *Bayesian* approach to modeling is used, regularization often shows up and can be associated with the prior knowledge.

In system identification, the *Akaike Information Criterion* (AIC, Akaike (1973)) is a well known way to balance the model fit against the model complexity. Regularization here acts as a price on model complexity.

Regularization has also had a great impact on many applications, and very much so in clinical imaging. In *e.g.,* breast cancer imaging, the number of sensors is physically restricted which leads to long scan times. Regularization and sparsity can be used to reduce that, as shown in Guo et al. (2010) and Brady et al. (2009). In *Magnetic Resonance Imaging* (MRI), the number of scans is physically limited and to obtain high resolution images, regularization plays a key role, see *e.g.,* Brady et al. (2009).

Example 1.3: Compressed Sensing

The *Nyquist-Shannon sampling criterion* states that for a *bandlimited* (no energy above a certain frequency) signal, the sampling frequency should be twice that of the bandlimit to guarantee the possibility to perfectly reconstruct the time-continuous signal (see *e.g.,* Oppenheim et al. (1996, p. 519)). That means that to obtain a (good) audio recording a sampling frequency of at least 40 kHz is needed, since our ears are sensitive to frequencies up to 20 kHz. However, MP3 files are often around 3 megabytes, not 30 megabytes (a three minute stereo recording gives $3 \cdot 60 \cdot 2 \cdot 40 \cdot 10^3 = 14.4 \cdot 10^6$ samples. A precision of 16 bits gives 28.8 megabytes). Data compression is of course the reason for this storage saving. A sound is hence sampled, stored and then compressed. In the compression, about 90% of the storage area is returned.

It may seem meaningless to measure a lot of information if 90% will be thrown away before someone even listened to the song. Since this thesis is about regularization, you may guess that regularization can help to sample more efficiently. And yes, a regularization technique called *Compressed Sensing* (CS, Donoho (2006); Candès et al. (2006)) is what is needed.

We continue and reveal the details behind compressed sensing in Chapter 4. An interesting and well written paper on compressed sensing which inspired to above example is given by Hayes (2009).

1.3 State Estimation

Dynamic systems are characterized by that their output depends on current and past inputs. The effect that these inputs have had on the system is gathered in the *state*. The state contains valuable information for *e.g.,* controllers and for decision making. The state is however often not directly measurable. It is therefore of interest to be able to estimate the state using the available measurements. The theory for doing this is called *state estimation*.

The focus of this thesis is not state estimation. A brief description necessary to understand the paper on state estimation in Part II is therefore only provided. In particular we discuss state estimation under process noise which is often zero but occasionally non-zero, leading to so called *load disturbances*.

1.4 Notation

It is strategic, before readers detach and jump to chapters of their choice, to explain some notational choices made throughout the thesis. Lower-case letters will be used for scalars and column vectors, while upper-case letters are used to denote matrices. "$(\cdot)$" will be used to pick out elements of vectors or matrices. $x(t)$ hence denotes the tth element of the vector x. ":" will be used, as in MATLAB, to pick-out a sequence of elements of a matrix or vector. $A(1:2,:)$ hence denotes the two top rows of the matrix A. Calligraphic letters will be used for sets. Models will be denoted by $f(\varphi,\theta)$, φ being a regressor and θ the model-parameters. $f_0(\varphi)$ will be used to denote the true system that we try to imitate using a model. "$\hat{}$" denotes an estimate of some quantity. $\hat{x}$ therefore denotes an estimate of x. A subscript will be used to index time or as sample index. x_t hence denotes the variable x at time or index t. In some papers of Part II, "$(\cdot)$" is used instead of subscript. Some exceptions to these notational choices exist.

See also listed mathematical symbols and abbreviations on pages xvii and xviii.

1.5 Publications

Published work of relevance to this thesis is listed below in chronological order. Publications marked with a "$*$" are included in Part II of this thesis.

H. Ohlsson, J. Roll, T. Glad, and L. Ljung. Using manifold learning for nonlinear system identification. In *Proceedings of the 7th IFAC Symposium on Nonlinear Control Systems (NOLCOS)*, Pretoria, South Africa, August 2007.

H. Ohlsson. *Regression on manifolds with implications for system identification*. Licentiate thesis no. 1382, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, December 2008.

H. Ohlsson, J. Roll, A. Brun, H. Knutsson, M. Andersson, and L. Ljung. Direct weight optimization applied to discontinuous functions. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008a.

H. Ohlsson, J. Roll, and L. Ljung. Manifold-constrained regressors in system identification. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008b.

* H. Ohlsson, J. Rydell, A. Brun, J. Roll, M. Andersson, A. Ynnerman, and H. Knutsson. Enabling bio-feedback using real-time fMRI. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008c.

A. Eklund, H. Ohlsson, M. Andersson, J. Rydell, A. Ynnerman, and H. Knutsson. Using real-time fMRI to control a dynamical system. In *Proceedings of the 17th Meeting of the International Society for Magnetic Resonance in Medicine (ISMRM)*, Honolulu, USA, April 2009a.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, J. Schoukens, and F. Dehairs. Three ways to do temperature reconstruction based on bivalve-proxy information. In *Proceedings of the 28th Benelux Meeting on Systems and Control*, Spa, Belgium, March 2009b.

H. Ohlsson and L. Ljung. Gray-box identification for high-dimensional manifold constrained regression. In *Proceedings of the 15th IFAC Symposium on System Identification, SYSID 2009*, Saint-Malo France, July 2009.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. On climate reconstruction using bivalve shells: Three methods to interpret the chemical signature of a shell. In *Proceedings of the 7th IFAC Symposium on Modelling and Control in Biomedical Systems*, Aalborg, Denmark, August 2009a.

A. Eklund, H. Ohlsson, M. Andersson, J. Rydell, A. Ynnerman, and H. Knutsson. Using real-time fMRI to control a dynamical system by brain activity classification. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'09)*, London, UK, September 2009b.

H. Ohlsson, M. Bauwens, and L. Ljung. On manifolds, climate reconstruction and bivalve shells. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, December 2009.

F. Lindsten, J. Callmer, H. Ohlsson, D. Törnqvist, T. B. Schön, and F. Gustafsson. Geo-referencing for UAV navigation using environmental classification. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, Alaska, May 2010.

K. Nguyen, A. Eklund, H. Ohlsson, F. Hernell, P. Ljung, C. Forsell, M. Andersson, H. Knutsson, and A. Ynnerman. Concurrent volume visualization of real-time fMRI. In *Proceedings of the IEEE International Symposium on Volume Graphics 2010*, Norrköping, Sweden, May 2010.

* H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010d.

A. Eklund, M. Andersson, H. Ohlsson, A. Ynnerman, and H. Knutsson. A brain computer interface for communication using real-time fMRI. In *Proceedings of the International Conference on Pattern Recognition 2010*, Istanbul, Turkey, August 2010.

H. Ohlsson and L. Ljung. Semi-supervised regression and system identification. In X. Hu, U. Jonsson, B. Wahlberg, and B. Ghosh, editors, *Three Decades of Progress in Control Sciences*. Springer Verlag, December 2010a. To appear.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State smoothing by sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

* H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010b. To appear.

T. Chen, T. B. Schön, H. Ohlsson, and L. Ljung. Decentralization of particle filters using arbitrary state partitioning. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. On climate reconstruction using bivalves: Three methods to interpret the chemical signature of a shell. *Computer Methods and Programs in Biomedicine*, 2010a. Accepted for publication.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. A nonlinear multi-proxy model based on manifold learning to reconstruct water temperature from high resolution trace element profiles in biogenic carbonates. *Geoscientific Model Development*, 2010b. Accepted for publication.

T. Chen, T. B. Schön, H. Ohlsson, and L. Ljung. Decentralized particle filter with arbitrary state partitioning. *IEEE Transactions on Signal Processing*, 2010b. Accepted for publication.

* H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State estimation under abrupt changes using sum-of-norms regularization. *Automatica*, 2010c. Submitted, under revision.

* H. Ohlsson and L. Ljung. Weight determination by manifold regularization. In *Distributed Decision-Making and Control*, Lecture Notes in Control and Information Sciences. Springer Verlag, 2010b. Submitted.

* H. Ohlsson and L. Ljung. Piecewise affine system identification using sum-of-norms regularization. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

* T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes – Revisited. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

  T. Falck, H. Ohlsson, L. Ljung, J. A.K. Suykens, and B. De Moor. Segmentation of times series from nonlinear dynamical systems. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

  M. P. Deisenroth and H. Ohlsson. General perspective to Gaussian filtering and smoothing: Explaining current and deriving new algorithms. In *Proceedings of the American Control Conference (ACC), 2011*, San Francisco, USA, 2011. Submitted.

1.6 Contributions

Sparseness has had a huge impact on neighboring scientific disciplines, such as machine learning and signal processing, but has had very little effect on system identification. One of the major contributions of this thesis is therefore the new developments in system identification using sparsity. Relevant readings are Papers A and B in Part II of this thesis. See also related contributions in signal processing, Papers C and D.

Manifold learning, unsupervised learning and semi-supervised learning are well establish areas in machine learning. In system identification, these subjects have hardly been given any consideration at all. A contribution of this thesis is therefore the increased understanding for these subjects and how they can be of use in system identification. Relevant reading is Paper E in Part II of this thesis.

The author of this thesis has also carried out research in *functional Magnetic Resonance Imaging* (fMRI). This contribution is described in Paper G in Part II of this thesis.

1.7 Thesis Outline

The thesis is divided into two parts. The first part contains motivations and background theory and the second part a collection of papers.

1.7.1 Outline of Part I

Chapter 2 serves as an introduction to mathematical modeling and regression and introduces the fundamental knowledge and the necessary notation for the subsequent chapters. Readers familiar with the subject can skip this chapter. Chapter 3 gives a brief introduction to state estimation. Chapter 4 discusses regularization for sparseness and Chapter 5 discusses regularization for smoothness. The last chapter of Part I gives a conclusion and discusses interesting future research directions.

1.7.2 Outline of Part II

Part II presents a collection of papers that is relevant for the thesis.

The four first papers further develop the theory presented in Chapters 3 and 4. **Paper A**,

> H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010d.

discusses what sparseness and segmented ARX models have in common. A new approach using regularization to estimate segmented ARX models is presented. The author of this thesis was the major contributor in writing this paper and in the research prior the paper. The author of this thesis also came up with the idea of using regularization for sparseness in the estimation of segmented ARX models. This paper inspired to several other applications of regularization for sparseness, see *e.g.,* Ohlsson et al. (2010a,b,c); Ohlsson and Ljung (2011); Falck et al. (2011). This work also initialized collaboration with Professor Stephen Boyd at Stanford University.

Paper B,

> H. Ohlsson and L. Ljung. Piecewise affine system identification using sum-of-norms regularization. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

extends the theory presented in Paper A to piecewise affine systems. A regularization approach is again taken. The author of this thesis was the major contributor in writing the paper and in the research prior the paper. The author of this thesis also came up with the idea of using regularization for sparseness in piecewise affine system identification.

Paper C,

> H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State estimation under abrupt changes using sum-of-norms regularization. *Automatica*, 2010c. Submitted, under revision.

discusses how sparseness can help in state estimation when abrupt changes are present, *e.g.,* load disturbances. The author of this thesis was the major contributor in writing the paper and in the research prior the paper. It was Professor

Lennart Ljung's idea to use regularization for sparseness together with state estimation. Parts of the theory presented in this paper have also been presented in Ohlsson et al. (2010a).

Paper D,

> H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010b. To appear.

presents a model-based trajectory generation scheme. Sparsity and regularization are here used to give a compact representation for the trajectory, something that is desired when communication and storage are limited. The author of this thesis was the major contributor in writing the paper and in the research prior the paper. It was Professor Fredrik Gustafsson's idea to use regularization for sparseness for trajectory generation.

The fifth paper, **Paper E**,

> H. Ohlsson and L. Ljung. Weight determination by manifold regularization. In *Distributed Decision-Making and Control*, Lecture Notes in Control and Information Sciences. Springer Verlag, 2010b. Submitted.

discusses a novel regression method *Weight Determination by Manifold Regularization* (WDMR). The regression method has strong bounds with manifold learning and has inherited properties thereof. Unlike most methods in system identification, WDMR is a semi-supervised regression method. WDMR uses regularization to control for smoothness and is therefore related to theory developed in Chapter 5. The author of this thesis was the major contributor in writing the paper and in the research prior the paper. A pre-study was presented in Ohlsson et al. (2007). WDMR, in its present formulation, was first presented in Ohlsson et al. (2008b). A number of interesting applications and extensions of WDMR have also been presented, *e.g.,* Ohlsson (2008); Ohlsson and Ljung (2009). The application to temperature reconstruction from bivalves is probably the most exciting, see *e.g.,* Ohlsson et al. (2009); Bauwens et al. (2009a, 2010a,b). The work behind WDMR has led an extensive collaboration with researchers at Vrije Universiteit Brussel. The author of this thesis came up with the idea behind WDMR.

Paper F,

> T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes – Revisited. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

continues the discussion of regularization for smoothness and examines how regularization can be used in linear system identification. The theory presented in Paper F is also related to theory developed in Chapter 5. The author of this thesis

was an active contributor in the work prior writing the paper and in writing the paper.

Paper G,

> H. Ohlsson, J. Rydell, A. Brun, J. Roll, M. Andersson, A. Ynnerman, and H. Knutsson. Enabling bio-feedback using real-time fMRI. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008c.

presents a real-time fMRI bio-feedback setup. fMRI is a method for measuring brain activity. The conventional use of fMRI is in "batch-mode". The subject is first scanned for 30 minutes. Then the data is analyzed and brain activity detected and located using smoothing on the batch of fMRI measurements. The setup presented here hence presents a real-time fMRI setup *i.e.,* fMRI measurements are analyzed as they are acquired. The setup presented led the way for several interesting real-time fMRI studies *e.g.,* Eklund et al. (2009a,b, 2010); Nguyen et al. (2010) and shows some more applied research conducted by the author of this thesis. The author of this thesis was the main contributor to the presented setup.

2

Mathematical Modeling and Regression

Models summarize available knowledge about the system. Available knowledge can be physical first principles describing the behavior of the system or it can be measurements of system specific quantities.

2.1 Types of Models and Modeling

When only physical first principles are used, modeling, or the act of finding a model, is referred to as *white-box modeling*. When modeling is solely based on measurements it is referred to as *black-box modeling* and when physical principles are combined with measurements, *gray-box modeling*.

A model (and also a system) is either *dynamic* or *static*. The output of a dynamic model depends on previous and current system inputs, while a static model only depends on the system input at the moment. One may say that a static model is memoryless, while a dynamic model contains a memory in which past inputs are stored. The words "dynamical" and "dynamic" are used interchangeably in the literature.

A *model* is made up of a *model structure* and a set of *model parameters*. Model parameters are quantities that are chosen to make the model imitate the specific system under consideration. For example, a mass-spring system can readily be modeled by a second order differential equation

$$\frac{d^2x_t}{dt^2} + a\frac{dx_t}{dt} + bx_t = c \tag{2.1}$$

in the position x of the mass. To make the model imitate a specific mass-spring system, the model parameters a, b and c have to be set. This could *e.g.,* be done

by comparing predicted mass positions of the model with observed positions. A second order differential equation is the model structure in this case and the coefficients a, b and c, the model parameters. For the second order differential equation model, the number of parameters is fixed and equal to three. That the number of model parameters is fixed characterizes a *parametric model*. The number of parameters of a *non-parametric model* typically grows with the number of observations available for estimating the model. It may seem a bit counter intuitive that a non-parametric model has parameters and often considerably more parameters than a parametric model, but that is the convention.

The quantity of interest can either belong to a set of a finite number of elements, and is then said to be *qualitative*. When the quantities are qualitative they are often denoted *labels* and the act of modeling, *classification*. Or, if on the other hand, the quantity of interest can take any value in *e.g.,* an interval, the act of modeling is referred to as *regression*. The considered quantities are then said to be *quantitative*. This thesis only treats quantitative quantities and the regression problem.

It is also common to separate a *Bayesian* approach to modeling from a non-Bayesian approach, sometimes called a *frequentist's* or a *classical* approach. Sections 2.4 and 2.5 take a non-Bayesian approach and Section 2.9 discusses a Bayesian approach to modeling.

2.2 The Regression Problem

Many problems in estimation and identification can be formulated as regression problems. In a regression problem we are seeking to determine the relationship between a *regression vector* φ (input, independent variable) and a quantity of interest, a quantitative variable y (output, dependent variable), here called the *output*. Basically this means that we would like to find the function f_0 that describes the relationship

$$y = f_0(\varphi). \tag{2.2}$$

With $\varphi \in \mathcal{R}^{n_\varphi}$ and $y \in \mathcal{R}$, f_0 is a mapping from $\mathcal{R}^{n_\varphi} \to \mathcal{R}$. For simplicity, $y \in \mathcal{R}$ will be assumed throughout the rest of this chapter.

Measuring always introduces some uncertainty, which motives the introduction of a discrepancy or noise term e,

$$y = f_0(\varphi) + e. \tag{2.3}$$

This implies that there is no longer a unique y corresponding to a φ. We will assume that the noise sequence $\{e\}$ obtained as f_0 is measured multiple times is constructed from *independent and identically distributed* (i.i.d.) zero mean stochastic variables. Let further p_e be the probability distribution associated with the random variable e.

In practice our estimate of $f_0(\varphi)$ has to be computed from a limited number of

observations of (2.3). The problem is hence to observe a number of connected pairs $\{\varphi, y\}$, and then based on this information be able to provide a guess or estimate for f_0 that is related to any given, new, value of φ.

The estimate of f_0, or the model, that we choose to work with can either be *linear* or *nonlinear*. For a linear model, the model output is a linear function of the regressors while for a nonlinear model, the model output is allowed to be a nonlinear function of the regressors.

2.3 Estimation, Validation and Test Data

Given a set of observations, $\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\mathsf{o}}$, $\mathcal{N}_\mathsf{o} \subset \mathcal{Z}$, it is often a good idea to separate the observation data set into three sets:

- The *estimation data* set is used to compute the model, *e.g.*, to compute the model parameters in a parametric model. The estimation data set will be denoted by $\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\mathsf{e}}$, $\mathcal{N}_\mathsf{e} \subseteq \mathcal{N}_\mathsf{o}$. Let also $N_\mathsf{e} \triangleq card(\mathcal{N}_\mathsf{e})$.
- The *validation data* set is used to examine an estimated model's ability to predict the output of a new set of regressor data. Having a number of prospective models of different structures, the validation data can be utilized to choose the best performing model structure. For example the number of delayed system inputs and outputs used in the regressors in a parametric model could be chosen using the validation data. The validation data set will be denoted by $\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\mathsf{v}}$, $\mathcal{N}_\mathsf{v} \subseteq \mathcal{N}_\mathsf{o}$, $\mathcal{N}_\mathsf{v} \bigcap \mathcal{N}_\mathsf{e} = \emptyset$. Let also $N_\mathsf{v} \triangleq card(\mathcal{N}_\mathsf{v})$. How the validation data is used is discussed in Section 2.5.
- The *test data* set is used to test the ability of the chosen model (with the parameter choice from the estimation step and the structure choice from the validation step) to predict new outputs. The test data set can be used to gain confidence for the chosen model. The test data set will be denoted by $\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\mathsf{t}}$, $\mathcal{N}_\mathsf{t} \subseteq \mathcal{N}_\mathsf{o}$, $\mathcal{N}_\mathsf{t} \bigcap \mathcal{N}_\mathsf{e} = \emptyset$, $\mathcal{N}_\mathsf{t} \bigcap \mathcal{N}_\mathsf{v} = \emptyset$. Let also $N_\mathsf{t} \triangleq card(\mathcal{N}_\mathsf{t})$.

2.4 Fitting a Model

Having divided the observations into an estimation, validation and test data set, we are ready to estimate a model. The conventional approach within system identification is to make use of a parametric model $f(\varphi_t, \theta)$, which is hopefully flexible enough to imitate the transformation f_0 in (2.3). Here θ is used to denote the model parameters. Examples of structures that will be used in this thesis are:

- The *Auto-Regressive with eXogenous variables* (ARX) model structure. This structure leads to a linear model. If we consider a single-input single-output dynamic system with the input u_t and the output y_t, the ARX model takes the form

$$f(\varphi_t, \theta) = \varphi_t^T \theta, \quad \varphi_t = \begin{bmatrix} -y_{t-1} & \dots & -y_{t-na} & u_{t-1} & \dots & u_{t-nb} \end{bmatrix}^T. \tag{2.4}$$

The quantities na and nb are parameters of the model structure.

- The *Finite Impulse Response* (FIR) model structure. This structure also leads to a linear model. If we again let u_t be an input of a single-input single-output dynamic system, the FIR model takes the form

$$f(\varphi_t, \theta) = \varphi_t^T \theta, \quad \varphi_t = \begin{bmatrix} u_{t-1} & \dots & u_{t-nb} \end{bmatrix}^T. \tag{2.5}$$

nb is the *order* of the FIR model.

For more on the model structures briefly introduced above, and several other model structures used in system identification, see *e.g.,* Ljung (1999, Chap. 4).

$f(\varphi_t, \theta)$ is adjusted to the regressor-output pairs of the estimation data set $\{(\varphi_t, y_t)\}_{t \in \mathcal{N}_e}$ by choosing θ as

$$\hat{\theta} = \arg\min_{\theta} \sum_{t \in \mathcal{N}_e} l\Big(y_t - f(\varphi_t, \theta)\Big), \tag{2.6}$$

where $l : \mathcal{R} \to \mathcal{R}$ is a function of the *prediction error* $y_t - f(\varphi_t, \theta)$ and typically chosen as a norm. In system identification, the use of (2.6) to estimate a model parameter is a special case of the *Prediction Error Method* (PEM, see *e.g.,* Ljung (1999, 2002)). Also, if we set l as the negative logarithm of the measurement noise distribution, *i.e.,* $l(\cdot) = -\log p_e(\cdot)$, then $\hat{\theta}$ of (2.6) equals the *Maximum Likelihood Estimate* (MLE) of θ (see *e.g.,* Ljung (2002)).

With measurement noise present, obtaining a perfect fit *i.e.,*

$$\sum_{t \in \mathcal{N}_e} l\Big(y_t - f(\varphi_t, \hat{\theta})\Big) = 0, \tag{2.7}$$

is not desirable and an extreme case of *overfitting*. Overfitting is a problem that can occur when fitting a model and means that the model has been adjusted to the particular measurement noise realization. Overfitting is primarily a problem for flexible models and to chose a model structure just flexible enough to imitate f_0 (and not flexible enough to be able to imitate the noise) would be ideal.

There are a number of approaches to find what is "just flexible enough". Most approaches can be seen belonging to either *cross validation* or *regularization*.

2.5 Cross Validation

In *Cross Validation* (CV) the validation data set $\{(\varphi_t, y_t)\}_{t \in \mathcal{N}_v}$ is utilized to find what is "just flexible enough". Since the measurement noise e of the validation data set is impossible to predict, the best possible would be to perfectly predict the outcome of the deterministic part of (2.3) *i.e.,* $f_0(\varphi)$. Therefore, for a number of candidate models $f_i(\varphi, \hat{\theta}_i)$, $i = 1, \dots, m$ ($\hat{\theta}$ found using (2.6)), a model is chosen

by

$$\underset{f_i(\varphi,\hat{\theta}_i),i=1,\ldots,m}{\arg\min} \sum_{t\in\mathcal{N}_{\mathrm{v}}} l\Big(y_t - f_i(\varphi_t,\hat{\theta}_i)\Big). \tag{2.8}$$

This type of cross-validation is the most common in system identification. There are however several other types of cross validation, see *e.g.,* (Hastie et al., 2001, pp. 214-217).

To evaluate (2.8) we need to evaluate $f(\varphi,\theta)$ at the regressors of the validation data set. To compute predictions for f_0 at regressors not included in the estimation data set is called *generalization* (Bishop, 2006, p. 2). For most practical purposes it is not enough to find a model $f(\varphi,\hat{\theta})$ that well imitates f_0 at the estimation data set, generalization is therefore an important property of a model. This is sometimes referred to as the model's ability to *generalize* to unseen data.

2.6 Regularization

Regularization is in general a methodology for making an *ill-posed* problem *well-posed*, but regularization can also be used to control for overfit. We care for both these applications in this thesis. We however choose to focus on the type of regularization (referred to as a *standard regularization method* in Poggio et al. (1985)) obtained by adding a penalty term J to the criterion of fit. The penalty J should be regarded as a means to introduce *a priori* knowledge.

In particular, given a number of candidate models $f_i(\varphi,\hat{\theta}_i)$, $i = 1,\ldots,m$ ($\hat{\theta}$ found using (2.6)), we can use regularization to select a model "just flexible enough" by considering a criterion

$$\underset{f_i(\varphi,\hat{\theta}_i),i=1,\ldots,m}{\arg\min} \sum_{t\in\mathcal{N}_{\mathrm{e}}} l\Big(y_t - f_i(\varphi_t,\hat{\theta}_i)\Big) + J(f_i). \tag{2.9}$$

J should then be a flexibility penalty conveying the message that we wish an as "simple" model as possible that fits the estimation data. Notice that to choose a model using (2.9) only requires the estimation data set while cross-validation requires both an estimation and a validation data set. Regularization may therefore be a good choice when the number of observation data is limited.

The *Akaike Information Criterion* (AIC, Akaike (1973)),

$$\underset{f_i(\varphi,\hat{\theta}_i),i=1,\ldots,m}{\arg\min} \; -2\sum_{t\in\mathcal{N}_{\mathrm{e}}} \log p_e\Big(y_t - f_i(\varphi_t,\hat{\theta}_i)\Big) + 2dim(\hat{\theta}_i), \tag{2.10}$$

with $\hat{\theta}_i$ found using $l(\cdot) = -\log p_e(\cdot)$ in (2.6) (MLE of θ), is an example of this type of usage of regularization.

Example 2.1: ARX and Model Selection

Consider a single-input single-output dynamic system with an input u_t and an output y_t. Let the candidate models be ARX models with different nb's (see (2.4) for ARX and nb). Let *e.g.,*

$$f_1(\varphi_t, \theta_1) = \varphi_t^T \theta_1, \quad na = 1,\ nb = 1, \tag{2.11a}$$

$$f_2(\varphi_t, \theta_2) = \varphi_t^T \theta_2, \quad na = 1,\ nb = 2, \tag{2.11b}$$

$$\vdots$$

$$f_m(\varphi_t, \theta_m) = \varphi_t^T \theta_m, \quad na = 1,\ nb = m, \tag{2.11c}$$

and compute θ_1, $\theta_2, \ldots, \theta_m$ using (2.6). The flexibility of an ARX model grows with nb, a suitable choice of penalty J in (2.9) could therefore be

$$J(f_i) = nb \text{ for } f_i \tag{2.12}$$

if an as "simple" model as possible but with a reasonable good fit is sought.

Regularization can also be used to control the regressor parameter value of a single model. $f(\varphi_t, \theta)$ is then adjusted to the observations by choosing θ as

$$\hat{\theta} = \arg\min_{\theta} \sum_{t \in \mathcal{N}_e} l\Big(y_t - f(\varphi_t, \theta)\Big) + \lambda J(\theta, \varphi_t), \tag{2.13}$$

rather than using (2.6). $J(\theta, \varphi_t)$ again serves as a cost on flexibility and is often used to penalize non-smooth estimates (this is discussed in Chapter 5). However, $J(\theta, \varphi_t)$ could also be used to express the prior knowledge of a sparse parameter vector θ (this is discussed in Chapter 4). $\lambda \in \mathcal{R}^+$ is seen as a design parameter and regulates the trade-off between fit to the estimation data and flexibility. Choosing the "just flexible enough" model structure is now a matter of choosing the right λ-value. λ is denoted the *regularization parameter* or *regularization constant* and $\hat{\theta}$ as a function of regularization parameter, the *regularization path*.

An expression of the form (2.13) is of great importance for this thesis and will be a key ingredient in the theory developed in Chapters 4 and 5 and in several of the papers of Part II. (2.13) is a type of *shrinkage method* as it is often used to shrink regression parameters toward zero (Hastie et al., 2001, p. 59).

Example 2.2: ARX and ℓ_2-Regularization

Consider again a single-input single-output dynamic system with an input u_t and an output y_t. Let us use an ARX model (2.4) and fix na and nb.

Let $l(\cdot) = (\cdot)^2$ in (2.6). For this particular choice, (2.6) is referred to as the *Least Squares* (LS) problem. Let $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$ be a given estimation data set. If we now define

$$y \triangleq \begin{bmatrix} y_1 & \ldots & y_{N_e} \end{bmatrix}^T, \qquad \Phi \triangleq \begin{bmatrix} \varphi_1 & \ldots & \varphi_{N_e} \end{bmatrix}^T, \tag{2.14}$$

(2.6) can be written as

$$\hat{\theta} = \arg\min_{\theta} \|y - \Phi\theta\|_2^2 = \arg\min_{\theta} (y - \Phi\theta)^T (y - \Phi\theta). \tag{2.15}$$

We can characterize the solution of (2.15) by determining if

$$y = \Phi\theta \tag{2.16}$$

is *overdetermined*, *underdetermined* or has a unique solution. It is useful to separate between the three cases:

(2.16) **is overdetermined.** In this case there are more observations than model parameters. This is the most studied case in system identification. If Φ has full column rank *i.e.,*

$$rank(\Phi) = dim(\theta), \tag{2.17}$$

then $\hat{\theta}$ in (2.15) can be computed explicitly to

$$\hat{\theta} = (\Phi^T\Phi)^{-1}\Phi^T y. \tag{2.18}$$

$(\Phi^T\Phi)^{-1}\Phi^T$ is known as the *Moore-Penrose pseudoinverse* and generally denoted by $\Phi^\dagger$. Geometrically, $\Phi\theta$ is a linear combination of the columns of Φ. $f(\varphi, \theta)$ is hence restricted to the plane spanned by the columns of Φ. (2.15) can then be interpreted as the problem of finding the vector in the plane spanned by the columns of Φ that is the closest, in an Euclidean sense, to the vector y. The orthogonal projection of y onto the plane spanned by the columns of Φ,

$$\Phi(\Phi^T\Phi)^{-1}\Phi^T y, \tag{2.19}$$

is well known to minimize this distance. (2.18) should therefore be seen as a projection onto the plane spanned by the columns of Φ. When Φ has full rank, (2.15) has a unique solution. If Φ does not have full rank, there exits a lower number columns ($< dim(\theta)$) that span the plane. $\hat{\theta}$ is therefore no longer unique.

The ARX model $f(\varphi_t, \hat{\theta})$ does not, in general, perfectly predict the outputs in the estimation data set, but since measurement noise is present, this is preferred over an overfit.

(2.16) **has a unique solution.** Assume Φ is quadratic and has full rank, $\mathcal{R}^{N_e}$ is then spanned by the columns of Φ which also make up a basis for $\mathcal{R}^{N_e}$. The task is now to express y in this basis. We hence want to solve the equation system

$$y = \Phi\theta. \tag{2.20}$$

(2.20) is solved by

$$\hat{\theta} = \Phi^{-1} y. \tag{2.21}$$

The inverse exists since Φ is quadratic and has full rank. For $\hat{\theta} = \Phi^{-1} y$ a

perfect fit is obtained, *i.e.,*

$$\|y - \Phi\hat{\theta}\|_2^2 = 0. \tag{2.22}$$

It is worth notice that the Moore-Penrose pseudoinverse in this case reduces to the ordinary inverse since

$$\Phi^\dagger = (\Phi^T\Phi)^{-1}\Phi^T = \Phi^{-1}\Phi^{-T}\Phi^T = \Phi^{-1}. \tag{2.23}$$

(2.18) hence still holds.

(2.16) **is underdetermined.** In this case, the columns of Φ construct an over complete basis for $\mathcal{R}^{N_e}$. There is hence an infinite number of θs that obtain a perfect fit *i.e.,*

$$\|y - \Phi\theta\|_2^2 = 0. \tag{2.24}$$

(2.15) is hence ill-posed. Regularization can here be used to express which one of these infinite solutions that is desired.

The Moore-Penrose pseudoinverse is for this case not well defined, since $\Phi^T\Phi$ is singular.

Remark 2.1. If (2.16) is either overdetermined or has a unique solution, (2.15) is a *strictly convex* optimization problem and has therefore a unique solution (see *e.g.,* Bertsekas et al. (2003, Prop. 2.1.2)). If (2.16) is underdetermined, (2.15) is *convex* and any minimizing $\hat{\theta}$ is therefore a *global minimum* (see *e.g.,* Boyd and Vandenberghe (2004, p. 138)). $\hat{\theta}$ may however not be unique in this case.

Let us assume that we have insight that tells us that θ should be "small". We could then use regularization to reduce the flexibility of $f(\varphi, \theta) = \varphi^T\theta$ and to only allow models with a small θ. That would *e.g.,* help us find a unique model if (2.16) is underdetermined. However, it could also be used to reduce the flexibility of a model to control for overfit and find a "just flexible enough" model ((2.16) does not need to be underdetermined to use regularization for this purpose). Let us say that we would be satisfied if $\|\theta\|_2^2$ is kept small. Using regularization we can express this prior knowledge/insight as

$$\hat{\theta} = \arg\min_\theta \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_2^2, \quad \lambda \in \mathcal{R}^+. \tag{2.25}$$

(2.25) is an ℓ_2*-regularized least squares problem*, often referred to as *ridge regression* or *Tikhonov regularization* (Hoerl and Kennard (1970), see also Hastie et al. (2001), p. 59). Since the objective function is quadratic in θ, an explicit expression for $\hat{\theta}$ can be computed. The gradient with respect to θ of the objective function of (2.25) becomes

$$\nabla_\theta \left(\|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_2^2\right) = -2\Phi^T(y - \Phi\theta) + 2\lambda\theta. \tag{2.26}$$

Setting the gradient equal to zero and solve gives

$$\hat{\theta} = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T y. \tag{2.27}$$

(2.27) and (2.18) take a very similar form. And in fact, adding a small diagonal

matrix λI to $\Phi^T\Phi$ to make the Moore-Penrose pseudoinverse well defined was the main motivation for ridge regression when it was introduced by Hoerl and Kennard (1970).

2.7 Bias-Variance Tradeoff

Let us assume that an estimate of f_0 at the regressor φ_* is desired. To find what is "just flexible enough" can then be shown to be a matter of finding a suitable tradeoff between *variance*

$$\mathsf{E}_{\hat{\theta}}\left[\left(\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]-f(\varphi_*,\hat{\theta})\right)^2\right] \tag{2.28}$$

and *bias*

$$f_0(\varphi_*)-\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]. \tag{2.29}$$

This can be understood as follows. Given an estimation data set, we estimate θ. Since the y-measurements in the estimation data set are noisy, they are inherently stochastic and so will also $\hat{\theta}$ be. It therefore makes sense to study the quantity

$$\mathsf{E}_{\hat{\theta}}\left[\left(f_0(\varphi_*)-f(\varphi_*,\hat{\theta})\right)^2\right] \tag{2.30}$$

as a measure of performance (for estimating f_0 at φ_*). The expectation is here taken with respect to $\hat{\theta}$. This quantity is called the *Mean Squared Error* (MSE). To minimize the MSE would be ideal and was earlier referred to as finding a model "just flexible enough". To see how the bias and variance relate to MSE, add and subtract $\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]$ in (2.30). We get

$$\begin{aligned}
\mathsf{E}_{\hat{\theta}}\left[\left(f_0(\varphi_*)-f(\varphi_*,\hat{\theta})\right)^2\right] &= \mathsf{E}_{\hat{\theta}}\left[\left(f_0(\varphi_*)-\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]+\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]-f(\varphi_*,\hat{\theta})\right)^2\right] \\
&= \mathsf{E}_{\hat{\theta}}\Big[\left(f_0(\varphi_*)-\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]\right)^2+\left(\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]-f(\varphi_*,\hat{\theta})\right)^2 \\
&\quad +2\left(f_0(\varphi_*)-\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]\right)\left(\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]-f(\varphi_*,\hat{\theta})\right)\Big] \\
&= \left(f_0(\varphi_*)-\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]\right)^2+\mathsf{E}_{\hat{\theta}}\left[\left(\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]-f(\varphi_*,\hat{\theta})\right)^2\right].
\end{aligned}$$

The first term

$$\left(f_0(\varphi_*)-\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]\right)^2 \tag{2.31}$$

is the squared bias and the second term

$$\mathsf{E}_{\hat{\theta}}\left[\left(\mathsf{E}_{\hat{\theta}}[f(\varphi_*,\hat{\theta})]-f(\varphi_*,\hat{\theta})\right)^2\right] \tag{2.32}$$

is the variance. The bias is due to limitations in our model structure and the variance term is due to the stochastic nature of our estimation data set (the measurement noise). However, both the bias and the variance also depend on the cost function used to find $\hat{\theta}$.

Flexible models generally give high variance, but low bias, whereas non-flexible models give low variance, but high bias.

Example 2.3: Regularization and the Bias-Variance Tradeoff

Consider the single-input single-output system ($\delta(\cdot)$ the Dirac delta function)

$$y_t = \sum_{k=1}^{n} g_k^0 u_{t-k} + e_t, \quad \mathsf{E}[e_t] = 0,\ \mathsf{E}[e_t e_s] = \delta(t-s)\sigma^2,\ \forall t, s \in \mathcal{N}. \tag{2.33}$$

The sequence $\{g_k^0\}_{k=1}^n$ is the *impulse response* of the system *i.e.,* the response to an impulse ($u_t = \delta(t)$ in (2.33) gives $y_t = g_t^0 + e_t$, $t = 1, \dots, n$, $y_t = e_t$, $t = n+1, n+2, \dots$). Let us estimate the impulse response. Assume that we use an nth order FIR model (see (2.5))

$$f(\varphi_t, \theta) = \varphi_t^T \theta, \quad \varphi_t = \begin{bmatrix} u(t-1) & \dots & u(t-n) \end{bmatrix}^T,\ \theta \in \mathcal{R}^n. \tag{2.34}$$

Let $\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}$ be the estimation data set and define

$$\begin{aligned} y &\triangleq \begin{bmatrix} y_1 & \dots & y_{N_\mathrm{e}} \end{bmatrix}^T, \quad \Phi \triangleq \begin{bmatrix} \varphi_1 & \dots & \varphi_{N_\mathrm{e}} \end{bmatrix}^T, \\ \Lambda &\triangleq \begin{bmatrix} e_1 & \dots & e_{N_\mathrm{e}} \end{bmatrix}^T, \quad \theta_0 \triangleq \begin{bmatrix} g_1^0 & \dots & g_n^0 \end{bmatrix}^T. \end{aligned} \tag{2.35}$$

Consider now the ℓ_2-regularized least squares criterion

$$\hat{\theta} = \arg\min_{\theta} \|y - \Phi\theta\|_2^2 + \theta^T D \theta, \quad D \in \mathcal{R}^{n \times n},\ D \geq 0, \tag{2.36}$$

with a solution (see (2.27))

$$\hat{\theta} = (\Phi^T\Phi + D)^{-1}\Phi^T y. \tag{2.37}$$

The bias for an estimate at φ_* is then readily computed to

$$\begin{aligned} \varphi_*^T\theta_0 - \mathsf{E}_{\hat{\theta}}[\varphi_*^T\hat{\theta}] &= \varphi_*^T\theta_0 - \mathsf{E}_y[\varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T y] && \text{(2.38a)} \\ &= \varphi_*^T\theta_0 - \varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\, \mathsf{E}_\Lambda[\Phi\theta_0 + \Lambda] && \text{(2.38b)} \\ &= \varphi_*^T\theta_0 - \varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\Phi\theta_0 && \text{(2.38c)} \end{aligned}$$

and the variance to

$$\begin{aligned} \mathsf{E}_{\hat{\theta}}\left[\left(\mathsf{E}_{\hat{\theta}}[\varphi_*^T\hat{\theta}] - \varphi_*^T\hat{\theta}\right)^2\right] &= \mathsf{E}_y\left[\left(\varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\Phi\theta_0 - \varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T y\right)^2\right] \\ &= \mathsf{E}_\Lambda\left[\left(\varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\left(\Phi\theta_0 - (\Phi\theta_0 + \Lambda)\right)\right)^2\right] \\ &= \mathsf{E}_\Lambda\left[\left(\varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\Lambda\right)^2\right] \\ &= \varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\, \mathsf{E}_\Lambda[\Lambda\Lambda^T]\Phi(\Phi^T\Phi + D)^{-1}\varphi_* \\ &= \sigma^2\varphi_*^T(\Phi^T\Phi + D)^{-1}\Phi^T\Phi(\Phi^T\Phi + D)^{-1}\varphi_*. \end{aligned} \tag{2.39}$$

Let now $D = \lambda I_n$, $\lambda \geq 0$. Then, if the estimation data input u_t is chosen as zero

mean white noise with variance μ and for a large N_e, it holds that

$$\Phi^T \Phi \approx N_e \mu I_n. \tag{2.40}$$

If $\Phi^T \Phi = N_e \mu I_n$ is used in (2.38) and (2.39) the bias becomes

$$\varphi_*^T \theta_0 - \mathsf{E}_{\hat{\theta}}[\varphi_*^T \hat{\theta}] = \left(\frac{\lambda}{N_e \mu + \lambda} \right) \varphi_*^T \theta_0 \tag{2.41}$$

and the variance

$$\mathsf{E}_{\hat{\theta}} \left[\left(\mathsf{E}_{\hat{\theta}}[\varphi_*^T \hat{\theta}] - \varphi_*^T \hat{\theta} \right)^2 \right] = \sigma^2 \frac{N_e \mu}{(N_e \mu + \lambda)^2} \varphi_*^T \varphi_*. \tag{2.42}$$

Notice that when $\lambda = 0$ we obtain the unbiased least squares estimate. The variance for the least squares estimate is however larger than the variance of an estimate obtained for a small positive λ. A small positive λ causes a biased estimate though. Figure 2.1 gives a sketch of how the typical variance and bias depend on λ.

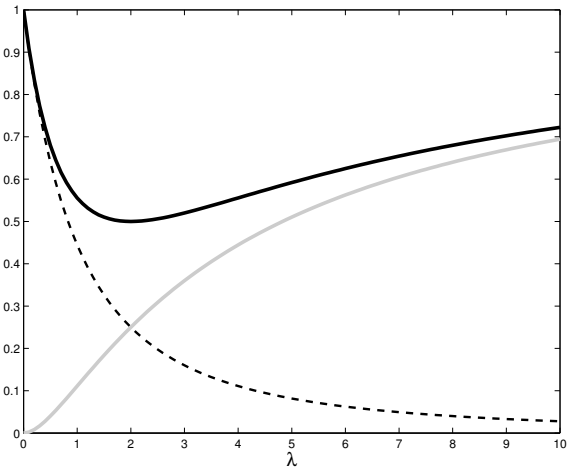

Figure 2.1: *Bias-variance visualization for regularization. The squared bias* $\left(f_0(\varphi_*) - \mathsf{E}_{\hat{\theta}}[f(\varphi_*, \hat{\theta})] \right)^2$ *is showed using the gray line, the variance* $\mathsf{E}_{\hat{\theta}} \left[\left(\mathsf{E}_{\hat{\theta}}[f(\varphi_*, \hat{\theta})] - f(\varphi_*, \hat{\theta}) \right)^2 \right]$ *using the dashed line and the MSE using the black line.*

We will return to impulse response identification in Paper F and explore more sophisticated choices of D-matrix. In fact, some of the most recent contributions in impulse response identification use ℓ_2-regularization, see *e.g.,* Pillonetto and De Nicolao (2010).

2.8 Performance Measures

To evaluate the prediction performance of different models a performance measure is needed. For a given test data set $\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\text{t}}$ and a model $f(\varphi, \hat{\theta})$, we choose to use

$$\left(1 - \sqrt{\frac{\sum_{t\in\mathcal{N}_\text{t}} \left|y_t - f(\varphi_t, \hat{\theta})\right|^2}{\sum_{t\in\mathcal{N}_\text{t}} \left|y_t - \frac{1}{N_\text{t}} \sum_{s\in\mathcal{N}_t} y_s\right|^2}}\right) \times 100 \tag{2.43}$$

as a performance measure. We will call the computed quantity *fit* and express us by saying that a prediction has a certain percentage fit to a set of data.

At some point in the thesis the *Mean Absolute Error* (MAE)

$$\frac{1}{N_\text{t}} \sum_{t\in\mathcal{N}_\text{t}} \left|y_t - f(\varphi_t, \hat{\theta})\right| \tag{2.44}$$

will also be used.

2.9 Bayesian Modeling

In *Bayesian modeling*, or *Bayesian inference*, probability distributions are used to represent stochasticity and uncertainty. For a parametric model, this implies that a distribution over parameter-values is computed rather than a single regressor parameter estimate $\hat{\theta}$. Also the predictions will be distributions over possible estimates rather than a single function-value for a given φ.

A Bayesian practitioner argues that there are two sources of information. The prior knowledge about the system and the observations. The prior knowledge or prior believes have to be formulated as a probability distribution, denoted a *prior*. The prior believes then get updated using observations to form a posterior, an updated probability distribution. How to weight together the prior and the observations is given by Bayes' theorem (Bayes, 1763):

Theorem 2.1 (Bayes' Theorem). *Let $p(\theta)$ be a prior, $p\left(\{y_t\}_{t\in\mathcal{N}_\text{e}} \middle| \theta, \{\varphi_t\}_{t\in\mathcal{N}_\text{e}}\right)$ the likelihood of observing the outputs $\{y_t\}_{t\in\mathcal{N}_\text{e}}$ given $\{\varphi_t\}_{t\in\mathcal{N}_\text{e}}$ and θ, and $p(\{y_t\}_{t\in\mathcal{N}_\text{e}}|\{\varphi_t\}_{t\in\mathcal{N}_\text{e}})$ the probability of observing the data $\{y_t\}_{t\in\mathcal{N}_\text{e}}$ given $\{\varphi_t\}_{t\in\mathcal{N}_\text{e}}$. The posterior distribution for θ given the observations is then given by*

$$p\left(\theta \middle| \{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\text{e}}\right) = \frac{p\left(\{y_t\}_{t\in\mathcal{N}_\text{e}} \middle| \theta, \{\varphi_t\}_{t\in\mathcal{N}_\text{e}}\right) p(\theta)}{p\left(\{y_t\}_{t\in\mathcal{N}_\text{e}} \middle| \{\varphi_t\}_{t\in\mathcal{N}_\text{e}}\right)}. \tag{2.45}$$

The model $f(\varphi, \theta)$ is in a Bayesian framework represented by the *predictive distribution*. Let y_* be an observation of $f_0(\varphi_*)$, $p\left(\theta \middle| \{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\text{e}}\right)$ the posterior distribution for θ given the observations (computed using Theorem 2.1) and let $p(y_*|\varphi_*, \theta)$ be the likelihood of observing the output y_* given φ_* and θ. The pre-

dictive distribution for y_* is then given by

$$p\Big(y_*\Big|\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\mathrm{e}}, \varphi_*\Big) = \int p\Big(y_*\Big|\varphi_*, \theta\Big)p\Big(\theta\Big|\{(\varphi_t, y_t)\}_{t\in\mathcal{N}_\mathrm{e}}\Big)d\theta. \tag{2.46}$$

The predictive distribution tells us how certain we are that the measured system response to φ_* takes a certain value.

It is common to let the prior $p(\theta)$ depend on a number of *hyperparameters*, let us call these θ_h. The prior hence takes the form $p(\theta|\theta_h)$. The hyperparameters are usually determined from data by maximizing the log marginal likelihood,

$$\log p\Big(\{y_t\}_{t\in\mathcal{N}_\mathrm{e}}\Big|\{\varphi_t\}_{t\in\mathcal{N}_\mathrm{e}}, \theta_h\Big) = \log \int p\Big(\{y_t\}_{t\in\mathcal{N}_\mathrm{e}}\Big|\{\varphi_t\}_{t\in\mathcal{N}_\mathrm{e}}, \theta\Big)p(\theta|\theta_h)d\theta. \tag{2.47}$$

This approach to estimating θ_h is referred to as *empirical Bayes* (see *e.g.,* Bishop (2006, p. 165)).

Example 2.4: ARX Cont'd

Consider the ARX-type of system

$$y_t = \varphi_t^T\theta + e_t, \quad e_t \sim N(0, \sigma^2), \tag{2.48}$$

with φ_t containing old system inputs and outputs. Assume that we are given the observations $\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}$, know the (i.i.d.) measurement noise variance σ^2 and that we have reason to believe that θ is small. Taking a Bayesian approach, we then compute the posterior distribution $p\Big(\theta\Big|\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}\Big)$ as (see Theorem 2.1)

$$p\Big(\theta\Big|\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}\Big) = \frac{\Pi_{t=1}^{N_\mathrm{e}} N(y_t; \varphi_t^T\theta, \sigma^2)p(\theta)}{p\Big(\{y_t\}_{t=1}^{N_\mathrm{e}}\Big|\{\varphi_t\}_{t=1}^{N_\mathrm{e}}\Big)}. \tag{2.49}$$

$p(\theta)$ is here the prior and $N(y_t; \varphi_t^T\theta, \sigma^2)$ is used to denote that $y_t \sim N(\varphi_t^T\theta, \sigma^2)$. To convey our belief of a small θ, and to get a closed-form expression for the posterior, we choose to use a Gaussian prior, say $N(0, I)$. If we first introduce

$$y \triangleq \begin{bmatrix} y_1 & \dots & y_{N_\mathrm{e}} \end{bmatrix}^T, \ \Phi \triangleq \begin{bmatrix} \varphi_1 & \dots & \varphi_{N_\mathrm{e}} \end{bmatrix}^T, \tag{2.50}$$

the posterior can be computed using standard Gaussian identities, see *e.g.,* Rasmussen and Williams (2005, p. 200), to

$$p\Big(\theta\Big|\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}\Big) = \frac{\Pi_{t=1}^{N_\mathrm{e}} N(y_t; \varphi_t^T\theta, \sigma^2)N(\theta; 0, I)}{\int \Pi_{t=1}^{N_\mathrm{e}} N(y_t; \varphi_t^T\theta, \sigma^2)N(\theta; 0, I)d\theta} \tag{2.51a}$$

$$= N(\theta; (\Phi^T\Phi + \sigma^2 I)^{-1}\Phi^T y, (\sigma^{-2}\Phi^T\Phi + I)^{-1}). \tag{2.51b}$$

The predictive distribution is now readily computed to

$$p\Big(y_*\Big|\varphi_*, \{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}\Big) = \int N(y_*; \varphi_*^T\theta, \sigma^2)p\Big(\theta\Big|\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}\Big)d\theta \tag{2.52}$$

$$= N\Big(\varphi_*^T(\Phi^T\Phi + \sigma^2 I)^{-1}\Phi^T y, \sigma^2 + \varphi_*^T(\sigma^{-2}\Phi^T\Phi + I)^{-1}\varphi_*\Big),$$

with $p\left(\theta \middle| \{(\varphi_t, y_t)\}_{t=1}^{N_e}\right)$ from (2.51).

Let us now explore what happens if we let the variance of the prior free and instead uses $N(0, \theta_h I)$, $\theta_h \in \mathcal{R}^+$, as a prior. We then see θ_h as a hyperparameter and compute it by maximizing the log marginal likelihood. Using basic Gaussian identities (see *e.g.*, Rasmussen and Williams (2005, p. 200)), (2.47) can in this particular setting be expressed as

$$\log p\left(\{y_t\}_{t=1}^{N_e} \middle| \{\varphi_t\}_{t=1}^{N_e}, \theta_h\right) = \log \int \Pi_{t=1}^{N_e} N(y_t; \varphi_t^T \theta, \sigma^2) N(\theta; 0, \theta_h I) d\theta \tag{2.53a}$$

$$= \log Z^{-1} \int N\left(\theta; \sigma^{-2} A^{-1} \Phi^T y, A^{-1}\right) d\theta \tag{2.53b}$$

$$= \log Z^{-1} \tag{2.53c}$$

with A and the normalizing constant Z defined as

$$Z^{-1} \triangleq \frac{1}{\theta_h^{dim(\theta)/2}} \frac{1}{(2\pi\sigma^2)^{N_e/2}} |A|^{-1/2} e^{-\frac{1}{2\sigma^2}\|y - \sigma^{-2}\Phi A^{-1}\Phi^T y\|_2^2 - \frac{1}{2\theta_h \sigma^4}\|A^{-1}\Phi^T y\|_2^2} \tag{2.54}$$

$$A \triangleq \theta_h^{-1} I + \sigma^{-2} \Phi^T \Phi. \tag{2.55}$$

θ_h is then chosen according to

$$\hat{\theta}_h = \arg\max_{\theta_h} \log Z^{-1}. \tag{2.56}$$

For more details see *e.g.*, Bishop (2006, pp. 152-158 and pp. 165-169).

Remark 2.2. Maximizing the posterior $p\left(\theta \middle| \{(\varphi_t, y_t)\}_{t \in \mathcal{N}_e}\right)$ with respect to θ gives the *Maximum A Posteriori* (MAP) estimate for θ. When the posterior is a Gaussian, the MAP is given by the mean of the Gaussian. In Example 2.4, using $N(0, I)$ as a prior, the MAP estimate for θ became

$$(\Phi^T \Phi + \sigma^2 I)^{-1} \Phi^T y. \tag{2.57}$$

This is the same expression as for ridge regression with $\lambda = \sigma^2$, see (2.27). In fact, most standard regularization methods can be given an interpretation as a MAP estimate.

2.10 High Dimensional Regression and Manifolds

We finish this chapter on mathematical modeling and regression by discussing high dimensional regression, manifolds and manifold learning. We will return to these subjects in Paper E.

High-dimensional regressors can lead to ill-posed regression problems. Especially if the dimension of the regressors exceeds the number of observations, special care is needed, as we saw in Example 2.2. There are a number of strategies for handling high-dimensional regression problems:

- The first strategy is *feature selection*. Feature selection is used to reduce the dimension of the high-dimensional regressors by eliminating elements

having *e.g.,* little correlation with the output. The "new" low-dimensional regressors are used instead of the original regressors in the regression algorithm. An example is *backward stepwise regression* (see *e.g.,* Daniel and Wood (1980, pp. 84-85)). Also many regression methods using regularization contain some type of feature selection. Popular regression methods here include lasso (see *e.g.,* Example 4.1) and ridge regression.

- The second strategy is *feature extraction*. Feature extraction is also used to reduce the dimension of the high-dimensional regressors. However, rather than eliminating elements, elements are combined. *Partial Least Squares* (PLS, Wold (1966)) and *Principle Component Analysis* (PCA, Pearson (1901)) are popular methods used for feature extraction. Also *manifold learning* discussed in the next section can be used for feature extraction. The regression method discussed in Paper E can also be seen using feature extraction.

Both feature selection and extraction are special cases of *dimensionality reduction* methods.

Another issue which high-dimensional regression algorithms have to deal with is the lack of data, commonly termed the *curse of dimensionality* (Bellman, 1961). For instance, imagine N samples uniformly distributed in a d-dimensional unit hypercube $[0, 1]^d$. The N samples could for example be the regressors in the set of observed data. To include 10% of the samples, we need on average to pick out a cube with the side 0.1 for $d = 1$ and a cube with the side 0.8 for $d = 10$, Figure 2.2 illustrates this. The data hence easily become sparse with increasing

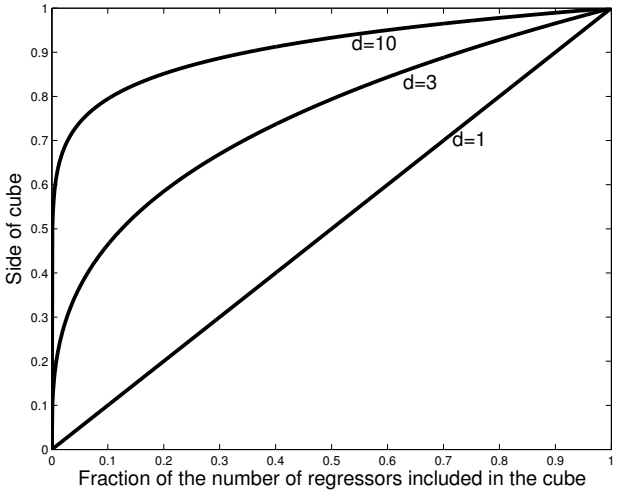

Figure 2.2: *An illustration of the curse of dimensionality. Assume that the N regressors are uniformly distributed in a d-dimensional unit cube. On average we then need to use a cube with a side of 0.1 to include $0.1N$ regressors for $d = 1$, while for $d = 10$ we will need a cube with a side of 0.8.*

dimensionality. Consequently, given a regressor, the likelihood of finding one of the estimation regressors close-by, gets smaller and smaller with increasing dimension. This means that for high-dimensional regression problems, considerably more samples are needed than for low-dimensional regression problems to make accurate predictions. This also implies that regression methods using pairwise distances between regressors, such as *nearest neighbor* (see *e.g.,* Hastie et al. (2001, p. 14)) and *support vector regression* (see Section 5.1), suffer. This follows since, as dimensionality grows the distances between regressors increase, become more similar and hence less expressive (see Figure 2.3 for an illustration and Chapelle et al. (2006) and Bengio et al. (2006) for further readings).

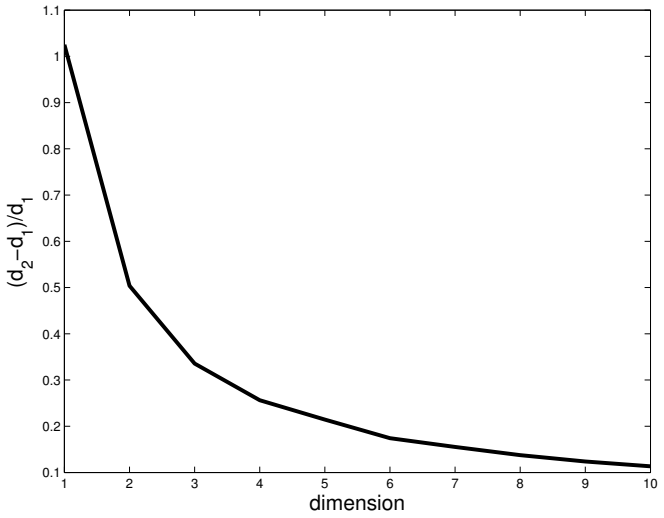

Figure 2.3: *As the dimension of the regressor space increases (keeping the number of regressors fixed) so does the distance from any regressor to all other regressors. The distance to the closest estimation regressor, d_1, of a regressor is hence increasing with dimension. The distance to the second closest estimation regressor, d_2, is also increasing. A prediction has then to be made based on more and more distant observations. In addition, the relative distance, $(d_2 - d_1)/d_1$, decreases, making the estimation data less expressive. Rephrased in a somewhat sloppy way, a given point in a high-dimensional space has many "nearest neighbors", but all far away.*

Very common, however, is that the regressors $\varphi \in \mathcal{R}^{n_\varphi}$ for various reasons are constrained to lie in a subset $\Omega \subset \mathcal{R}^{n_\varphi}$. A specific example could be a set of images of human faces. An image of a human face is a $p \times p$ matrix, each entry of the matrix giving the gray tone in a pixel. If we vectorize the image, the image becomes a point in $\mathcal{R}^{p^2}$. However, since features, such as eyes, mouth and nose, will be found in all images, the images will not be uniformly distributed in $\mathcal{R}^{p^2}$.

It is of special interest if Ω is a manifold.

Definition 2.1 (Manifold). A space $\mathcal{M} \subseteq \mathcal{R}^{n_\varphi}$ is said to be a n_z-dimensional *manifold* if there for every point $\varphi \in \mathcal{M}$ exists an open set $\mathcal{O} \subseteq \mathcal{M}$ satisfying:

- $\varphi \in \mathcal{O}$.
- $\mathcal{O}$ is *homeomorphic* to $\mathcal{R}^{n_z}$, meaning that there exists a one-to-one relation between $\mathcal{O}$ and a set in $\mathcal{R}^{n_z}$.

For details see *e.g.,* Lee (2000, p. 33).

For the set of $p \times p$ pixel images of human faces *e.g.,* the constraints implied by the different features characterizing a human face, make the images reside on a manifold enclosed in $\mathcal{R}^{p^2}$, see *e.g.,* Zhang et al. (2004). For *fMRI* (functional Magnetic Resonance Imaging) the situation is similar. For further discussions on fMRI data and manifolds, see Shen and Meyer (2005); Thirion and Faugeras (2004); Hu et al. (2006). Basically all sets of data for which data points can be parameterized using a set of parameters (fewer than the number of dimensions of the data) reside on a manifold. Any algebraic relation between regressor elements will therefore lead to regressors constrained to a manifold.

It is convenient to introduce the term *intrinsic description* for a n_z-dimensional parameterization of a manifold $\mathcal{M}$. We will not associate any properties to this description more than that it is n_z-dimensional. An intrinsic description of a one-dimensional manifold could for example be the distance from a specific point.

Remark 2.3. To express regressors in an intrinsic description is a way of doing feature extraction. Using an intrinsic description of the regressors instead of the original regressors in the regression algorithm may therefore be a way of making the regression problem well-posed, see *e.g.,* Ohlsson et al. (2007).

We illustrate the concepts of a manifold and intrinsic description with an example.

Example 2.5: Manifold and Intrinsic Description

Lines and circles are examples of one-dimensional manifolds. A two-dimensional manifold could for example be the surface of the earth. An intrinsic description associated with a manifold is a parametrization of the manifold, for example latitude and longitude for the earth surface manifold. Since the *Universal Transverse Mercator* (UTM) coordinate system is another two-dimensional parametrization of the surface of the earth and an intrinsic description, an intrinsic description is not unique.

A common assumption in regression is to assume smoothness. We will refer to the following assumption as the smoothness assumption:

Assumption A1 (The Smoothness Assumption). If two regressors φ_1, φ_2 are close, then so should their corresponding outputs $f_0(\varphi_1)$, $f_0(\varphi_2)$ be.

If regressors are constrained to a manifold there is an alternative to the smoothness assumption, commonly referred to as semi-supervised smoothness assumption. The semi-supervised smoothness assumption reads (Chapelle et al., 2006):

Assumption A2 (The Semi-Supervised Smoothness Assumption). Two outputs $f_0(\varphi_1)$, $f_0(\varphi_2)$ are assumed close if their corresponding regressors φ_1, φ_2 are close on the manifold.

"Close on the manifold" here means that there is a short path included in the manifold between the two regressors. The concept of geodesic distance is here useful. The *geodesic distance* between two points on a manifold $\mathcal{M}$ is the length of the shortest path included in $\mathcal{M}$ between the two points. The geodesic distance is assumed to be measured in the metric of the space in which the manifold is embedded. "Close on the manifold" can therefore be replaced by "close in terms of geodesic distance".

It should be noticed that the semi-supervised smoothness assumption is less conservative than the smoothness assumption. Hence, a function satisfying the semi-supervised smoothness assumption does not necessarily need to satisfy the smoothness assumption. Assumption A2 is illustrated in Example 2.6.

Example 2.6: The Semi-Supervised Smoothness Assumption

Assume that we are given a set of output-regressor pairs as shown in Figure 2.4. The regressors contain the position data (latitude, longitude) of an airplane

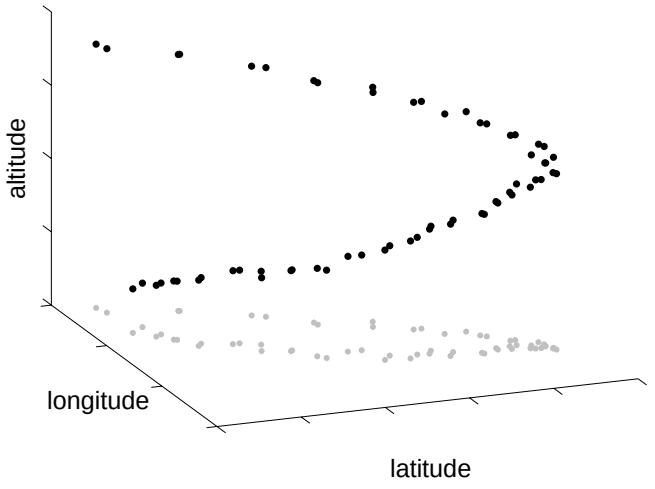

Figure 2.4: *Longitude, latitude and altitude measurement (black dots) of an airplane shortly after takeoff. Gray dots show the black dots projection onto the regressor space.*

shortly after takeoff. The output is chosen as the altitude of the airplane. The regressors thus being in $\mathcal{R}^2$ and the regressor/output space is $\mathcal{R}^3$. After takeoff the plane makes a turn during climbing and more or less returns along the same path in latitude and longitude as it just flown. The flight path becomes a one-dimensional curve, a manifold, in $\mathcal{R}^3$. However, the regressors for this path also belong to a curve, a manifold, in $\mathcal{R}^2$. This is therefore a case where the regressors are constrained to a manifold. The distance between two regressors in the regressor space can now be measured in two ways: the Euclidean $\mathcal{R}^2$ distance between points, and the geodesic distance measured along the curve, the manifold path. It is clear that the output, the altitude, is not a smooth function of regressors in the Euclidean space, since the altitudes vary substantially as the airplane comes back close to the earlier positions during climbing. However, if we use the geodesic distance in the regressor space, the altitude varies smoothly with regressor distance.

To see what the consequences are for predicting altitudes, suppose that for some reason, altitude measurements were lost for 8 consecutive time samples shortly after takeoff. To find a prediction for the missing measurements, the average of the three closest (in the regressor space, measured with Euclidean distance) altitude measurements were computed. The altitude prediction for one of the regressors is shown in Figure 2.5. The airplane turned and flew back on almost the same path as it just had flown, the three closest estimation regressors will

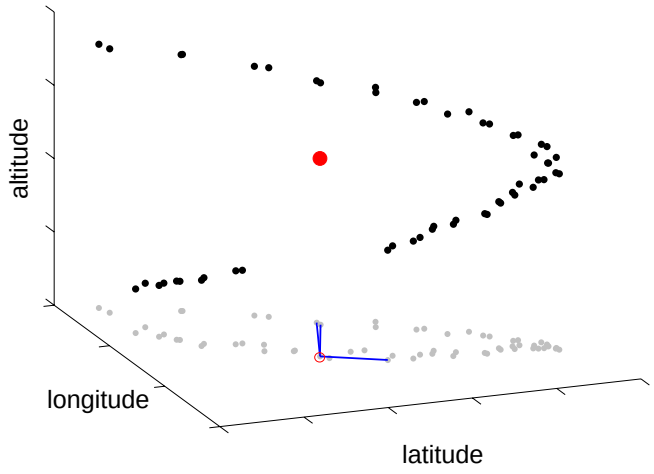

Figure 2.5: *The prediction of a missing altitude measurement (big filled circle). The encircled dot shows the position for which the prediction was computed. The three lines show the path to the three closest estimation regressors.*

therefore sometimes come from both before and after the turn. Since the altitude is considerably larger after the turn, the predictions will for some positions become heavily biased. In this case, it would have been better to use the three closest measurements along the flown path of the airplane. The example also motivates the semi-supervised smoothness assumption in regression.

Under the semi-supervised smoothness assumption, regression algorithms can be aided by incorporating the knowledge of a manifold. High-dimensional regression methods therefore have been modified to make use of the manifold and to estimate it (Belkin et al., 2006; Yang et al., 2006; Ohlsson et al., 2007). Since the regressors themselves contain information concerning the manifold, some regression methods use both regression-output pairs and regressors. This type of method is called *semi-supervised regression* or *semi-supervised modeling methods*. In contrast, in *supervised modeling* a relation between regressors and outputs is sought using a number of examples thereof *i.e.,* regression-output pairs. Most regression methods in system identification are supervised modeling methods. In *unsupervised modeling* the situation is rather different. Only one quantity is considered there and the task is rather to find patterns in the set of observations of this quantity. Semi-supervised modeling can be seen as a combination of supervised and unsupervised modeling.

2.11 Manifold Learning

Manifold learning is a fairly new research area aimed at finding, as the name suggests, descriptions of data on manifolds or intrinsic descriptions. The area has its roots in machine learning, and is a special form of *nonlinear dimensionality reduction* or *nonlinear feature extraction*. Some of the best known manifold learning algorithms are *isomap* (Tenenbaum et al., 2000), *Locally Linear Embedding* (LLE, Roweis and Saul (2000), discussed in the following section), *Laplacian eigenmaps* (Belkin and Niyogi, 2003) and *Hessian eigenmaps* (HLLE, Donoho and Grimes (2003)).

All manifold learning algorithms take as input a set of points sampled from some unknown manifold. The points are then expressed in a parameterization of the manifold, an intrinsic description (a set of points of the same dimension as the manifold), by searching for a set of new points preserving certain properties of the data. For example, Laplacian eigenmaps tries to preserve the Euclidean distance between neighboring points. Isomap tries to preserve the geodesic distances *i.e.,* the distance along the manifold, between points and locally linear embedding and Hessian eigenmaps make assumptions about local linearity and point neighborhoods which are aimed to be preserved. Manifold learning algorithms are unsupervised algorithms and most will not give an explicit expression for the map between high-dimensional points and their associated parameterization values.

2.11.1 Locally Linear Embedding

For finding intrinsic descriptions of data on a manifold, the manifold learning technique *Locally Linear Embedding* (LLE) can be used. LLE is a manifold learning technique which aims at preserving neighbors. In other words, given a set of points $\{\varphi_t\}_{t=1}^N$ residing on some n_z-dimensional manifold in $\mathcal{R}^{n_\varphi}$, LLE aims to find a new set of coordinates $\{z_1, \ldots, z_N\}$, $z_i \in \mathcal{R}^{n_z}$, satisfying the same neighbor-relations as the original points. The LLE algorithm can be divided into two steps:

Step 1: Define the w_{ij}s

Given data consisting of N real-valued vectors φ_i of dimension n_φ, the first step minimizes the cost function

$$\varepsilon(w) = \sum_{i=1}^{N} \left\| \varphi_i - \sum_{j=1}^{N} w_{ij}\varphi_j \right\|_2^2 \tag{2.58a}$$

with respect to w under the constraints

$$\begin{cases} \sum_{j=1}^{N} w_{ij} = 1, \\ w_{ij} = 0 \text{ if } \|\varphi_i - \varphi_j\|_2 > C_i(K) \text{ or if } i = j. \end{cases} \tag{2.58b}$$

Here, $C_i(K)$ is chosen so that only K weights w_{ij} become nonzero for every i. In the basic formulation of LLE, the number K and the choice of lower dimension $n_z \leq n_\varphi$ are the only design parameters, but it is also common to add a regularization

$$F_r(w) \triangleq \frac{r}{K} \sum_{i=1}^{N} [w_{i1}, \ldots, w_{iN}] \begin{bmatrix} w_{i1} \\ \vdots \\ w_{iN} \end{bmatrix} \sum_{j:w_{ij}\neq 0}^{N} \|\varphi_j - \varphi_i\|_2^2 \tag{2.59}$$

to (2.58a), see de Ridder and Duin (2002); Roweis and Saul (2000).

Step 2: Define the z_{ij}s

In the second step, w is now fixed. Let z_i be of dimension n_z and minimize

$$\Phi(z) = \sum_{i=1}^{N} \left\| {}_2 z_i - \sum_{j=1}^{N} w_{ij} z_j \right\|^2 \tag{2.60a}$$

with respect to $z = [z_1, \ldots, z_N]$, and subject to

$$\frac{1}{N} \sum_{i=1}^{N} z_i z_i^T = I \tag{2.60b}$$

using the weights w_{ij} computed in the first step. The solution z to this optimization problem is the desired set of n_z-dimensional coordinates which will work as an intrinsic description of the manifold. By expanding the squares we can rewrite

$\Phi(z)$ as

$$\Phi(z) = \sum_{i,j}^{N} (\delta_{ij} - w_{ij} - w_{ji} + \sum_{l}^{N} w_{li} w_{lj}) z_i^T z_j \tag{2.61a}$$

$$\triangleq \sum_{i,j}^{N} M_{ij} z_i^T z_j = \sum_{k}^{n_z} \sum_{i,j}^{N} M_{ij} z_{ki} z_{kj} = Tr(zMz^T) \tag{2.61b}$$

with M a symmetric $N \times N$ matrix with the ijth element

$$M_{ij} = \delta_{ij} - w_{ij} - w_{ji} + \sum_{l}^{N} w_{li} w_{lj}. \tag{2.62}$$

The solution to (2.60) is obtained by using *Rayleigh-Ritz theorem*, see *e.g.,* Horn and Johnson (1990, p. 176).

Theorem 2.2. *With Φ given by* (2.61)*, M by* (2.62) *and with v_i the unit length eigenvector of M associated with the ith smallest eigenvalue,*

$$\left[\begin{array}{c} v_1, \ldots, v_{n_z} \end{array} \right]^T = \arg\min_z \Phi(z) \quad s.t.\ zz^T = NI. \tag{2.63}$$

Remark 2.4. Notice that no explicit mapping is given, but more so an algorithm for computing an intrinsic description. If new points are introduced, the algorithm has to be rerun causing the intrinsic description for the old points to change.

The following example demonstrates how manifold learning or nonlinear feature extraction can be used in regression.

Example 2.7: Climate Reconstruction Cont'd

Let us now return to the climate reconstruction example in the introductory chapter, Example 1.1. Let us consider 10 shells grown in Belgium (see Ohlsson et al. (2009) for details). Since the temperature in the water had been monitored for these shells, this data set provides excellent means to test the ability to predict water temperature from chemical composition measurements. For these shells, the chemical composition measurements had been taken along the growth axis of the shells and paired up with temperature measurements. Between 30 and 52 chronologically ordered measurement were provided from each shell, corresponding to a time period of a couple of months.

Measurements from five of these shells are shown in Figure 2.6. The figure shows measurements of the relative concentrations of Sr/Ca, Mg/Ca and Ba/Ca (Pb/Ca is also measured, but not shown in the figure). The line shown between measurements connects the measurements coming from a shell and gives the chronological order of the measurements (two in time following measurements are connected by a line). As seen in the figure, measurements are highly restricted to a small region in the measurement space. Also, the water temperature (gray level coded in Figure 2.6) varies smoothly in the high-density regions. This together with that it is a biological process generating data, motivates the semi-

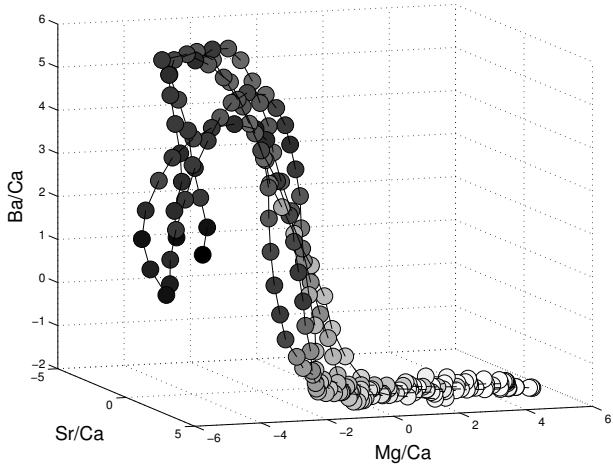

Figure 2.6: *A plot of the Sr/Ca, Mg/Ca and Ba/Ca concentration ratio measurements from five shells. Lines connects measurements (ordered chronologically) coming from the same shell. The temperatures associated with the measurements were color coded and are shown as different gray scales on the measurement points.*

supervised smoothness assumption when trying to estimate water temperature (outputs) from chemical composition measurements (4-dimensional regressors). Let us assume that the regressors are constrained to a one-dimensional manifold. LLE can then be applied to the regressors of the 10 shells to give a parameterization of the assumed one-dimensional manifold, an intrinsic description. This intrinsic description plotted against the measured water temperature is shown in Figure 2.7.

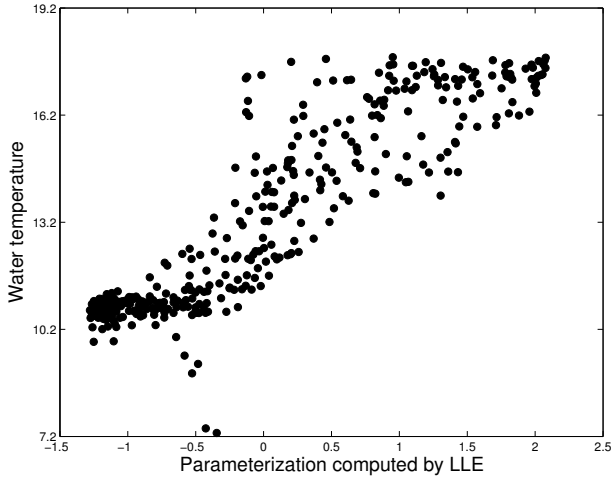

Figure 2.7: *The regressors (expressed using an intrinsic description) plotted against the measured water temperature. The intrinsic description was computed by using LLE.*

As seen in Figure 2.7, a linear estimate in the LLE parameterization would achieve a reasonably good estimate of the temperature.

2.12 Conclusion

This chapter served as an introduction to mathematical modeling and regression and introduced the fundamental knowledge and the necessary notation for the subsequent chapters. Several of the topics discussed are further discussed in papers of Part II. For example, impulse response identification discussed in Example 2.3 is the topic of Paper F and high dimensional regression, manifolds and manifold learning are discussed in Paper E.

3

State Estimation

Dynamic systems are characterized by that their output depends on current and past inputs. The effect that these inputs have had on the system is gathered in the *state*, which contains valuable information for *e.g.,* controllers and for decision making. It is a common situation that only parts of the state can be measured. Methods for recovering the full state of a dynamic system from these measurements are referred to as *state estimation techniques*. State estimation techniques use models to interpret the measured information.

3.1 The Standard Linear State-Space Model

The discrete-time standard linear state-space model with stochastic disturbances (see *e.g.,* Kailath et al. (2000, p. 161)) is given by

$$\begin{aligned} x_{t+1} &= A_t x_t + B_t u_t + G_t v_t, \\ y_t &= C_t x_t + e_t, \end{aligned} \tag{3.1a}$$

where x is the state, u a known input, v process noise, y the output and e the measurement noise. t index time. The process noise v and measurement noise e are here assumed to be zero mean *white noises* (see *e.g.,* Kailath et al. (2000, p. 4)): sequences of independent random vectors

$$\begin{aligned} \mathsf{E}[v_t] &= 0, & \mathsf{E}[e_t] &= 0 \quad \forall t \\ \mathsf{E}[v_t v_s^T] &= 0, & \mathsf{E}[e_t e_s^T] &= 0 \quad \text{if } t \neq s \\ \mathsf{E}[v_t v_t^T] &= Q_t, & \mathsf{E}[e_t e_t^T] &= R_t. \end{aligned} \tag{3.1b}$$

The independence of the noise sequences is required in order to make x_t a *Markov process*.

The model (3.1) with the process noise v being Gaussian is a standard model for control applications. v then represents the combined effect of all those non-measurable inputs that in addition to u affect the states. However, an equally common situation is that v corresponds to an *unknown input*. It could be

- a *load disturbance e.g.,* a step change in moment load of an electric motor, a (up or down) hill for a vehicle, *etc.* (Sometimes, the term load disturbance is used only for the case $B_t = G_t$.)
- an event that causes the state to jump, *a change*, see *e.g.,* Gustafsson (2001).

Such unknown inputs are not naturally modeled as Gaussian noise. Instead it is convenient to capture their unpredictable nature by (*cf.* eq (2.10)-(2.11) in Ljung (1999))

$$v_t = \delta_t \eta_t, \tag{3.2}$$

where (not to be confused with the Dirac delta function denoted by $\delta(\cdot)$)

$$\delta_t \triangleq \begin{cases} 0 & \text{with probability } 1-\mu, \\ 1 & \text{with probability } \mu, \end{cases} \qquad \eta_t \sim N(0, Q). \tag{3.3}$$

This makes $Q_t = \mu Q$ in (3.1b). The matrices A_t and G_t in (3.1a) may further model the waveform of the disturbance as a response to the pulse in v. Notice that if δ_t is known, v_t is Gaussian while an unknown δ_t leads to a non-Gaussian distributed v_t.

Example 3.1: DC Motor with Unknown Torque Load

Consider the discrete time model of a DC motor (see *e.g.,* Ljung (1999, pp. 95-97), $T_s = 0.1$ s, $\tau = 0.286$, $\beta = 40$)

$$\begin{aligned} x_{t+1} &= \begin{bmatrix} 0.7047 & 0 \\ 0.08437 & 1 \end{bmatrix} x_t + \begin{bmatrix} 11.81 \\ 0.6250 \end{bmatrix} (u_t + v_t), \\ y_t &= \begin{bmatrix} 0 & 1 \end{bmatrix} x_t + e_t. \end{aligned} \tag{3.4}$$

Here, x contains the angle and angular velocity of the motor shaft, y is noisy measurements of the motor shaft angle and u the applied voltage. The process noise v models a torque disturbance or an unknown torque load. Assuming that v is Gaussian is probably a bad assumption and in most applications a more sound assumption for v would probably be to model the process noise as in (3.2). The process noise v could also be set to pass through an integrator to model step changes.

We will get back to this example in Paper C and estimate the state x from the observed output y.

Example 3.2: Target Tracking

In *target tracking*, the goal is to estimate the state of a object given a number of sensor measurement. The object could be an airplane and the measurements, radar measurements, or it could be magnetometers placed in a crossing to track cars passing.

It is common to assume a dynamic motion model to model the kinematics of the object. The *continuous-time constant acceleration model* (see Chapter 13 in Gustafsson (2010)),

$$\begin{aligned} \dot{x}_t &= \begin{bmatrix} 0 & I_n & 0 \\ 0 & 0 & I_n \\ 0 & 0 & 0 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 0 \\ I_n \end{bmatrix} v_t, \\ y_t &= \begin{bmatrix} I_n & 0 & 0 \end{bmatrix} x_t + e_t, \end{aligned} \tag{3.5}$$

is a common choice. The state x contains the position, velocity and acceleration in n dimensions. The output y contains position measurements. The process noise v, the jerk (the derivative of the acceleration), is unknown and models the combined effect of all inputs that affect the state. e is the measurement noise of the sensor. The measurement noise e may very well be modeled by a Gaussian random variable. The lumped unknown inputs of the object gathered in v, however, is probably better modeled by *e.g.,* a piecewise constant signal. A piecewise constant signal is obtained by integrating a sequence of Dirac delta functions, this is illustrated in Figure 3.1.

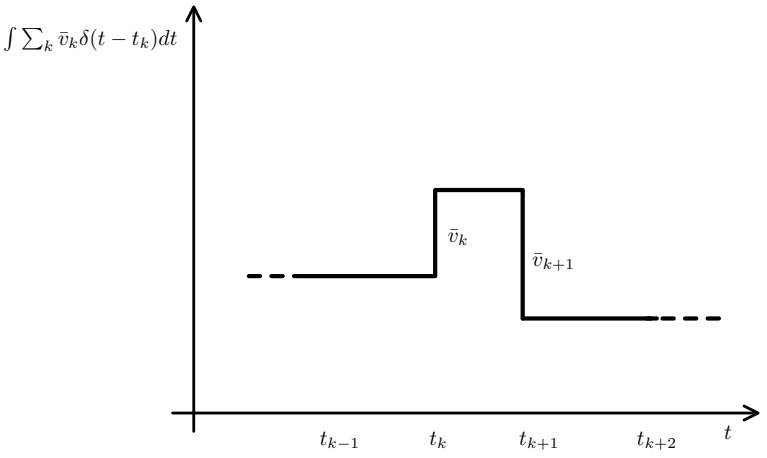

Figure 3.1: *Illustration of how a piecewise constant signal is obtained by integrating a sequence of Dirac delta functions. In this particular example there are impulses at t_k and t_{k+1} of sizes $\bar{v}_k$ and $\bar{v}_{k+1}$. These cause shifts of $\bar{v}_k$ and $\bar{v}_{k+1}$ at t_k and t_{k+1}.*

We can formulate this as

$$\begin{aligned}\dot{x}_t &= \begin{bmatrix}0 & I_n & 0 & 0\\ 0 & 0 & I_n & 0\\ 0 & 0 & 0 & I_n\\ 0 & 0 & 0 & 0\end{bmatrix} x_t + \begin{bmatrix}0\\0\\0\\I_n\end{bmatrix} \sum_k \bar{v}_k \delta(t - t_k),\\ y_t &= \begin{bmatrix}I_n & 0 & 0 & 0\end{bmatrix} x_t + e_t.\end{aligned} \tag{3.6}$$

Discretizing (3.6) with a sampling time $T_s = 0.1$ and under the assumption that $t_k = r_k T_s$, $r_k \in \mathcal{Z}$, give the discrete-time model (use *e.g.,* `sysd=c2d(sysc,Ts,'imp')` in MATLAB)

$$\begin{aligned}x_{kT_s+T_s} &= \begin{bmatrix}I_n & 0.1I_n & 0.005I_n & 0.0002I_n\\ 0 & I_n & 0.1I_n & 0.005I_2\\ 0 & 0 & I_n & 0.1I_n\\ 0 & 0 & 0 & I_n\end{bmatrix} x_{kT_s} + \begin{bmatrix}0.0002I_n\\0.005I_n\\0.1I_n\\I_n\end{bmatrix} \bar{v}_k,\\ y_{kT_s} &= \begin{bmatrix}I_n & 0 & 0 & 0\end{bmatrix} x_{kT_s}.\end{aligned} \tag{3.7}$$

To model $\bar{v}_k$ using the distribution given in (3.2) is now a good choice.

The relation between the ARX model (see (2.4)) and the state space model should be made clear. If we identify

$$x_t \leftrightarrow \theta_t,\ C_t \leftrightarrow \varphi_t^T,\ A_t \leftrightarrow I,\ B_t \leftrightarrow 0,\ G_t \leftrightarrow I, \tag{3.8}$$

the state space equation (3.1a) takes the form

$$\begin{aligned}\theta_{t+1} &= \theta_t + v_t,\\ y_t &= \varphi_t^T \theta_t + e_t,\end{aligned} \tag{3.9}$$

which is an ARX model with time varying parameters. This link between linear regression and state-space models is very well known, and described *e.g.,* in the classical survey by Åström and Eykhoff (1971). Possible knowledge of the parameter variations can be captured in more refined choices of A_t and G_t. θ is in (3.9) a *random walk*. If v is Gaussian, a (slowly) drifting model is described. For Gaussian noise v, the model (3.9) has been used to devise good tracking algorithms, *e.g.,* Section 11.6 in Ljung (1999). A piece-wise constant θ corresponds to a v as in (3.2) and that will be further discussed in Paper A.

3.2 State Estimation

Let us consider the estimation of x_t based on a set Y of the observations $\{y_t\}_{t=1}^N$. Write the estimate as

$$\hat{x}_t = F(Y). \tag{3.10}$$

There are two conceptually different cases:

- $\hat{x}_t$ is restricted to be a function of measurement up to and including time

t *i.e.,* $Y = \{y_t, y_{t-1}, y_{t-2}, \dots\}$. The estimation process is then referred to as *filtering*.

- $\hat{x}_t$ is based on measurements taken up to, including and later than time t *i.e.,* $Y = \{y_1, \dots, y_{t+1}, y_t, y_{t-1}, \dots, y_N\}$. The process of estimating x_t is then referred to as *smoothing*.

It is also common to distinguish between *linear* and *nonlinear filters* and *smoothers*. In linear filtering and smoothing F is a linear function of the elements in Y (and the initial state estimate). For a nonlinear filtering and smoothing algorithm, F is nonlinear in the elements of Y.

Two useful quantities when discussing filtering and smoothing are *bias* and *variance* of the estimate. A state estimate is said to be conditionally *unbiased* if

$$\mathsf{E}_{x_t}[\hat{x}_t - x_t | Y] = 0 \tag{3.11}$$

and otherwise conditionally *biased*. Note that this is equivalent to $\mathsf{E}_{x_t}[x_t|Y] = \hat{x}_t$. The conditional *covariance* of the estimate is given by

$$\mathsf{E}_{x_t}\left[\left(\hat{x}_t - x_t\right)\left(\hat{x}_t - x_t\right)^T \middle| Y\right]. \tag{3.12}$$

Alternatively, Y could be considered unknown and the expectations carried out over this quantity also. The state estimate is then said to be (unconditionally) unbiased if

$$\mathsf{E}_{x_t,Y}[\hat{x}_t - x_t] = 0. \tag{3.13}$$

The (unconditioned) covariance of the estimate is

$$\mathsf{E}_{x_t,Y}\left[\left(\hat{x}_t - x_t\right)\left(\hat{x}_t - x_t\right)^T\right]. \tag{3.14}$$

For the discrete-time standard linear state-space model with stochastic disturbances (3.1), the *Best Linear Unbiased Estimator* (BLUE) is given by the *Kalman Filter* (KF, Kalman (1960)) or *smoother* (*e.g.,* Kailath et al. (2000, p. 387)). We next give an introduction to the Kalman smoother and explain what "best" in "best linear unbiased estimator" refers to. We will only handle the smoothing case and not discuss filtering.

3.3 Kalman Smoother

In this thesis it is of interest to view the Kalman smoother as an explicit minimization problem. To arrive at the optimization formulation of the Kalman smoother, let first $\{y_t\}_{t=1}^N$ be a given set of observations satisfying (3.1) and let the initial state x_0 be a random variable independent of the noises e and v. Then, from (3.1b) it follows that the joint probability distribution can be written as

$$p\left(\{e_t\}_{t=1}^N, \{v_t\}_{t=1}^N, x_0\right) = p(x_0)\Pi_{t=1}^N p_e(e_t)p_v(v_t). \tag{3.15}$$

Assume now that $x_0 \sim N(0, \Gamma)$ and that e_t, v_t, $t = 1, \ldots, N$ are Gaussian distributed. Then (3.15) can be rewritten as

$$p\Big(\{e_t\}_{t=1}^N, \{v_t\}_{t=1}^N, x_0\Big) \propto e^{-\frac{1}{2}\|\Gamma^{-1/2}x_0\|_2^2} e^{-\frac{1}{2}\sum_{t=1}^N \|Q_t^{-1/2}e_t\|_2^2} e^{-\frac{1}{2}\sum_{t=1}^N \|R_t^{-1/2}v_t\|_2^2}. \tag{3.16}$$

Since, for $t = 1, \ldots, N$,

$$v_t = x_{t+1} - A_t x_t - B_t u_t, \quad e_t = y_t - C_t x_t, \tag{3.17}$$

(3.16) can be rewritten in terms of $\{x_t\}_{t=0}^N$ and $\{y_t\}_{t=1}^N$ as

$$\begin{aligned}\log p\Big(\{y_t\}_{t=1}^N \Big| \{x_t\}_{t=0}^N\Big) \propto &-\Big\|\Gamma^{-1/2}x_0\Big\|_2^2 - \sum_{t=1}^N \Big\|R_t^{-1/2}(y_t - C_t x_t)\Big\|_2^2 \\ &-\Big\|Q_{t-1}^{-1/2}(x_t - Ax_{t-1} - Bu_{t-1})\Big\|_2^2.\end{aligned} \tag{3.18}$$

Maximizing this quantity with respect to $\{x_t\}_{t=0}^N$ leads to the maximum likelihood estimate (MLE) for $\{x_t\}_{t=0}^N$. The MLE for $\{x_t\}_{t=0}^N$ can equivalently be written as

$$\underset{x_t, t=0,\ldots,N}{\arg\min} \|\Gamma^{-1/2}x_0\|_2^2 + \sum_{t=1}^N \Big\|R_t^{-1/2}(y_t - C_t x_t)\Big\|_2^2 + \Big\|Q_{t-1}^{-1/2}(x_t - A_{t-1}x_{t-1} - B_{t-1}u_{t-1})\Big\|_2^2 \tag{3.19}$$

which is recognized as the classical Kalman smoothing estimate, *e.g.,* Kailath et al. (2000, p. 387). Note that (3.19) is a (ℓ_2-regularized) least squares problem. The solution can therefore be shown to be linear in $\{y\}_{t=1}^N$ (and x_0). The solution is usually given in various recursive filter forms, see *e.g.,* Ljung and Kailath (1976).

When all densities are Gaussian (e_t, v_t, x_0 Gaussian), (3.19) gives the best unbiased estimate (among both linear and nonlinear estimators) since no other unbiased estimator can obtain a smaller variance. That is, let $\hat{x}_t$ be the Kalman estimate and let $\bar{x}_t$ be any other unbiased state estimate. Then, with expectation over both x_t and Y,

$$\mathsf{E}_{x_t,Y}\Big[(\bar{x}_t - x_t)(\bar{x}_t - x_t)^T\Big] - \mathsf{E}_{x_t,Y}\Big[(\hat{x}_t - x_t)(\hat{x}_t - x_t)^T\Big] \geq 0. \tag{3.20}$$

This also implies that no other unbiased estimator can obtain a lower MSE *i.e.,*

$$tr\,\mathsf{E}_{x_t,Y}\Big[(\hat{x}_t - x_t)(\hat{x}_t - x_t)^T\Big] = \mathsf{E}_{x_t,Y}\Big[(\hat{x}_t - x_t)^T(\hat{x}_t - x_t)\Big]. \tag{3.21}$$

(3.20) and (3.21) also hold if the expectations is taken w.r.t x_t and conditional on Y. It further holds that x_t given $\{y_t\}_{t=1}^N$ is Gaussian (the mean given by $\hat{x}_t$, *i.e.,* $\hat{x}_t = \mathsf{E}[x_t|y_1, \ldots, y_N]$).

If e_t, v_t or x_0 is not Gaussian, the Kalman smoother is still the best unbiased linear estimator. That means that we can not do better than using a Kalman smoother if v is distributed as (3.2), the sequence δ_t is unknown and the smoother is restricted to be linear. If we knew the δ_t-sequence (and the measurement noise was Gaussian), the Kalman smoother would be the best estimator among both linear and nonlinear estimators, since all noises would be Gaussian (with time varying noise covariance).

See Anderson and Moore (1979, Chap. 7) for more on the Kalman smoother and its properties.

3.4 Kalman Filter (Smoother) Banks

Based on the process noise model (3.2), a number of nonlinear methods have been developed. If δ_t is unknown, we could hypothesize in each time step that it is either 0 or 1. This leads to a large bank (2^N) of Kalman smoothers as the optimal solution. The posterior probability of each smoother can be estimated from this bank, which consists of a weighted sum of the state estimates from each smoother. See Chapter 10 in Gustafsson (2010) for more on smoother banks.

In practice, the number of smoothers in the bank must be limited due to computational limitations, and there are two main options (see Chapter 10 in Gustafsson (2010)):

- Merging trajectories of different δ_t sequences. This includes the well known *Interacting Multiple Model* filter (IMM filter, Blom and Bar-Shalom (1988)).
- Pruning, where unlikely sequences are deleted from the filter bank.

3.5 Conclusion

This chapter gave a brief introduction to filtering and smoothing. We continue the discussion on smoothing and impulsive process noise in Paper C. In particular we explore the fact that the sequence generated by (3.2), arranged as a vector, contains elements identical to zero, it will be a *sparse* vector. This leads us to the concept of sparseness and regularization for sparseness. Sparseness and regularization for sparseness are discussed in the next chapter, Chapter 4.

4

Regularization for Sparseness

Sparseness is all about zeros. A matrix or vector is said to be *sparse* if it contains a relatively large number of zeros. If a quantity is given to be sparse, it is often a computational remedy *e.g.,* when solving equation systems or in optimization. However, sparsity has also shown great importance for other reasons, in *e.g.,* statistical learning and signal processing. The hype around sparsity in statistical learning is mostly due to the success of *lasso* (least absolute shrinkage and selection operator, Tibsharani (1996); Chen et al. (1998), see also Hastie et al. (2001, p. 64)) and in signal processing sparsity has got attention due to the sampling protocol *Compressed Sensing* (CS, Donoho (2006); Candès et al. (2006)).

Formally, sparse is defined as (see *e.g.,* Zibulevsky and Elad (2010)):

Definition 4.1 (Sparse). A vector $z \in \mathcal{R}^n$ is said to be sparse if

$$\|z\|_0 \ll n. \tag{4.1}$$

$\|\cdot\|_0$ here denotes the zero (quasi-)norm. The zero norm is the number of nonzero elements of a vector (see Appendix A).

4.1 When is Sparsity a Desirable Property?

Sparsity is wanted in various situations. Sparsity can *e.g.,* be used for variable selection, as in lasso, for image denoising and filter design as in Starck et al. (2002); Bioucas-Dias (2006) or as a sample protocol, as in compressed sensing. What the above applications have in common is that the underlying problem has a combinatorial nature. The problem could *e.g.,* be to select a subset of variables, basis

functions, times instances *etc.* that solves some problem in an optimal manner.

The following three examples give a flavor for when, where and how sparseness can be used. We will revisit these examples at later points of the chapter as well.

Example 4.1: Lasso

Consider the task of estimating a linear regression model

$$f(\varphi, \theta) = \varphi^T \theta. \tag{4.2}$$

Assume that an estimation data set $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$, $y_t \in \mathcal{R}$, $\varphi_t \in \mathcal{R}^{n_\varphi}$ is given for this purpose. Also assume that $n_\varphi > N_e$. Minimizing the sum of squared residuals

$$\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 \tag{4.3}$$

to determine θ leads to an ill-posed problem (see Example 2.2). In particular, the solution will not be unique. We saw previously how ℓ_2-regularization (see Example 2.2) can be used to transform (4.3) into a well-posed problem

$$\min_\theta \sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 + \lambda \|\theta\|_2^2, \quad \lambda \in \mathcal{R}^+. \tag{4.4}$$

The ℓ_2-regularization added in (4.4) favors small $\|\theta\|_2^2$. However, typically all θ-elements turn out non-zero and it may therefore be difficult to understand which regressor elements that are meaningful. Besides, one also needs to continue to acquire the whole regressor vector φ to use the model. If each element in φ_t requires a measurement to be done, acquiring the whole regressor vector may be impractical if n_φ is large.

The idea of lasso is to find a regression parameter θ so that the model (4.2) gives a good fit to the estimation data *i.e.,* makes

$$\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 \tag{4.5}$$

small and at the same time obtain a θ which is sparse. The sparsity constraint will cause a large number of θ-elements to be zero. Lasso therefore gives the possibility to interpret and say what regression elements that are meaningful for a good prediction result. Zeros in θ mean that the associated regressor elements are not needed, time and money can therefore be saved by only measuring the φ-elements associated with non-zero θ-elements. The idea of lasso leads to a criterion

$$\min_\theta \sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 + \lambda \|\theta\|_0, \quad \lambda \in \mathcal{R}^+. \tag{4.6}$$

We will come back to lasso and the mathematical details in Example 4.4.

Example 4.2: Compressed Sensing Cont'd

Let us return to the discussion of audio compression and sampling given in the introductory part of the thesis, Example 1.3. We there argued that it was rather meaningless to measure a lot of information if 90% will be thrown away before someone even listened to the song, or as Donoho (2006) wrote,

> "Why go to so much effort to acquire all the data when most of what we get will be thrown away? Can we not just directly measure the part that will not end up being thrown away?"

In an MP3 encoder, the audio stream is divided into several frequency bands. The audio of a frequency band is then discarded if it is weaker than some certain threshold (Brandenburg, 1999; Hayes, 2009). The problem is that even though an audio recording can well be represented using the audio in a small number of frequency bands, we do not know what bands that are going to be discarded before we start sample. We therefore need to sample all frequency bands and then compress and throw away a major part of our sampled data. This was what many thought before compressed sensing was introduced in 2006.

Let $x \in \mathcal{R}^{n_x}$ be a quantity that we are interested in. In *compressed sensing* (also known as *compressive sensing*, *compressive sampling*, *compressed sampling*) it is assumed that the signal x is composed of a very limited number of *atoms* from a *dictionary* containing a large number of typical signal shapes or *basis functions*. Let these signal shapes be columns in the matrix $A \in \mathcal{R}^{n_x \times n_z}$, typically $n_z \gg n_x$. The signal x is hence assumed to have the property

$$x = Az, \quad z \in \mathcal{R}^{n_z} \text{ sparse.} \tag{4.7}$$

A dictionary, or A, that has these properties is in compressed sensing assumed known. It could *e.g.*, be suitable to chose a dictionary containing sampled sine and cosine signals of difference frequencies if x contains a sequence of audio samples.

Remark 4.1. All signals that people find meaningful can be decomposed as in (4.7) (Hayes, 2009). A sequence of independent random numbers is an example of a signal that can not be decomposed using a sparse z.

Let $M \in \mathcal{R}^{n_y \times n_x}$, $n_x \gg n_y$ and define $y \in \mathcal{R}^{n_y}$ by

$$y \triangleq Mx = MAz. \tag{4.8}$$

What is important is that y has considerably lower dimmension than x. Hence, y can be seen as a compressed version of x. The idea of compressed sensing is now to measure y rather than x. That is, to measure a few linear combinations of the elements in x rather than x directly. This means that we should construct a number (n_y) of microphones that each give a sample which *e.g.*, is a weighted average of sounds during the last second. The microphones should not be identical, they all need to form different weighted averages. Assume also that the weights used to form these averages are known, that is, we know M.

We now have y, which we have acquired using less sampling and can store using

less memory space than x would have needed. How do we recover x so to be able to listen to the second of audio?

Since M, A and y are all known and z was assumed sparse, it is natural to seek for an estimate $\hat{z}$ using

$$\hat{z} = \arg\min_z \|z\|_0 \quad \text{s.t. } y = MAz. \tag{4.9}$$

We can then obtain $\hat{x}$ as $\hat{x} = A\hat{z}$. What is remarkable is that under certain rather mild assumptions on the matrices A and M and if z satisfies (4.7) and is sufficiently sparse, $\hat{x} = x$ (see *e.g.,* Bruckstein et al. (2009)). The audio can hence be perfectly recovered even though $n_x \gg n_y$!

We will return to compressed sensing in Example 4.5 and there present the mathematical details.

Remark 4.2 (Nyquist-Shannon Sampling Criterion and Compressed Sensing). The *Nyquist-Shannon sampling criterion* states that for a *bandlimited signal* (no energy above some certain frequency) the sampling frequency should be twice that of the bandlimit to guarantee the possibility to perfectly reconstruct the time-continuous signal (see *e.g.,* Oppenheim et al. (1996, p. 519)). With no further information, to use a sample frequency twice that of the bandlimit is actually the best thing to do (Tropp et al., 2010). However, if the signal is known to be *e.g.,* a combination of a few basis functions, a perfect reconstruction can be obtained at a lot lower sampling frequencies.

Example 4.3: The Huber Loss Function

Consider the following setup

$$y_t = \varphi_t^T \theta_0 + e_t + \tau_t, \quad y_t \in \mathcal{R},\ e_t \sim N(0, \sigma^2), \tag{4.10}$$

where $\theta_0 \in \mathcal{R}^{n_\theta}$ is an unknown vector and e_t the measurement noise. The scalar variable τ_t models an *outlier* and will therefore be zero for most t but occasionally non zero. Let $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$ be a given estimation data set.

Desiring an estimate of θ_0, we can use the least squares estimate,

$$\hat{\theta}_{ls} = \arg\min_\theta \sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 \tag{4.11a}$$

$$= \arg\min_\theta \sum_{t=1}^{N_e} (\varphi_t^T \theta_0 + e_t + \tau_t - \varphi_t^T \theta)^2. \tag{4.11b}$$

If $\tau_t \neq 0$ for some $t = 1, \ldots, N_e$, it is likely that the estimate of θ_0 is adjusted to fit these fluctuations in τ_t. We can try to get around this by estimating τ_t and then subtract the estimate from our measurements y_t.

As outliers, by definition, appear seldom, a realization of the time series $\{\tau_t\}_{t=1}^{N_e}$

arranged as a vector, will be a sparse vector. We are therefore led to consider

$$\min_{\theta,\eta_1,\eta_2,\ldots,\eta_{N_e}} \sum_{t=1}^{N_e} (y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda \left\| [\eta_1\ \eta_2 \ldots \eta_{N_e}]^T \right\|_0, \tag{4.12}$$

for some $\lambda \in \mathcal{R}^+$. Here, $\{\eta_t\}_{t=1}^{N_e}$ serves as an estimate of the realization of $\{\tau_t\}_{t=1}^{N_e}$ associated with estimation data $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$. We return to this example in Example 4.6.

Bruckstein et al. (2009); Zibulevsky and Elad (2010) further exemplify and motivate sparsity in signal processing and modeling.

4.2 Methods for Obtaining Sparsity

The ℓ_0-norm causes optimization problems to be non-convex and combinatorial. Solving the optimization problem (4.9)

$$\min_z \|z\|_0, \quad \text{s.t. } y = MAz, \tag{4.13}$$

e.g., boils down to an exhaustive combinatorial search: Fix all element in z except the first to zero and check if there is a z satisfying $y = MAz$. If not, continue by fixing all except the second element to zero and check if there is a z satisfying $y = MAz$. Go through the whole vector z if necessary, letting one element free and fixing all other to zero, one by one. If no z satisfying $y = MAz$ is found, go through different combinations of two nonzero elements in a search for a z satisfying $y = MAz$. And so on. See *e.g.,* Bruckstein et al. (2009). Not very surprising, (4.13) can actually be shown to be NP-hard (Natarajan, 1995).

The optimization problems (4.6) and (4.12) are of the form

$$\min_\theta \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_0, \quad \lambda \in \mathcal{R}^+, \tag{4.14}$$

and are also in general NP-hard (Huo and Ni, 2007). If the measurements (4.8) in compressed sensing are noisy, an optimization problem of the form (4.14) replaces (4.9), see Candès et al. (2006). Note that many model selection criteria *e.g.,* AIC (see (2.10) for AIC) has also the form (4.14) for a linear regression model, see *e.g.,* Huo and Ni (2007).

The combinatorial optimization problem that (4.13) and (4.14) lead to is often impractical to solve and several approximation techniques have therefore been proposed.

Greedy algorithms (see *e.g.,* Tropp (2004); Bruckstein et al. (2009)) start with a z identical to zero (or θ identical to zero if (4.14) is considered). Greedy algorithms then let the element in z which *e.g.,* increases the fit the most free and estimate z. The greedy algorithm then continues by, one by one, letting the z element that leads to the best fit free and re-estimating z. The algorithm terminates when a good enough fit has been obtained. Under the assumption that the z solving

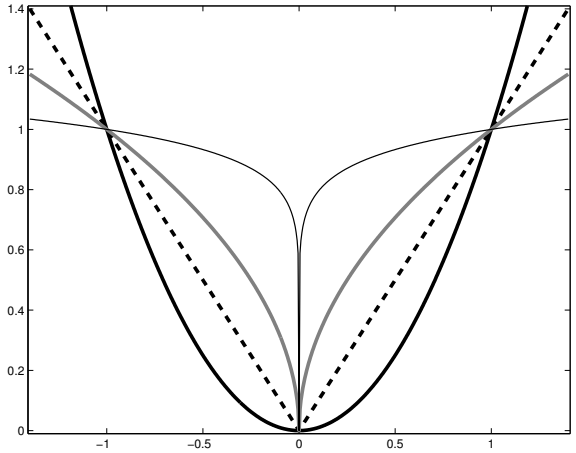

Figure 4.1: *For a one-dimensional variable, the (squared) ℓ_2-norm, $(\cdot)^2$, with solid black thick line, the ℓ_1-norm, $|\cdot|$, showed with dashed black line, $|\cdot|^{1/2}$ with gray line and $|\cdot|^{1/10}$ with solid thin black line.*

(4.13) is sufficiently sparse and MA sufficiently *incoherent* (see *e.g.,* Candès et al. (2010)) *i.e.,*

$$\max_{j<k} \frac{|(MA)(:,j)^T(MA)(:,k)|}{\|(MA)(:,j)\|_2\|(MA)(:,k)\|_2} \ll 1, \tag{4.15}$$

some greedy algorithms give the same solution as that of (4.13), see *e.g.,* Bruckstein et al. (2009). For the problem (4.14) the correct support can be guaranteed if the solution of (4.14) is sufficiently sparse, the signal to noise ratio is sufficiently good and Φ sufficiently incoherent, see *e.g.,* Bruckstein et al. (2009). Many variants of greedy algorithms exist. *Forward stepwise regression* (see *e.g.,* Daniel and Wood (1980, pp. 84-85), known as *matching pursuit* in signal processing, see *e.g.,* Mallat and Zhang (1993)) may be the one most known to the system identification community. However, also *e.g., Least Angle Regression* (LARS, Efron et al. (2004)) is a variant of greedy algorithm.

The *FOCUSS* (FOCal Underdetermined System Solver, see *e.g.,* Bruckstein et al. (2009)) method is another approximation method. In FOCUSS an approximation to (4.13) is sought by searching for a local minimum of the ℓ_p, $0 < p < 1$, regularized problem

$$\min_z \|z\|_p, \quad \text{s.t. } y = MAz. \tag{4.16}$$

This is an non-convex problems.

The "closest" convex problem to (4.13) and (4.14) is obtained by replacing the ℓ_0-norm with the ℓ_1-norm, see Figure 4.1. This is a *convex relaxation* of the problem. If (4.13) is relaxed by replacing the zero-norm with the ℓ_1-norm,

$$\min_z \|z\|_1, \quad \text{s.t. } y = MAz, \tag{4.17}$$

we obtain what is referred to as the *basis pursuit* (Chen et al., 1998). This problem can be solved using linear programming (see *e.g.,* Donoho (2006)).

If (4.14) is relaxed by replacing the zero-norm with the ℓ_1-norm,

$$\min_{\theta} \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_1, \quad \lambda \in \mathcal{R}^+, \tag{4.18}$$

we obtain what is referred to as the *basis pursuit denoise* (Chen et al. (1998)) in the signal processing community and lasso in the statistical community. The problem given in (4.18) is a ℓ_1-regularized least squares problems. The next section discusses the usage of ℓ_1 regularization for obtaining sparsity.

4.3 ℓ_1-Regularization

ℓ_1-regularization is by no means a new concept (see Appendix I of Tropp (2006) for a historical review). In fact, it has been a regularization technique and a known way to obtain sparsity since the 1970s. It has gained a lot of popularity and publicity lately though.

A ℓ_1-regularized problem has the form

$$\min_{\theta} V(\theta) + \lambda\|\theta\|_1, \quad \lambda \in \mathcal{R}^+, \tag{4.19}$$

where V is the criterion of fit, $\|\cdot\|_1$ the ℓ_1-norm and λ is the regularization parameter. The criterion of fit $V(\theta)$ is often the least squares criterion $\|y - \Phi\theta\|_2^2$, as in (4.18), but there are many other interesting choices, *e.g.,* Riezler and Vasserman (2004); Chen et al. (2009).

For the ℓ_1-regularized least squares procedure ($V(\theta) = \|y - \Phi\theta\|_2^2$ in (4.19))

$$\min_{\theta} \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_1, \tag{4.20}$$

it has been shown that the solution (for a proper value for λ) has the same zero elements (but possibly more) as the solution of

$$\min_{\theta} \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_0, \tag{4.21}$$

if Φ is sufficiently incoherent (see (4.15)) and the measurement noise weakly correlated with Φ (Tropp, 2006). The solution may however not be unique, since (4.20) is not necessarily strictly convex, see *e.g.,* Bertsekas et al. (2003, Prop. 2.1.2)).

Example 4.4: Lasso Cont'd

Let us return to Example 4.1 and lasso. The idea in lasso is to find a θ so that the a linear model (4.2) gives a good fit to the estimation outputs and at the same time obtains a θ which is sparse. We formulated this as

$$\min_{\theta} \sum_{t=1}^{N_e} (y_t - \varphi_t^T\theta)^2 + \lambda\|\theta\|_0, \tag{4.22}$$

for some $\lambda \in \mathcal{R}^+$. This problem is combinatorial and a convex relaxation is therefore used to obtain the ℓ_1-regularized least squares, or the lasso, criterion

$$\min_{\theta} \sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 + \lambda \|\theta\|_1. \tag{4.23}$$

The ℓ_1-regularization in lasso penalizes elements of θ different than zero and therefore causes elements of θ that do not provide a significant decrease of the fit term to be zero. The property of the ℓ_1-regularization that causes elements to become identical to zero, and not only small as in the ℓ_2-regularization, is discussed in Section 4.3.1.

The θ resulting from solving (4.23), let us say $\hat{\theta}$, will be biased since terms that do provide a better fit also are being penalized and dragged towards zero. The bias is often adjusted for by re-estimating the regression parameters according to

$$\min_{\theta} \sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 \quad \text{s.t. } \theta(i) = 0 \text{ if } \hat{\theta}(i) = 0,\ i = 1, \ldots, dim(\hat{\theta}), \tag{4.24}$$

with $\hat{\theta}$ from (4.23). This makes the regression parameter unbiased if lasso correctly identified the zero elements in θ.

Example 4.5: Compressed Sensing Cont'd

We now return to Example 4.2 to carry out the mathematical details. We argued that to reconstruct x it was natural to seek for an estimate $\hat{z}$ using

$$\hat{z} = \arg\min_{z} \|z\|_0 \quad \text{s.t. } y = MAz, \tag{4.25}$$

and then obtain $\hat{x}$ as $\hat{x} = A\hat{z}$. Due to the combinatorial complexity of (4.25) we are led to consider *e.g.,* a convex relaxation, such as replacing the ℓ_0-norm with the ℓ_1-norm. If the measurements y are noisy, the equality constraint in (4.25) is removed and $\|y - MAz\|_2^2$ added to the objective function. We are led to consider the ℓ_1-regularized least-squares problem

$$\hat{z} = \arg\min_{z} \|y - MAz\|_2^2 + \lambda \|z\|_1, \quad \lambda \in \mathcal{R}^+. \tag{4.26}$$

What is remarkable is that with considerably fewer samples than what the Nyquist-Shannon sampling criterion would have told you to use and with the relaxed ℓ_0-norm, a close to perfect reconstruction of x can be obtained, see *e.g.,* Candès and Wakin (2008). In fact, it has been shown that compressed sensing is nearly as effective as if having an oracle telling us what elements of z that are nonzero and we would have measured only those (Candès and Wakin, 2008).

Example 4.6: The Huber Loss Function Cont'd

Let use finally return to Example 4.3. We there assumed that we got measurements $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$ from

$$y_t = \varphi_t^T \theta_0 + v_t + \tau_t, \quad v_t \sim N(0, \sigma^2), \tag{4.27}$$

where τ_t modeled outliers and was assumed to be an in time sparse variable. The model parameter θ_0 is an unknown vector. Desiring an estimate of θ_0, we can use the least squares estimate,

$$\theta_{\text{ls}} = \arg\min_{\theta} \sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 \tag{4.28a}$$

$$= \arg\min_{\theta} \sum_{t=1}^{N_e} (\varphi_t^T \theta_0 + e_t + \tau_t - \varphi_t^T \theta)^2. \tag{4.28b}$$

If $\tau_t \neq 0$ for some $t = 1, \ldots, N_e$, it is likely that the estimate of θ_0 is adjusted to fit these fluctuations in τ_t. We can try to get around this by estimating τ_t and then subtract the estimate from our measurements y_t. As we have assumed that τ_t is sparse, it is motivated to minimize

$$\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta - \eta_t)^2 + \lambda \left\| \begin{bmatrix} \eta_1 & \eta_2 & \ldots & \eta_{Ne} \end{bmatrix} \right\|_0, \quad \lambda \in \mathcal{R}^+, \tag{4.29}$$

with respect to the outlier estimate η and θ. Here, λ is seen as a design parameter that controls the sparsity of η. Using a convex relaxation, we arrive at the less computationally intensive ℓ_1-regularized least squares problem

$$\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta - \eta_t)^2 + \lambda \left\| \begin{bmatrix} \eta_1 & \eta_2 & \ldots & \eta_{Ne} \end{bmatrix} \right\|_1. \tag{4.30}$$

As shown in Appendix B, (4.30) is equivalent to

$$\sum_{t=1}^{N_e} \psi\big(y_t - \varphi_t^T \theta\big) \tag{4.31}$$

with

$$\psi(x) \triangleq \begin{cases} |x|^2, & \text{if } |x| < \lambda/2, \\ \lambda|x| - \lambda^2/4 & \text{otherwise.} \end{cases} \tag{4.32}$$

The function $\psi(\cdot)$ is called the *Huber loss function* or the *Huber norm* (Huber, 1973). The Huber loss function has been applied frequently within regression and classification since its introduction in the 1970s by Huber. Its popularity is due to its ability to reduce the affect of an outlier and thereby gain robustness to the algorithm. The Huber loss function, $\psi(\cdot)$, shown in Figure 4.2, is a hybrid between the ℓ_1 and the ℓ_2-norm. That the assumption of a sparse outlier τ here leads to the Huber loss function is rather intuitive but still illustrative.

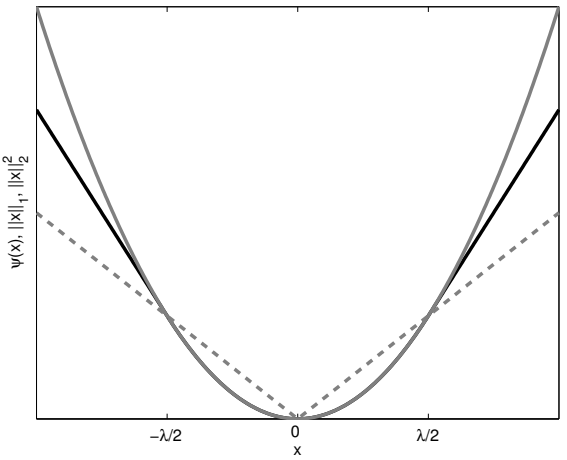

Figure 4.2: *The Huber loss function $\psi(x)$ plotted with thick solid black line for a one-dimensional x. The ℓ_1 and squared ℓ_2-norm are also shown, dashed and solid gray line, respectively.*

It is interesting to notice that minimizing

$$\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 + \lambda \|\theta\|_1, \tag{4.33}$$

$\lambda \in \mathcal{R}^+$, can be interpreted as a MAP estimate of a posterior distribution proportional to

$$e^{-\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 / 2\sigma^2} e^{-\lambda \|\theta\|_1 / 2\sigma^2}. \tag{4.34}$$

Using Bayes' theorem, see Theorem 2.1 on page 26, the first term in (4.34) can be interpreted as the likelihood

$$p\left(\{y_t\}_{t=1}^{N_e} \middle| \theta, \{\varphi_t\}_{t=1}^{N_e}\right) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\sum_{t=1}^{N_e} (y_t - \varphi_t^T \theta)^2 / 2\sigma^2} \tag{4.35}$$

associated with

$$y_t = \varphi_t^T \theta_0 + e_t, \quad e_t \sim N(0, \sigma^2). \tag{4.36}$$

The second term in (4.34) can be interpreted as a prior $p(\theta) = \frac{1}{4\sigma^2} e^{-\lambda \|\theta\|_1 / 2\sigma^2}$. The prior associated with the ℓ_1-regularization is hence $p(\theta) = \frac{1}{4\sigma^2} e^{-\lambda \|\theta\|_1 / 2\sigma^2}$. In the literature this is referred to as a *Laplace* or an *independent double exponential* prior (see *e.g.,* Hastie et al. (2001, p. 72)).

4.3.1 What Property of the ℓ_1-Regularization Causes Sparseness?

Let us investigate why ℓ_1-regularization causes sparseness. Consider

$$\min_{\theta} \|y - \Phi\theta\|_2^2 \quad \text{s.t. } \|\theta\|_1 \leq \eta. \tag{4.37}$$

This problem is identical to that of (4.20) in the sense that, for any $\lambda \in \mathcal{R}^+$, there exists a η ($= \|\theta^*\|_1$, where θ^* minimizes (4.20)) so that the minimizing θ is the same for (4.20) and (4.37). The *Karush-Kuhn-Tucker* (KKT, see *e.g.,* Boyd and Vandenberghe (2004, p. 244)) conditions can be used to show this.

Consider now the left of Figure 4.3. The gray square at the origin shows the

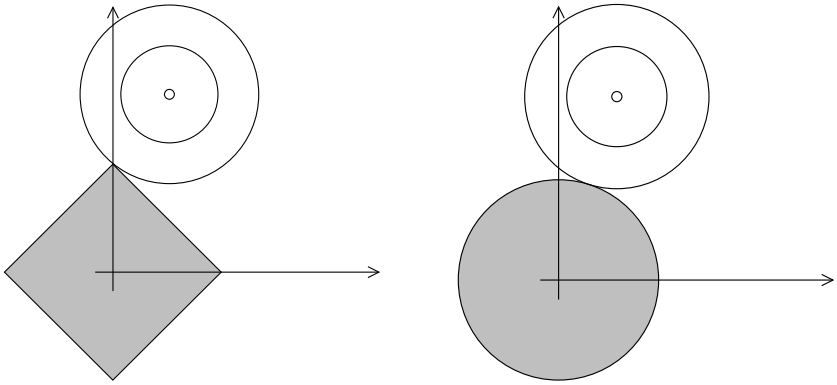

Figure 4.3: *Left figure: An illustration of $\|\theta\|_1 \leq \eta$ (gray area) and the level-curves of $\|y - \Phi\theta\|_2^2$. Right figure: An illustration of $\|\theta\|_2^2 \leq \eta$ (gray area) and the level-curves of $\|y - \Phi\theta\|_2^2$. In both the right and the left figure, $\|y - \Phi\theta\|_2^2$ is assumed to have a unique minimum. If $\|y - \Phi\theta\|_2^2$ does not have a unique minimum, there will be a continuum of points, on a line, minimizing $\|y - \Phi\theta\|_2^2$ and the level curves would be parallel to that line.*

neighborhood $\|\theta\|_1 \leq \eta$ for a two dimensional regressor (*i.e.,* $dim(\theta) = 2$). The level-curves of $\|y - \Phi\theta\|_2^2$ are also shown. These are depicted as circles (generally these level curves are ellipses) centered at $\arg\min_\theta \|y - \Phi\theta\|_2^2$. From the illustration it is seen that the θ minimizing (4.37) must be the θ-value at the intersection between the square and one of the level-curves. Note now that when this intersection happens on one of the axis, the optimal θ get one zero element. Try to move around the level-curves of $\|y - \Phi\theta\|_2^2$. Most choices gives an intersection at an axis. For a higher dimensional case ($dim(\theta)$ large), the gray square turns into a hyper-cube. When intersection happens on *e.g.,* one of the vertexes, the optimal θ has elements equal to zero and therefore turns out as sparse.

Consider now the right part of Figure 4.3. The right part illustrates what happens if the regularization is chosen as $\|\cdot\|_2^2$ (ridge regression, see Example 2.2) instead

of $\|\cdot\|_1$ as in the ℓ_1-regularization. Consider

$$\min_{\theta} \|y - \Phi\theta\|_2^2 \quad \text{s.t. } \|\theta\|_2^2 \leq \eta, \tag{4.38}$$

which for a particular choice of η gives the same solution as

$$\min_{\theta} \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_2^2. \tag{4.39}$$

The gray circle now illustrates $\|\theta\|_2^2 \leq \eta$ which is a disc centered at the origin. The level-curves of $\|y - \Phi\theta\|_2^2$ are also shown, just as in the left of Figure 4.3. The solution to (4.38) can now be seen given by the intersection between the disc and a level-curve. Try to move the level-curves around, the intersection is this time very seldom on an axis. The minimizing θ will therefore generally not be sparse.

An Explicit Solution

For illustration, let us consider a special case which has an explicit solution. Consider

$$\min_{\theta} \|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_1, \tag{4.40}$$

where $\lambda \in \mathcal{R}^+$, $\theta \in \mathcal{R}^{n_\theta}$, and assume that Φ is orthonormal, *i.e.,* $\Phi^T\Phi = \Phi\Phi^T = I$. Equation (4.40) can then be rewritten as

$$\min_{\theta} \left\|\Phi(\Phi^T y - \theta)\right\|_2^2 + \lambda\|\theta\|_1, \tag{4.41}$$

and since Φ can be seen as a rotation, which does not change the Euclidean length, of the vector $(\Phi^T y - \theta)$, $\left\|\Phi(\Phi^T y - \theta)\right\|_2^2 = \left\|\Phi^T y - \theta\right\|_2^2$. We can further rewrite $\|\Phi^T y - \theta\|_2^2$ using that $\Phi^T\Phi = I$ so that $\|\Phi^T y - \theta\|_2^2 = \left\|(\Phi^T\Phi)^{-1}\Phi^T y - \theta\right\|_2^2$. If we notice that $(\Phi^T\Phi)^{-1}\Phi^T y$ is the least squares solution *i.e.,*

$$\theta_{\text{ls}} = \arg\min_{\theta} \|y - \Phi\theta\|_2^2 = (\Phi^T\Phi)^{-1}\Phi^T y, \tag{4.42}$$

the solution of (4.40) can be written as

$$\min_{\theta} \|\theta_{\text{ls}} - \theta\|_2^2 + \lambda\|\theta\|_1 = \min_{\theta} \sum_{i=1}^{n_\theta} \Big(\theta_{\text{ls}}(i) - \theta(i)\Big)^2 + \lambda|\theta(i)|. \tag{4.43}$$

We can now consider the estimate of each of the elements of θ separately. Let us consider $\theta(i)$. Taking the derivative w.r.t. $\theta(i)$ of

$$\Big(\theta_{\text{ls}}(i) - \theta(i)\Big)^2 + \lambda|\theta(i)| \tag{4.44}$$

gives

$$-2\Big(\theta_{\text{ls}}(i) - \theta(i)\Big) + \lambda \operatorname{sign}\Big(\theta(i)\Big), \quad \theta(i) \neq 0. \tag{4.45}$$

We have to handle $\theta(i) = 0$ separately. Setting the derivative equal to zero and solving gives

$$\theta(i) = \begin{cases} \theta_{\text{ls}}(i) - \lambda/2 & \text{if } \theta_{\text{ls}}(i) - \lambda/2 > 0 \\ \theta_{\text{ls}}(i) + \lambda/2 & \text{if } \theta_{\text{ls}}(i) + \lambda/2 < 0 \end{cases} \tag{4.46}$$

or

$$\theta(i) = \text{sign}\Big(\theta_{\text{ls}}(i)\Big)\Big(|\theta_{\text{ls}}(i)| - \lambda/2\Big). \tag{4.47}$$

For $|\theta_{\text{ls}}(i)| < \lambda/2$, $\theta(i) = 0$. The $\theta(i)$ minimizing (4.44) is hence

$$\theta(i) = \text{sign}\Big(\theta_{\text{ls}}(i)\Big)\min\Big(0, |\theta_{\text{ls}}(i)| - \lambda/2\Big). \tag{4.48}$$

Note that (4.48) holds for $i = 1, \ldots, n_\theta$. The relation, for this special case, between the least squares estimate θ_{ls} and the estimate from lasso is visualized in Figure 4.4. We see that lasso shrinks the least squares estimate and if the least squares parameter estimate is close enough to zero, lasso gives a parameter estimate identical to zero.

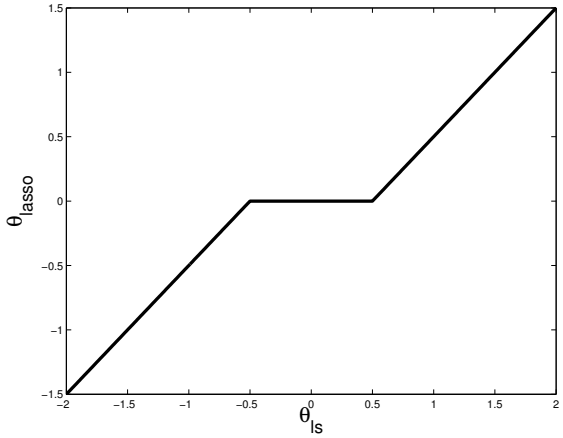

Figure 4.4: *The relation between the least squares estimate θ_{ls} and the estimate from lasso θ_{lasso} in the case where $\Phi^T\Phi = I$.*

4.3.2 Critical Parameter Value

Let us consider the ℓ_1-regularized least squares problem (4.20). A basic result from convex analysis tells us that there is a value $\lambda^{\max}$ for which the solution of the problem is equal to zero, if and only if $\lambda \geq \lambda^{\max}$. In other words, $\lambda^{\max}$ gives the threshold above which $\theta \equiv 0$. The critical parameter value $\lambda^{\max}$ is very useful in practice, since it gives a very good starting point in finding a suitable value of λ.

Proposition 4.1 (Critical Parameter Value $\lambda^{\max}$). *Let $\Phi \in \mathcal{R}^{N_e \times n}$ and $y \in \mathcal{R}^{N_e}$ be given. Let $\lambda^{\max}$ be such that θ minimizing*

$$\|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_1 \tag{4.49}$$

is zero if and only if $\lambda \geq \lambda^{\max}$. It holds that

$$\lambda^{\max} = \left\|2\Phi^T y\right\|_\infty. \tag{4.50}$$

The infinity-norm $\|\cdot\|_\infty$ is defined in Appendix A.

Proof: Define $\bar{e}_i$ as the n-dimensional row-vector with the ith element as one and the rest equal to zero. The subdifferential at $\theta = 0$ is readily computed to

$$\partial_{\theta(i)}\Big(\|y - \Phi\theta\|_2^2 + \lambda\|\theta\|_1\Big)\Big|_{\theta=0} = \Big[-2\bar{e}_i\Phi^T y - \lambda, -2\bar{e}_i\Phi^T y + \lambda\Big]. \tag{4.51}$$

For $\theta = 0$ to be an optima, it is necessary and sufficient (see *e.g.,* (Bertsekas et al., 2003, Prop. 4.7.2)) that

$$0 \in \Big[-2\bar{e}_i\Phi^T y - \lambda, -2\bar{e}_i\Phi^T y + \lambda\Big], \quad \forall i = 1, \ldots, n, \tag{4.52}$$

which is equivalent to

$$\lambda \geq \left\|2\Phi^T y\right\|_\infty. \tag{4.53}$$

(4.50) follows since $\lambda^{\max}$ is the smallest λ-value that makes $\theta = 0$ an optima. □

4.3.3 Sum-of-Norms Regularization

A ℓ_1-related regularization is the *sum-of-norms regularization*. A sum-of-norms regularized problem takes the form

$$\min_\theta V(\theta) + \lambda \sum_{i=1}^{s} \|\Gamma(i,:)\theta\|_p, \tag{4.54}$$

with $s \in \mathcal{N}$, Γ an $s \times dim(\varphi)\,(0,1)$-matrix and $\lambda \in \mathcal{R}^+$. The matrix Γ picks out groups of θ-elements. With $V(\theta) = \|y - \Phi\theta\|_2^2$ and $p = 2$ in (4.54),

$$\min_\theta \|y - \Phi\theta\|_2^2 + \lambda \sum_{i=1}^{s} \|\Gamma(i,:)\theta\|_2, \tag{4.55}$$

the formulation is often referred to as *group-lasso* (Yuan and Lin, 2006) in statistics. Note that the sum-of-norms regularization reduces to a ℓ_1-regularization if $\Gamma = I$ and $p = 1$ in (4.54).

We should comment on the difference between using an ℓ_1-regularization and some other type of sum-of-norms regularization, such as sum-of-Euclidean norms with $\Gamma \neq I$. When we use sum-of-norms regularization, the vector $\Gamma\theta$ will be sparse and when an element of the vector $\Gamma\theta$ is non-zero, say element i, then in general most of the θ-elements picked out by $\Gamma(i,:)$ are non-zero. The sum-of norms regularization hence makes sure that θ is sparse on a group-level, rather than an individual level.

Remark 4.3. Notice that (4.54) can be rewritten as

$$\min_{\theta} V(\theta) + \lambda \|\bar{\theta}\|_1, \quad \bar{\theta}(i) \triangleq \|\Gamma(i,:)\theta\|_p, \; i = 1, \ldots, s. \tag{4.56}$$

This clarifies the relation to the ℓ_1-regularization and provides an intuition for why groups of θ-elements ($\Gamma(i,:)\theta$, $i = 1, \ldots, s$) come out as zero or non-zero.

We continue the discussion on sum-of-norms regularization in Paper A, B, C and D.

4.3.4 Solution Methods

Many standard methods of convex optimization can be used to solve the problems (4.20) and (4.55). Software packages such as `CVX` (Grant and Boyd, 2010, 2008) or YALMIP (Löfberg, 2004) can readily handle the sum-of-norms regularization, by converting the problem to a cone problem and calling a standard interior-point cone solver. For the special case when the ℓ_1 norm is used as the regularization norm, more efficient special purpose algorithms and software can be used, such as `l1_ls` (Kim et al., 2007).

Recently many authors have developed fast first order methods for solving ℓ_1-regularized problems, and these methods can be extended to handle the sum-of-norms regularization, see *e.g.*, Roll (2008§2.2). Both interior-point and first-order methods have a complexity that scales linearly with N ($= dim(y)$ in (4.20)).

It has also been shown how solving ℓ_1-regularized problems can considerably be speeded up by pre-computing certain quantities (Mattingley and Boyd, 2010). It was shown how real-time performance can be met in many scenarios where ℓ_1-regularization previously was considered to be computationally too heavy.

CVX, YALMIP and l1_ls

CVX and YALMIP are very useful tools for solving ℓ_1 and sum-of-norms regularized (convex) problems. Both CVX and YALMIP are integrated with MATLAB. If we let

$$y = \begin{bmatrix} y_1 & y_2 & \ldots & y_{N_e} \end{bmatrix}^T, \; \Phi = \begin{bmatrix} \varphi_1 & \varphi_2 & \ldots & \varphi_{N_e} \end{bmatrix}^T, \; \Phi \in \mathcal{R}^{N_e \times n}, \; \lambda \in \mathcal{R}^+, \tag{4.57}$$

the ℓ_1-regularized least squares problem

$$\min_{\theta} \|y - \Phi\theta\|_2^2 + \lambda \|\theta\|_1 \tag{4.58}$$

can be solved using the CVX code given in Listing 4.1 and the YALMIP code given in Listing 4.2, assuming that the CVX respectively the YALMIP code package has been downloaded and installed. "y", "Phi", "n" and "lambda" also need to be available in the MATLAB workspace according to (4.57).

Listing 4.1: *CVX code for solving* (4.58)

```
cvx_begin
variable theta(n)
minimize((y-Phi*theta)'*(y-Phi*theta) ...
  +lambda*norm(theta,1))
cvx_end
```

Listing 4.2: *YALMIP code for solving* (4.58)

```
theta=sdpvar(n,1);
ops=sdpsettings('verbose',0);
solvesdp([],(y-Phi*theta)'*(y-Phi*theta) ...
   +lambda*norm(theta,1),ops)
```

A MATLAB package dedicated to ℓ_1-regularized least squares problems is l1_ls. With "y", "Phi" and "lambda" available in the MATLAB workspace according to (4.57) and the l1_ls package downloaded and installed, (4.58) can be solved as shown in Listings 4.3.

Listing 4.3: *l1_ls code for solving* (4.58)

```
rel_tol = 0.01; % relative target duality gap
theta=l1_ls(Phi,y,lambda,rel_tol)
```

4.4 Conclusion

This chapter has demonstrated how regularization can be used to obtain sparsity. There are a number of problems in system identification and signal processing that well fit into the framework developed. We therefore return to sparsity and regularization in Paper A, B, C and D.

5

Regularization for Smoothness

Regularization can be used to obtain meaningful results from ill-posed problems and to control for overfit. We care for both these applications in this thesis. However, we chose to focus on the type of regularization (referred to as a *standard regularization method* in Poggio et al. (1985)) obtained by adding a penalty term J to the criterion of fit,

$$\hat{\theta} = \arg\min_{\theta} \sum_{t \in \mathcal{N}_e} l\Big(y_t - f(\varphi_t, \theta)\Big) + \lambda J(\varphi_t, \theta), \quad \lambda \in \mathcal{R}^+. \tag{5.1}$$

The penalty J should be regarded as a means to introduce a *priori* knowledge and can be used to impose signal and model properties such as sparsity (discussed in Chapter 4) and smoothness. We discuss regularization for smoothness in this chapter. Geometrically, regularization for smoothness means that we seek the least rough function that gives a certain degree of fit to the observed data. Smoothness is in the regularization-literature used interchangeably with *curvature*, *non-rough*, *simplest* and *least complex*. The regularization parameter λ is used to control the trade-off between fit and smoothness.

Examples of regression methods that can be interpreted as regression methods that use regularization for smoothness are *support vector regression* and *Gaussian processes*. We give an introduction to these two methods in the following two sections.

5.1 Support Vector Regression

Let $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$, $y_t \in \mathcal{R}$, $\varphi_t \in \mathcal{R}^{n_\varphi}$, be a given estimation data set and let $\{h_k(\cdot) : \mathcal{R}^{n_\varphi} \to \mathcal{R},\ k = 1, \ldots, n\}$ be a set of basis functions. It could *e.g.,* be the n first basis

functions of a Fourier series expansion. Consider now the task of estimating the basis function coefficients $\theta_k \in \mathcal{R}$, $k = 1, \ldots, n$, in the basis function expansion model

$$f(\varphi, \theta) = \sum_{k=1}^{n} h_k(\varphi)\theta_k. \tag{5.2}$$

Assume that $n > N_e$. Seeking the model parameters that minimize the sum of squared residuals

$$\sum_{t=1}^{N_e} \Big(y_t - \sum_{k=1}^{n} h_k(\varphi_t)\theta_k\Big)^2 \tag{5.3}$$

leads to an ill-posed problem since $n > N_e$ (see Example 2.2). In particular, the solution will generally not be unique. We saw previously, see Example 2.2, how ℓ_2-regularization can be used to transform (5.3) into a well-posed problem. If we introduce

$$y \triangleq \begin{bmatrix} y_1 & \ldots & y_{N_e} \end{bmatrix}^T, \; \theta \triangleq \begin{bmatrix} \theta_1 & \ldots & \theta_n \end{bmatrix}^T, \; h(\varphi_t) \triangleq \begin{bmatrix} h_1(\varphi_t) & \ldots & h_n(\varphi_t) \end{bmatrix}^T \tag{5.4}$$

and the matrix $H \in \mathcal{R}^{N_e \times n}$

$$H \triangleq \begin{bmatrix} h(\varphi_1) & \ldots & h(\varphi_{N_e}) \end{bmatrix}^T, \tag{5.5}$$

the ℓ_2-regularized least-squares criterion can be written as

$$\min_{\theta} \|y - H\theta\|^2 + \lambda\|\theta\|^2, \quad \lambda \in \mathcal{R}^+. \tag{5.6}$$

The minimizing θ is then readily computed as (see *e.g.,* (2.27))

$$\hat{\theta} = (H^T H + \lambda I_n)^{-1} H^T y. \tag{5.7}$$

Let now φ_* be a given new regressor. The basis function model (5.2) evaluated at φ_* takes the form

$$f(\varphi_*, \hat{\theta}) = h(\varphi_*)^T \hat{\theta} = h(\varphi_*)^T \Big(H^T H + \lambda I_n\Big)^{-1} H^T y \tag{5.8}$$

or equivalently

$$f(\varphi_*, \hat{\theta}) = h(\varphi_*)^T H^T \Big(HH^T + \lambda I_{N_e}\Big)^{-1} y. \tag{5.9}$$

We could be satisfied and stop here. The sought basis function coefficients are provided by (5.7) and (5.9) gives a formula for the basis function expansion evaluated at a new regressor φ_*. Let us continue and consider what happens when n gets very large. It can then become computationally impossible to evaluate (5.8) and (5.9). To be able to handle large n, define $k(\varphi_i, \varphi_j) : \mathcal{R}^{n_\varphi \times n_\varphi} \to \mathcal{R}$ as

$$k(\varphi_i, \varphi_j) \triangleq h(\varphi_i)^T h(\varphi_j). \tag{5.10}$$

(5.9) can then be rewritten as

$$f(\varphi_*, \hat{\theta}) = k(\varphi_*, \Phi)\Big(k(\Phi, \Phi) + \lambda I_{N_e}\Big)^{-1} y \tag{5.11}$$

where

$$\Phi = \begin{bmatrix} \varphi_1 & \dots & \varphi_{N_e} \end{bmatrix}^T, \tag{5.12}$$

$$k(\varphi_*, \Phi) = \begin{bmatrix} k(\varphi_*, \varphi_1) & \dots & k(\varphi_*, \varphi_{N_e}) \end{bmatrix}, \tag{5.13}$$

$$k(\Phi, \Phi) = \begin{bmatrix} k(\varphi_1, \varphi_1) & k(\varphi_1, \varphi_2) & \dots & k(\varphi_1, \varphi_{N_e}) \\ k(\varphi_2, \varphi_1) & k(\varphi_2, \varphi_2) & & k(\varphi_2, \varphi_{N_e}) \\ \vdots & & \ddots & \vdots \\ k(\varphi_{N_e}, \varphi_1) & k(\varphi_{N_e}, \varphi_2) & \dots & k(\varphi_{N_e}, \varphi_{N_e}) \end{bmatrix}. \tag{5.14}$$

In this way, we have avoided the basis functions h_k, $k = 1, \dots, n$, but anyway found a way to evaluate the model (5.2). Also when n is infinite the solution is given by (5.11), as shown by the *representer theorem* (see *e.g.,* Kimeldorf and Wahba (1971)). This is useful! This means that we can replace the computation of an infinite number of basis function coefficients with $N_e^2 + N_e$ evaluations of $k(\cdot, \cdot)$. One may wonder when it is possible to rewrite the dot-product $h(\varphi_i)^T h(\varphi_j)$ as in (5.10). And also, when is it possible to rewrite a function $k(\varphi_i, \varphi_j)$ as a dot-product between basis functions? In fact, in practice the function $k(\varphi_i, \varphi_j)$ is chosen and the particular form of the basis functions often not derived or thought of. To guarantee that $k(\varphi_i, \varphi_j)$ can be written as a dot-product between basis functions, $k(\varphi_i, \varphi_j)$ should be chosen as a symmetric, positive semi-definite kernel (see Mercer's theorem *e.g.,* Evgeniou et al. (2000) or Schölkopf and Smola (2001, p. 37), see also Appendix A). The squared exponential kernel has these properties (see Appendix A for definition and examples of more kernels).

The kernel can here be seen as a way to redefine the dot-product in the regressor space. This trick of redefining the dot-product can be used in regression methods where regressors only enter as products. These types of methods are surprisingly many and the usage of this trick results in the *kernelized*, or simply kernel, version of the method. (5.11) is a special case of *Least Squares Support Vector Machines regression* (LS-SVM regression or LS-SVR, see *e.g.,* Saunders et al. (1998); Suykens and Vandewalle (1999)).

By kernelizing a regression method, the regressor space is transformed by h to a possibly infinite dimensional new space in which the regression takes place. The transformation of the regression problem to a new high-dimensional space is commonly referred to as the *kernel trick* (Boser et al., 1992).

Example 5.1: Illustration of the Kernel Trick

Let $\varphi_1 = \begin{bmatrix} \varphi_1(1) & \varphi_1(2) \end{bmatrix}^T$, $\varphi_2 = \begin{bmatrix} \varphi_2(1) & \varphi_2(2) \end{bmatrix}^T$ and $\varphi_* = \begin{bmatrix} \varphi_*(1) & \varphi_*(2) \end{bmatrix}^T$ be three regressors in $\mathcal{R}^2$. Observe that if we use

$$k(\varphi_1, \varphi_2) = \varphi_1^T \varphi_2 = \varphi_1(1)\varphi_2(1) + \varphi_1(2)\varphi_2(2) \tag{5.15}$$

in (5.11) we get exactly the same expression as in (2.27) *i.e.,* ridge regression. Let us now use the kernel (polynomial (inhomogeneous) kernel, see Appendix A)

$$\tilde{k}(\varphi_1, \varphi_2) = (1 + \varphi_1^T \varphi_2)^2. \tag{5.16}$$

This could also be thought of as changing the definition of the dot-product between two regression vectors. We see that the regressors now affect the regression algorithm through

$$\begin{aligned}\tilde{k}(\varphi_1, \varphi_2) =& (1 + \varphi_1^T \varphi_2)^2 && \text{(5.17a)}\\ =& 1 + 2\varphi_1(1)\varphi_2(1) + 2\varphi_1(2)\varphi_2(2) + \varphi_1(1)^2\varphi_2(1)^2 \\ &+ \varphi_1(2)^2\varphi_2(2)^2 + 2\varphi_1(1)\varphi_1(2)\varphi_2(1)\varphi_2(2). && \text{(5.17b)}\end{aligned}$$

We can rewrite this as the dot-product between the vector valued function $h(\cdot)$ evaluated at φ_1 and φ_2

$$\tilde{k}(\varphi_1, \varphi_2) = h(\varphi_1)^T h(\varphi_2) \tag{5.18}$$

with

$$h(\varphi_1) = \begin{bmatrix} 1 & \sqrt{2}\varphi_1(1) & \sqrt{2}\varphi_1(2) & \varphi_1(1)^2 & \varphi_1(2)^2 & \sqrt{2}\varphi_1(1)\varphi_1(2) \end{bmatrix}^T \tag{5.19}$$

and $h(\varphi_2)$ accordingly. The polynomial (inhomogeneous) kernel hence transform the regressor space into a 6-dimensional space. If we now assume that an estimation data set $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$ is given. Then in the particular case of LS-SVR, a linear model in $\mathcal{R}^6$ would be estimated to fit the transformed estimation data $\{(h(\varphi_t), y_t)\}_{t=1}^{N_e}$ using ridge regression. Reformulated in terms of the original regressors, the model evaluated at φ_* becomes

$$\begin{aligned}f(\varphi_*, \theta) =& \theta_1 + \sqrt{2}\theta_2\varphi_*(1) + \sqrt{2}\theta_3\varphi_*(2) + \theta_4\varphi_*(1)^2 + \theta_5\varphi_*(2)^2 \\ &+ \sqrt{2}\theta_6\varphi_*(1)\varphi_*(2). && \text{(5.20)}\end{aligned}$$

We see that by using this modified definition of the dot-product in LS-SVR we obtain a, in the regressors, polynomial predictor. We can hence compute nonlinear predictors by simply redefining the dot-product used in the regression algorithms.

We return to LS-SVR in Example 5.2.

5.2 Gaussian Process Regression

Consider the setup

$$y_t = f_0(\varphi_t) + e_t, \quad e_t \sim N(0, \sigma^2),\ \varphi_t \in \mathcal{R}^{n_\varphi},\ y_t \in \mathcal{R}. \tag{5.21}$$

Let $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$ be a given estimation data set and consider the task of finding an estimate for f_0 at a regressor φ_*. In *Gaussian Process Regression* (GPR, see *e.g.,* Rasmussen and Williams (2005), also called *Kriging*, see *e.g.,* Matheron (1973)) the output $f_0(\varphi)$ is assumed to be a *stochastic process*, a *Gaussian Process* (GP). Any samples taken from a (zero-mean) Gaussian process are by definition related by a (zero-mean) Gaussian probability distribution. In particular, $f_0(\varphi_i)$ and $f_0(\varphi_j)$

will be related by

$$\begin{bmatrix} f_0(\varphi_i) \\ f_0(\varphi_j) \end{bmatrix} \sim N\left(0_{2\times 1}, \begin{bmatrix} k(\varphi_i,\varphi_i) & k(\varphi_i,\varphi_j) \\ k(\varphi_j,\varphi_i) & k(\varphi_j,\varphi_j) \end{bmatrix}\right) \tag{5.22}$$

for some kernel k. Let now $\Phi \in \mathcal{R}^{N_\mathrm{e}\times n_\varphi}$ contain the estimation regressors

$$\Phi \triangleq \begin{bmatrix} \varphi_1 & \dots & \varphi_{N_\mathrm{e}} \end{bmatrix}^T, \tag{5.23}$$

φ_* be a new regressor and let $k(\varphi_*,\Phi)$ and $k(\Phi,\Phi)$ be as in (5.13) and (5.14). Then, using (5.22) we have that

$$\begin{bmatrix} f_0(\varphi_1) & f_0(\varphi_2) & \dots & f_0(\varphi_{N_\mathrm{e}}) & f_0(\varphi_*) \end{bmatrix}^T \sim N\left(0_{N_\mathrm{e}+1\times 1}, \begin{bmatrix} k(\Phi,\Phi) & k(\varphi_*,\Phi)^T \\ k(\varphi_*,\Phi) & k(\varphi_*,\varphi_*) \end{bmatrix}\right).$$

If we let y denote the estimation outputs, $y \triangleq \begin{bmatrix} y_1 & \dots & y_{N_\mathrm{e}} \end{bmatrix}^T$, then y and $f_0(\varphi_*)$ are related by

$$\begin{bmatrix} y^T & f_0(\varphi_*) \end{bmatrix}^T \sim N\left(0_{N_\mathrm{e}+1\times 1}, \begin{bmatrix} k(\Phi,\Phi)+\sigma^2 I_{N_\mathrm{e}} & k(\varphi_*,\Phi)^T \\ k(\varphi_*,\Phi) & k(\varphi_*,\varphi_*) \end{bmatrix}\right). \tag{5.24}$$

The predictive (or conditional) distribution for the stochastic variable $f_0(\varphi_*)$ given the estimation data can then be expressed as

$$\begin{aligned} p\Big(f_0(\varphi_*)\Big|\{(\varphi_t,y_t)\}_{t=1}^{N_\mathrm{e}}\Big) = N\Big(&k(\varphi_*,\Phi)(k(\Phi,\Phi)+\sigma^2 I_{N_\mathrm{e}})^{-1}y, \\ &k(\varphi_*,\varphi_*) - k(\varphi_*,\Phi)(k(\Phi,\Phi)+\sigma^2 I_{N_\mathrm{e}})^{-1}k(\varphi_*,\Phi)^T\Big) \end{aligned} \tag{5.25}$$

using identities for Gaussian distributions, see *e.g.,* (Rasmussen and Williams, 2005, p. 200). Notice that the (5.25) gives the distribution for the value of $f_0(\varphi_*)$ and not a measurement of f_0 at φ_*. To get the distribution for a measurement of f_0 at φ_*, σ^2 should be added to the covariance in (5.25). The kernel k defines the correlation between $f_0(\varphi_i)$ and $f_0(\varphi_j)$. This correlation is most often unknown and seen as a design choice in GPR. A popular choice is the squared exponential kernel, see Appendix A.

The predictive mean (mean of the distribution in (5.25)) takes exactly the same form as the prediction in least squares support vector regression, see (5.11). Gaussian process regression can hence also be given an interpretation as a regularization method.

Example 5.2: Gaussian Processes Regression (and LS-SVR)

Let $\{(\varphi_t,y_t)\}_{t=1}^{N_\mathrm{e}}$, $N_\mathrm{e}=10$, be generated by

$$y_t = 5\sin\varphi_t + e_t, \quad e_t \sim N(0,1),\ \varphi_t \sim U(0,5). \tag{5.26}$$

The estimation data are shown with '+'-marks in Figure 5.1. If Gaussian process regression with k as a scaled squared exponential kernel

$$k(\varphi_i,\varphi_j) = \gamma^2 e^{-\|\varphi_i-\varphi_j\|_2^2/2\ell^2}, \tag{5.27}$$

with a length scale $\ell = 1$, $\gamma = 5$ and noise standard deviations $\sigma = 0.1,\ 1,\ 5$ and 20 are used, the predictive distributions (for noisy measurements of f_0) visualized in Figure 5.1 are obtained.

The predictive mean (mean of the distribution in (5.25)) takes exactly the same form as the prediction in least squares support vector regression, see (5.11). Hence, the solid line in Figure 5.1 could equally well have been the result from LS-SVR with the kernel (5.27) and $\lambda = \sigma^2$. As seen in Figure 5.1, σ^2, or the regularization parameter, controls the smoothness of the predictive mean. If we let σ^2 go to infinity, the function-estimate will approach zero and a very smooth function. If we instead let σ^2 go to zero, the function estimate will become more and more non-smooth. This behavior is rather intuitive since σ^2 has an interpretation as the measurement noise covariance.

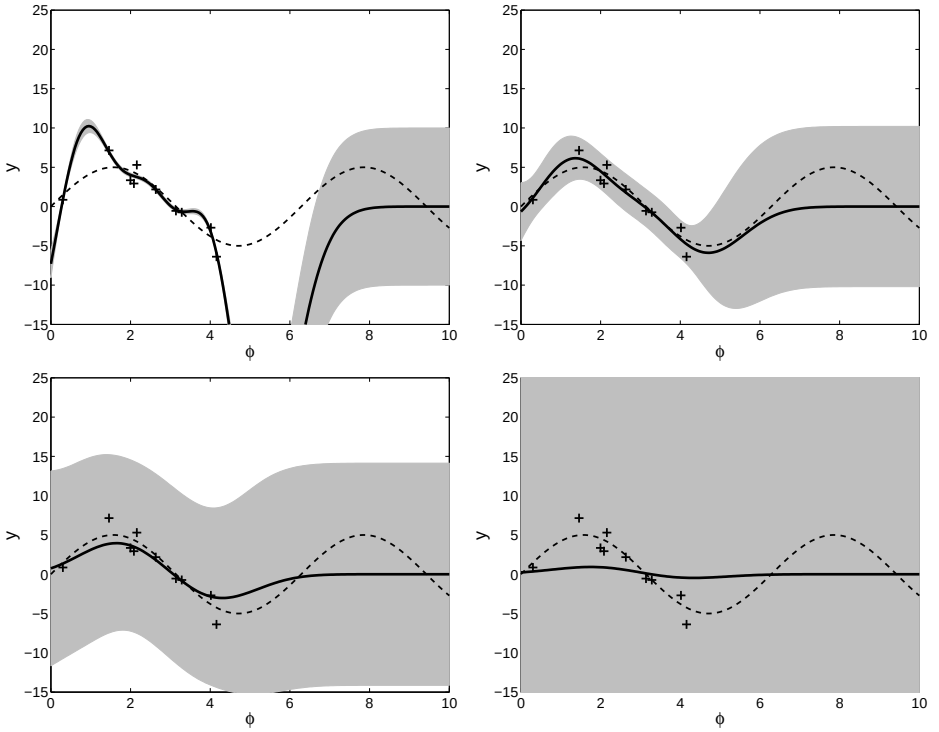

Figure 5.1: *Posterior (or predictive) distributions for a Gaussian process with $\ell = 1$, $\gamma = 5$ and $\sigma = 0.1$ (left top plot), 1 (right top plot), 5 (left bottom plot) and 20 (right bottom plot). The estimation data are shown with '+'-marks, the dashed line shows $5\sin(\cdot)$ and the solid line shows the mean of the predictive distribution or the LS-SVR estimate. The gray area shows the two standard deviations confidence interval for noisy measurements of f_0.*

Smoothness of the mean of the predictive distribution (5.25) is highly dependent on σ^2 (the regularization parameter). Parameters, such as σ and possible parameters of the kernel, that have to be set, are denoted *hyperparameters* (see Section 2.9 for hyperparmeters). The hyperparameters could be chosen using cross validation, but if few observations are available, maximizing the marginal likelihood is a good alternative (see Section 2.9).

Example 5.3: Gaussian Processes Regression Cont'd

Let us return to Example 5.2 and find the hyperparameters ℓ and γ of the scaled squared exponential kernel

$$k(\varphi_i, \varphi_j) = \gamma^2 e^{-\|\varphi_i - \varphi_j\|_2^2 / 2\ell^2} \tag{5.28}$$

and the measurement noise variance σ^2 by maximizing the marginal likelihood. The parameters were estimated to

$$\ell = 1.4,\ \gamma = 4.2,\ \sigma = 1.6, \tag{5.29}$$

using `GPML` (Rasmussen and Nickisch, 2010). `GPML` is a MATLAB toolbox for GPR. The code for estimating the hyperparameters using `GPML` are given in Listing 5.1.

Listing 5.1: *Estimation of hyperparameters ℓ, γ and σ using `GPML`.*

```
covfunc={'covSum',{'covSEard','covNoise'}};
loghyper=minimize([-1;-1;-1],'gpr',-100,covfunc,Phi,y);
[l gamma sigma]=exp(loghyper);
```

The resulting predictive distribution for noisy measurements of f_0 is visualized in Figure 5.2.

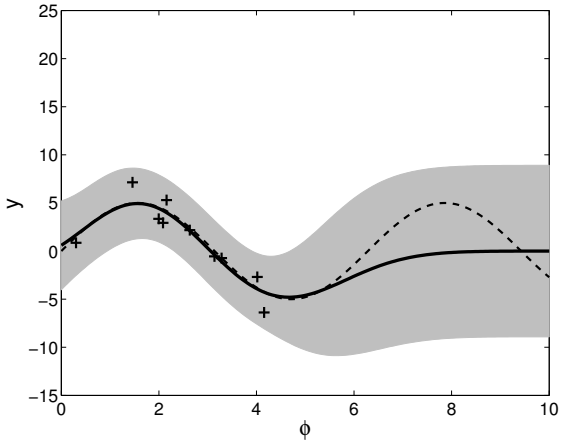

Figure 5.2: *The predictive distribution for noisy measurements of f_0. Mean given as a solid line and the gray area shows the two standard deviations confidence interval. The '+'-marks show the estimation data and the dashed line* $5\sin(\cdot)$.

5.3 Conclusion

Regularization for smoothness is essential in the estimation of many nonparametric models to obtain smooth estimates and control for overfitting. We have seen how both least squares support vector machines and Gaussian processes regression use regularization and how the regularization controlled the smoothness of the estimated model. We will continue the discussion on regularization and smoothness in Paper E and derive a novel regularization method, *Weight Determination by Manifold Regularization* (WDMR). Also Paper F discusses regularization for smoothness and in particular how it can be used to estimate impulse responses.

6

Concluding Remarks

6.1 Conclusion

The introductory part of this thesis was aimed to motivate and give a background to the papers of Part II. The focus was regularization and in particular, regularization for sparseness and smoothness. A number of examples of previous usages of regularization for sparseness and smoothness was given along with illustrative applications.

Part II of this thesis consists of a collection of papers. The first four papers utilize regularization for sparseness. First out is a novel optimization formulation for the identification of segmented ARX models, Paper A. Regularization for sparsity is there applied to control for overfitting. Paper B provides a novel system identification approach to piecewise affine systems. Regularization for sparsity is utilized to control for overfitting. Paper C discusses state estimation and provides a novel nonlinear smoother. The smoother works under the assumption that the process noise is impulsive, that is, often zero but occasionally nonzero. Regularization for sparsity again plays an important role to control for overfitting. The theory presented in this paper could be suitable in *e.g.,* target tracking applications. Paper D presents a novel model-based approach to trajectory generation. Regularization for sparsity is here used to find trajectories with compact representations. Paper E discusses regularization for smoothness. A novel regularization method *Weight Determination by Manifold Regularization* (WDMR) is presented. WDMR is inspired by manifold learning and applications in biology and has inherited properties thereof. WDMR uses regularization for smoothness to obtain smooth estimates. Paper F applies regularization for smoothness to linear system identification. In particular, high order FIR models are studied. Last, Paper G presents a real-time fMRI bio-feedback setup. The setup has served as a

proof of concept and shows that useful information can be read out, in real-time, from the brain activity measurements.

6.2 Future Research

It would be interesting to look at some more theoretical questions concerning the regularization methods and techniques developed in this thesis. A rather extensive theory has been developed around compressed sensing. This theory is not directly applicable to the methods presented in the papers of Part II on regularization for sparsity. It however provides tools for developing a deeper theoretical understanding. Interesting theoretical questions are:

- Under what assumptions can the correct sparsity pattern be found?
- How sensitive are the methods using regularization for sparsity for measurement noise? For example, how sensitive are the segmentation algorithm presented in Paper A to measurement noise?
- What happens if the number of estimation data samples goes to infinity? What is the asymptotic behavior?

There are also several possible application areas for regularization for sparseness which have not been explored. Multi-target tracking and event based sampling and control may for example be interesting areas for further research using regularization for sparseness.

It would also be interesting to investigate what techniques, such as, *General Principal Component Analysis* (GPCA, Vidal et al. (2003a,b, 2005)) can do for system identification and signal processing. GPCA has relations to sparsity techniques and has *e.g.,* been used in the identification of segmented ARX models, see *e.g.,* Vidal et al. (2003b). In particular, GPCA can be used to ensure that at least one element of a quantity is zero.

Interesting is also the development of new techniques and theories in machine learning. Many machine learning techniques are not directly applicable to dynamic systems, but they give a suitable foundation for the development of algorithms for dynamic systems. WDMR, presented in Paper E, is one example of such development. WDMR has shown useful in several applications, and there are for sure many interesting suitable applications as well as theoretical findings to be explored.

The last paper of this thesis, Paper G, discusses a real-time fMRI biofeedback setup. The potential of real-time fMRI is very exciting and applications of fMRI biofeedback have recently attract quite some attention in media and literature. It has *e.g.,* been shown how subjects can be trained to control their own pain using fMRI biofeedback (DeCharms et al., 2005). Our setup has been used as a communication interface (Eklund et al., 2010) and for real-time brain activity visualization (Nguyen et al., 2010). Many exciting applications remain to be explored, however.

6.3 Further Readings

For readers familiar with system identification that would like to read more about the mathematical background on underdetermined systems, sparseness and regularization, a very nice reading is Bruckstein et al. (2009). The paper by Zibulevsky and Elad (2010) also gives a nice introduction to sparsity. For a nice book that discusses several different regularization methods, Hastie et al. (2001) is to recommend. For the reader interested in machine learning and Bayesian modeling, Bishop (2006) is a good reference. Gaussian processes are nicely presented in Rasmussen and Williams (2005).

A

Kernels and Norms

This appendix lists a number of kernels and norms used in this thesis. Some properties of kernels are also discussed.

A.1 Kernels

In machine learning, a *kernel* $k : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$ is a general name for a function of two arguments mapping to $\mathcal{R}$. A kernel is said to be *symmetric* (see *e.g.,* Rasmussen and Williams (2005, p. 80)) if

$$k(\varphi_i, \varphi_j) = k(\varphi_j, \varphi_i), \tag{A.1}$$

for any two $\varphi_i, \varphi_j \in \mathcal{X}$. If the kernel is going to be used in GPR as a covariance function, it needs to be symmetric. A kernel is said to be *stationary* (see *e.g.,* Rasmussen and Williams (2005, p. 79)) if $k(\varphi_i, \varphi_j)$ can be written as

$$k(\varphi_i, \varphi_j) = \bar{k}(\varphi_i - \varphi_j), \quad \varphi_i, \varphi_j \in \mathcal{X}, \tag{A.2}$$

for some function $\bar{k} : \mathcal{X} \to \mathcal{R}$. It is *non-stationary* if not stationary. Last, a kernel is said to be *positive semi-definite* (see *e.g.,* Rasmussen and Williams (2005, p. 80)) if for any number of inputs $\varphi_1, \ldots, \varphi_N$ in $\mathcal{X}$, the *Gram matrix* K with element ij given by $k(\varphi_i, \varphi_j)$ is positive semi-definite.

A symmetric positive semi-definite kernel k can be written as a dot-product

$$k(\varphi_i, \varphi_j) = h^T(\varphi_i)h(\varphi_j), \quad \varphi_i, \varphi_j \in \mathcal{X}. \tag{A.3}$$

This follows from Mercer's theorem (see *e.g.,* Schölkopf and Smola (2001, pp. 37-38)). $h(\cdot)$ is called a *feature map*.

See Rasmussen and Williams (2005, Chap. 4) or Schölkopf and Smola (2001, Chap. 2) for further discussions on kernels and their properties.

Remark A.1. The precise mathematical definition of a kernel states that a kernel is a function $k : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$ that is both symmetric and positive semi-definite. We use the more liberal definition of machine learning.

A.1.1 Squared Exponential Kernel

For two vectors $\varphi_i, \varphi_j \in \mathcal{R}^n$, define the *squared exponential kernel* (sometimes called a *Gaussian kernel* or *Gaussian radial basis kernel*) as

$$k(\varphi_i, \varphi_j) \triangleq e^{-\|\varphi_i - \varphi_j\|_2^2 / 2\ell^2}, \tag{A.4}$$

where ℓ is a parameter of the kernel and denoted the *length scale*. The squared exponential kernel is symmetric, stationary and positive definite (Micchelli, 1986).

A.1.2 Polynomial Kernel

For two vectors $\varphi_i, \varphi_j \in \mathcal{R}^n$, define the *polynomial (inhomogeneous) kernel* as

$$k(\varphi_i, \varphi_j) \triangleq (\varphi_i^T \varphi_j + 1)^d, \quad d \in \mathcal{N}. \tag{A.5}$$

The *feature map*, or h, associated with the polynomial (inhomogeneous) kernel contains all monomials of order up to d (*e.g.,* Schölkopf et al. (2001, Prop. 2.17)). The polynomial kernel is symmetric, non-stationary and positive definite (see *e.g.,* Vapnik (1995, p. 460)).

A.2 Norms

A.2.1 Infinity Norm

For a vector $x \in \mathcal{R}^n$, define the infinity-norm as

$$\|x\|_\infty \triangleq \max_{i=1,\ldots,n} |x(i)|. \tag{A.6}$$

A.2.2 ℓ_0-Norm

For a vector $x \in \mathcal{R}^n$, define the zero (quasi-)norm as

$$\|x\|_0 \triangleq card\Big(\Big\{i\Big|x(i) \neq 0\Big\}\Big). \tag{A.7}$$

The zero norm is the number of nonzero elements of the vector x. The zero norm is a quasi-norm since it is not *positive homogeneous*. That is, the zero norm does not satisfy

$$\|\alpha x\|_0 \neq |\alpha| \|x\|_0, \quad \alpha \in \mathcal{R}, \tag{A.8}$$

which all norms should.

A.2.3 ℓ_p-Norm ($0 < p < \infty$)

For a vector $x \in \mathcal{R}^n$, define the ℓ_p-norm, $0 < p < \infty$, as

$$\|x\|_p \triangleq \left(\sum_{i=1}^{n} |x(i)|^p \right)^{1/p}. \tag{A.9}$$

The ℓ_2-norm is referred to as the *Euclidean norm*. See Figure 4.1, p. 52, for a visualization of some different ℓ_p-norms.

B

Huber Cost Function as a ℓ_1-Regularized Least Squares Problem

We use this appendix to show that the ℓ_1-regularized least squares formulation

$$\min_{\theta,\eta_1,\dots,\eta_N} \sum_{t=1}^{N_e} (y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda \left\| \begin{bmatrix} \eta_1 & \eta_2 & \dots & \eta_{Ne} \end{bmatrix} \right\|_1. \tag{B.1}$$

derived in Examples 4.3 and 4.6 is minimized by the same θ as

$$\min_{\theta} \sum_{t=1}^{N_e} \psi\big(y_t - \varphi_t^T\theta\big) \tag{B.2}$$

with

$$\psi(x) \triangleq \begin{cases} |x|^2, & \text{if } |x| < \lambda/2, \\ \lambda|x| - \lambda^2/4 & \text{otherwise.} \end{cases} \tag{B.3}$$

First notice that (B.1) is equivalent to

$$\min_{\theta,\eta_1,\dots,\eta_N} \sum_{t=1}^{N_e} \big((y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda|\eta_t|\big). \tag{B.4}$$

We now aim to show that

$$\min_{\eta_t} (y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda|\eta_t| = \psi(y_t - \varphi_t^T\theta). \tag{B.5}$$

Let us consider the left hand side of (B.5) and step-by-step derive the right hand side. First, notice that $|\eta_t| = \operatorname{sign}(\eta_t)\eta_t$ and

$$\frac{d}{d\eta_t}|\eta_t| = \frac{d}{d\eta_t}\operatorname{sign}(\eta_t)\eta_t = 2\delta(\eta_t)\eta_t + \operatorname{sign}(\eta_t), \tag{B.6}$$

the function $\delta(\cdot)$ denoting the Dirac delta function. Then

$$\frac{d}{d\eta_t}\left((y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda|\eta_t|\right) = -2(y_t - \varphi_t^T\theta - \eta_t) + 2\lambda\delta(\eta_t)\eta_t + \lambda\,\mathrm{sign}(\eta_t).$$

Equating the derivative to zero and solve for η_t gives

$$\eta_t^* = y_t - \varphi_t^T\theta - \lambda\delta(\eta_t^*)\eta_t^* - \lambda/2\,\mathrm{sign}(\eta_t^*), \tag{B.7}$$

which is implicit in η_t^*. For a $\eta_t^* > 0$, (B.7) reduces to

$$\eta_t^* = y_t - \varphi_t^T\theta - \lambda/2 \tag{B.8}$$

which implies that $y_t - \varphi_t^T\theta > \lambda/2$. Equivalent, a $\eta_t^* < 0$ implies that $\eta_t^* = y_t - \varphi_t^T\theta + \lambda/2$ and $y_t - \varphi_t^T\theta < -\lambda/2$. Now, if $\lambda/2 \geq y_t - \varphi_t^T\theta \geq 0$, then $\eta_t \geq 0$, since otherwise it dose not counteract on the positive $y_t - \varphi_t^T\theta$ in the left hand side of (B.5). Using this, the left hand side of (B.5) becomes

$$\min_{\eta_t:\eta_t\geq 0}(y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda\eta_t = \min_{\eta_t:\eta_t\geq 0}\eta_t(\eta_t + 2(\lambda/2 - (y_t - \varphi_t^T\theta))). \tag{B.9}$$

Since $\lambda/2 - (y_t - \varphi_t^T\theta) \geq 0$, $\eta_t^* = 0$ minimizes (B.9). Similarly, if $-\lambda/2 \leq y_t - \varphi_t^T\theta \leq 0$, then $\eta_t \leq 0$ which leads to

$$\min_{\eta_t:\eta_t\leq 0}(y_t - \varphi_t^T\theta - \eta_t)^2 - \lambda\eta_t = \min_{\eta_t:\eta_t\leq 0}\eta_t(\eta_t - 2(\lambda/2 + y_t - \varphi_t^T\theta)) \tag{B.10}$$

and again the same solution, $\eta_t^* = 0$. All together

$$\eta_t^* = \begin{cases} y_t - \varphi_t^T\theta - \lambda/2, & y_t - \varphi_t^T\theta > \lambda/2, \\ 0, & |y_t - \varphi_t^T\theta| < \lambda/2, \\ y_t - \varphi_t^T\theta + \lambda/2, & y_t - \varphi_t^T\theta < -\lambda/2. \end{cases} \tag{B.11}$$

(B.11) inserted in $(y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda|\eta_t|$ gives

$$\begin{aligned}
&\min_{\eta_t}(y_t - \varphi_t^T\theta - \eta_t)^2 + \lambda|\eta_t| && \text{(B.12a)}\\
&=(y_t - \varphi_t^T\theta - \eta_t^*)^2 + \lambda|\eta_t^*| && \text{(B.12b)}\\
&=\begin{cases} \lambda^2/4 + \lambda|y_t - \varphi_t^T\theta - \lambda/2|, & \text{if } y_t - \varphi_t^T\theta > \lambda/2 \\ (y_t - \varphi_t^T\theta)^2, & |y_t - \varphi_t^T\theta| < \lambda/2 \\ \lambda^2/4 + \lambda|y_t - \varphi_t^T\theta + \lambda/2|, & y_t - \varphi_t^T\theta < -\lambda/2 \end{cases} && \text{(B.12c)}\\
&=\begin{cases} \lambda(y_t - \varphi_t^T\theta) - \lambda^2/4, & \text{if } y_t - \varphi_t^T\theta > \lambda/2 \\ (y_t - \varphi_t^T\theta)^2, & |y_t - \varphi_t^T\theta| < \lambda/2 \\ -\lambda(y_t - \varphi_t^T\theta) - \lambda^2/4, & y_t - \varphi_t^T\theta < -\lambda/2 \end{cases} && \text{(B.12d)}\\
&=\psi\left(y_t - \varphi_t^T\theta\right) && \text{(B.12e)}
\end{aligned}$$

where the last equality holds from the definition (B.3) of the Huber loss function. Since (B.5) holds for any θ, it follows that θ minimizing (B.1) also minimizes (B.2).

Bibliography

H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Proceedings of the 2nd International Symposium on Information Theory*, pages 267–281, Akademiai Kiado, Budapest, 1973.

B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, N.J., 1979.

K. J. Åström and P. Eykhoff. System identification – A survey. *Automatica*, 7: 123–162, 1971.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. On climate reconstruction using bivalve shells: Three methods to interpret the chemical signature of a shell. In *Proceedings of the 7th IFAC Symposium on Modelling and Control in Biomedical Systems*, Aalborg, Denmark, August 2009a.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, J. Schoukens, and F. Dehairs. Three ways to do temperature reconstruction based on bivalve-proxy information. In *Proceedings of the 28th Benelux Meeting on Systems and Control*, Spa, Belgium, March 2009b.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. On climate reconstruction using bivalves: Three methods to interpret the chemical signature of a shell. *Computer Methods and Programs in Biomedicine*, 2010a. Accepted for publication.

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. A nonlinear multi-proxy model based on manifold learning to reconstruct water temperature from high resolution trace element profiles in biogenic carbonates. *Geoscientific Model Development*, 2010b. Accepted for publication.

T. Bayes. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

Y. Bengio, O. Delalleau, and N. Le Roux. The curse of highly variable functions for local kernel machines. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18 of *Neural Information Processing*. MIT Press, 2006.

D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

J. M. Bioucas-Dias. Bayesian wavelet-based image deconvolution: A GEM algorithm exploiting a class of heavy-tailed priors. *IEEE Transactions on Image Processing*, 15(4):937–951, April 2006.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

H. A. P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783, August 1988.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th annual workshop on Computational learning theory (COLT'92)*, pages 144–152, New York, NY, USA, 1992. ACM.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. P. Brady, M. N. Do, and R. Bhargava. Reconstructing FT-IR spectroscopic imaging data with a sparse prior. In *Proceedings of the 16th IEEE International Conference on Image Processing (ICIP)*, pages 829–832, November 2009.

K. Brandenburg. MP3 and AAC explained. In *Proceedings of the Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding*, Florence, Italy, September 1999.

A. M. Bruckstein, D. L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1): 34–81, 2009.

E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30, March 2008.

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, February 2006.

E. J. Candès, Y. C. Eldar, and D. Needell. Compressed sensing with coherent and redundant dictionaries. *CoRR*, abs/1005.2613, 2010.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

B. Chen, J. Paisley, and L. Carin. Sparse linear regression with beta process priors. In *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 1234–1237, Dallas, TX, March 2009.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

T. Chen, T. B. Schön, H. Ohlsson, and L. Ljung. Decentralization of particle filters using arbitrary state partitioning. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

T. Chen, T. B. Schön, H. Ohlsson, and L. Ljung. Decentralized particle filter with arbitrary state partitioning. *IEEE Transactions on Signal Processing*, 2010b. Accepted for publication.

T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes – Revisited. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

C. Daniel and F. S. Wood. *Fitting Equations to Data: Computer Analysis of Multifactor Data*. John Wiley & Sons, Inc., New York, NY, USA, 1980.

D. de Ridder and R. P.W. Duin. Locally linear embedding for classification, 2002. Technical Report, PH-2002-01, Pattern Recognition Group, Dept. of Imaging Science & Technology, Delft University of Technology, Delft, The Netherlands.

R. C. DeCharms, F. Maeda, G. H. Glover, D. Ludlow, J. M. Pauly, S. Whitfield, J. D. E. Gabrieli, and S. C. Mackey. Control over brain activation and pain learned by using real-time functional MRI. *Proc Natl Acad Sci USA*, 102:18626–18631, 2005.

M. P. Deisenroth and H. Ohlsson. General perspective to Gaussian filtering and smoothing: Explaining current and deriving new algorithms. In *Proceedings of the American Control Conference (ACC), 2011*, San Francisco, USA, 2011. Submitted.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences of the United States of America*, 100(10):5591–5596, 2003.

B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

A. Eklund, H. Ohlsson, M. Andersson, J. Rydell, A. Ynnerman, and H. Knutsson. Using real-time fMRI to control a dynamical system. In *Proceedings of the 17th Meeting of the International Society for Magnetic Resonance in Medicine (ISMRM)*, Honolulu, USA, April 2009a.

A. Eklund, H. Ohlsson, M. Andersson, J. Rydell, A. Ynnerman, and H. Knutsson. Using real-time fMRI to control a dynamical system by brain activity classification. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI'09)*, London, UK, September 2009b.

A. Eklund, M. Andersson, H. Ohlsson, A. Ynnerman, and H. Knutsson. A brain computer interface for communication using real-time fMRI. In *Proceedings of the International Conference on Pattern Recognition 2010*, Istanbul, Turkey, August 2010.

T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

T. Falck, H. Ohlsson, L. Ljung, J. A.K. Suykens, and B. De Moor. Segmentation of times series from nonlinear dynamical systems. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, August 2010.

Z. Guo, C. Li, L. Song, and Wang L. V. Compressed sensing in photoacoustic tomography in vivo. *Journal of Biomedical Optics*, 15(2), 2010.

F. Gustafsson. *Adaptive Filtering and Change Detection*. Wiley, New York, 2001.

F. Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur AB, 2010.

J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, 13:49–52, 1902.

T. Hastie, R Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

B. Hayes. The best bits. *American Scientist*, 97(4):276–280, 2009.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990. Cambridge.

J. Hu, J. Tian, and L. Yang. Functional feature embedded space mapping of fMRI data. In *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS'06)*, pages 1014–1017, 2006.

P. J. Huber. Robust regression: Asymptotics, conjectures and Monte Carlo. *The Annals of Statistics*, 1(5):799–821, 1973.

X. Huo and X. Ni. When do stepwise algorithms meet subset selection criteria? *Annals of Statistics*, 35(2):870–887, August 2007.

T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice-Hall, Englewood Cliffs, NJ, 2000.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, December 2007.

G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(82–95), 1971.

J. M. Lee. *Introduction to Topological Manifolds (Graduate Texts in Mathematics)*. Springer, May 2000.

F. Lindsten, J. Callmer, H. Ohlsson, D. Törnqvist, T. B. Schön, and F. Gustafsson. Geo-referencing for UAV navigation using environmental classification. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, Alaska, May 2010.

L. Ljung. *System Identification - Theory for the User*. Prentice-Hall, Upper Saddle River, N.J., 2nd edition, 1999.

L. Ljung. Prediction error estimation methods. *Circuits, Systems, and Signal Processing*, 21:11–21, 2002.

L. Ljung and T. Kailath. A unified approach to smoothing formulas. *Automatica*, 12(2):147–157, 1976.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL `http://control.ee.ethz.ch/~joloef/yalmip.php`.

S.G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, December 1993.

G. Matheron. The intrinsic random functions and their applications. *Advances in Applied Probability*, 5(3):439–468, 1973.

J. Mattingley and S. Boyd. Real-time convex optimization in signal processing. *IEEE Signal Processing Magazine*, 27(3):50–61, 2010.

C. A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995.

A. Neumaier. Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM Rev.*, 40(3):636–666, 1998.

K. Nguyen, A. Eklund, H. Ohlsson, F. Hernell, P. Ljung, C. Forsell, M. Andersson, H. Knutsson, and A. Ynnerman. Concurrent volume visualization of real-time fMRI. In *Proceedings of the IEEE International Symposium on Volume Graphics 2010*, Norrköping, Sweden, May 2010.

H. Ohlsson. *Regression on manifolds with implications for system identification*. Licentiate thesis no. 1382, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, December 2008.

H. Ohlsson and L. Ljung. Gray-box identification for high-dimensional manifold constrained regression. In *Proceedings of the 15th IFAC Symposium on System Identification, SYSID 2009*, Saint-Malo France, July 2009.

H. Ohlsson and L. Ljung. Semi-supervised regression and system identification. In X. Hu, U. Jonsson, B. Wahlberg, and B. Ghosh, editors, *Three Decades of Progress in Control Sciences*. Springer Verlag, December 2010a. To appear.

H. Ohlsson and L. Ljung. Weight determination by manifold regularization. In *Distributed Decision-Making and Control*, Lecture Notes in Control and Information Sciences. Springer Verlag, 2010b. Submitted.

H. Ohlsson and L. Ljung. Piecewise affine system identification using sum-of-norms regularization. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

H. Ohlsson, J. Roll, T. Glad, and L. Ljung. Using manifold learning for nonlinear system identification. In *Proceedings of the 7th IFAC Symposium on Nonlinear Control Systems (NOLCOS)*, Pretoria, South Africa, August 2007.

H. Ohlsson, J. Roll, A. Brun, H. Knutsson, M. Andersson, and L. Ljung. Direct weight optimization applied to discontinuous functions. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008a.

H. Ohlsson, J. Roll, and L. Ljung. Manifold-constrained regressors in system identification. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008b.

H. Ohlsson, J. Rydell, A. Brun, J. Roll, M. Andersson, A. Ynnerman, and H. Knutsson. Enabling bio-feedback using real-time fMRI. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008c.

H. Ohlsson, M. Bauwens, and L. Ljung. On manifolds, climate reconstruction and bivalve shells. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, December 2009.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State smoothing by sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010b. To appear.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State estimation under abrupt changes using sum-of-norms regularization. *Automatica*, 2010c. Submitted, under revision.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010d.

A. V. Oppenheim, A. S. Willsky, and S. H. Nawab. *Signals & Systems (2nd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.

G. Pillonetto and G. De Nicolao. A new kernel-based approach for linear system identification. *Automatica*, 46(1):81–93, January 2010.

T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317(26):314–319, 1985.

C. E. Rasmussen and H. Nickisch. GPML Gaussian processes for machine learning toolbox, 2010. Version 2.0, `http://www.gaussianprocess.org/gpml/code`.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, December 2005.

S. Riezler and A. Vasserman. Incremental feature selection and l1 regularization for relaxed maximum-entropy modeling. In D. Lin and D. Wu, editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 174–181, Barcelona, Spain, July 2004. Association for Computational Linguistics.

J. Roll. Piecewise linear solution paths with application to direct weight optimization. *Automatica*, 44:2745–2753, 2008.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the 15th International Conference on Machine Learning*, pages 515–521. Morgan Kaufmann, 1998.

B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory (COLT '01/EuroCOLT '01)*, pages 416–426, London, UK, 2001. Springer-Verlag.

X. Shen and F. G. Meyer. *Analysis of Event-Related fMRI Data Using Diffusion Maps*, volume 3565/2005 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, July 2005.

J.-L. Starck, E. J. Candes, and D. L. Donoho. The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, 11(6):670–684, June 2002.

J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

B. Thirion and O. Faugeras. Nonlinear dimension reduction of fMRI data: The Laplacian embedding approach. *IEEE International Symposium on Biomedical Imaging: Nano to Macro*, 1:372–375, 2004.

R. Tibsharani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B (Methodological)*, 58(1):267–288, 1996.

A.-I N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York, 1977.

J. A. Tropp. Just relax: Convex programming methods for identifying sparse signals in noise. *IEEE Transactions on Information Theory*, 52(3):1030–1051, March 2006.

J. A. Tropp, J. N. Laska, M. F. Duarte, J. K. Romberg, and R. G. Baraniuk. Beyond Nyquist: Efficient sampling of sparse bandlimited signals. *IEEE Transactions on Information Theory*, 56(1):520–544, January 2010.

J.A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, October 2004.

V. Vapnik. *Estimation of Dependences Based on Empirical Data (in Russian)*. Nauka, USSR, 1979.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*, pages 1063–1069, June 2003a.

R. Vidal, S. Soatto, Y. Ma, and S. Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, volume 1, pages 167–172, December 2003b.

R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(12):1945–1959, December 2005.

H. Wold. Estimation of principal components and related models by iterative least squares. In P.R. Krishnaiah, editor, *Multivariate Analysis*, pages 391–420. New York: Academic Press, 1966.

X. Yang, H. Fu, H. Zha, and J. Barlow. Semi-supervised nonlinear dimensionality reduction. In *Proceedings of the 23rd international conference on Machine learning (ICML '06)*, pages 1065–1072, New York, NY, USA, 2006. ACM.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

J. Zhang, S. Z. Li, and J. Wang. Manifold learning and applications in recognition. *Intelligent Multimedia Processing with Soft Computing*, 2004. Springer-Verlag, Heidelberg.

M. Zibulevsky and M. Elad. L1-L2 optimization in signal and image processing. *Signal Processing Magazine, IEEE*, 27(3):76–88, May 2010.

Part II

Publications

Paper A

Segmentation of ARX-Models Using Sum-of-Norms Regularization

Authors: Henrik Ohlsson, Lennart Ljung and Stephen Boyd

Edited version of the paper:

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010d.

Segmentation of ARX-Models Using Sum-of-Norms Regularization

Henrik Ohlsson*, Lennart Ljung* and Stephen Boyd**

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{ohlsson,ljung}@isy.liu.se

**Dept. of Electrical Engineering,
Stanford University,
Stanford, CA 94305 USA
boyd@stanford.edu

Abstract

Segmentation of time-varying systems and signals into models whose parameters are piecewise constant in time is an important and well studied problem. It is here formulated as a least-squares problem with sum-of-norms regularization over the state parameter jumps, a generalization of ℓ_1-regularization. A nice property of the suggested formulation is that it only has one tuning parameter, the regularization constant which is used to trade off fit and the number of segments.

1 Model Segmentation

Estimating linear regression models

$$y(t) = \varphi^T(t)\theta \tag{1}$$

is probably the most common task in system identification. It is well known how ARX-models

$$\begin{aligned} y(t) + a_1 y(t-1) + \cdots + a_n y(t-n) \\ = b_1 u(t-nk-1) + \cdots + b_m u(t-nk-m) \end{aligned} \tag{2}$$

with inputs u and outputs y can be cast in the form (1). Time-series AR models, without an input u are equally common.

The typical estimation method is least-squares,

$$\hat{\theta}(N) = \arg\min_{\theta} \sum_{t=1}^{N} \left\| y(t) - \varphi^T(t)\theta \right\|^2, \tag{3}$$

where $\|\cdot\|$ denotes the Euclidean or ℓ_2 norm.

A common case is that the system (model) is time-varying:

$$y(t) = \varphi^T(t)\theta(t). \tag{4}$$

A time-varying parameter estimate $\hat{\theta}$ can be provided by various tracking (online, recursive, adaptive) algorithms. A special situation is when the system parameters are piecewise constant, and change only at certain time instants t_k that are more or less rare:

$$\theta(t) = \theta_k, \quad t_k < t \leq t_{k+1}. \tag{5}$$

This is known as *model or signal segmentation* and is common in *e.g.,* signal analysis (like speech and seismic data), failure detection and diagnosis. There is of course a considerable literature around all this and its ramifications, *e.g.,* Ljung (1999), Gustafsson (2001), Basseville and Nikiforov (1993).

The segmentation problem is often addressed using multiple detection techniques, multiple models and/or Markov models with switching regression, see, *e.g.,* Lindgren (1978), Tugnait (1982), Bodenstein and Praetorius (1977). The function `segment` for the segmentation problem in the System Identification Toolbox (Ljung, 2007), is based on a multiple model technique (Andersson, 1985).

2 Our Method

We shall in this contribution study the segmentation problem from a slightly different perspective. If we allow all the parameter values in (4) to be free in a least-squares criterion we would get

$$\min_{\theta(t),\, t=1,\ldots,N} \sum_{t=1}^{N} \left\| y(t) - \varphi^T(t)\theta(t) \right\|^2. \tag{6}$$

Since the number of parameters then exceeds or equals the number of observations we would get a perfect fit, at the price of models that adjust in every time step, following any momentary noise influence. Such a grossly over-fit model would have no generalization ability, and so would not be very useful.

2.1 Sum-of-Norms Regularization

To penalize model parameter changes over time, we add a penalty or regularization term (see *e.g.,* Boyd and Vandenberghe (2004, p. 308)) that is a sum of norms of the parameter changes:

$$\min_{\theta(t),\, t=1,\ldots,N} \sum_{t=1}^{N} \left\| y(t) - \varphi^T(t)\theta(t) \right\|^2 + \lambda \sum_{t=2}^{N} \left\| \theta(t) - \theta(t-1) \right\|_{\text{reg}}, \tag{7}$$

where $\|\cdot\|_{\text{reg}}$ is the norm used for regularization, and λ is a positive constant that is used to control the trade-off between model fit (the first term) and time variation of the model parameters (the second term). The regularization norm $\|\cdot\|_{\text{reg}}$ could be any vector norm, like ℓ_1 or ℓ_p, but it is crucial that it is a sum of norms, and not a sum of squared norms, which is the more usual Tikhonov regularization.

When the regularization norm is taken to be the ℓ_1 norm, *i.e.,* $\|z\|_1 = \sum_{k=1}^{n} |z_k|$,

the regularization in (7) is standard ℓ_1 regularization of the least-squares criterion. Such regularization has been very popular recently, *e.g.,* in the much used lasso method (Tibsharani, 1996) or compressed sensing (Donoho, 2006; Candès et al., 2006). There are two key reasons why the parameter fitting problem (7) is attractive:

- It is a convex optimization problem, so the global solution can be computed efficiently. In fact, its special structure allows it to be solved in $O(N)$ operations, so it is quite practical to solve it for a range of values of λ, even for large values of N.
- The sum-of-norms form of the regularization favors solutions where "many" (depending on λ) of the regularized variables come out as exactly zero in the solution. In this case, this means estimated parameters that change infrequently (with the frequency of changes controlled roughly by λ).

We should comment on the difference between using an ℓ_1 regularization and some other type of sum-of-norms regularization, such as sum-of-Euclidean norms. With ℓ_1 regularization, we obtain a time-varying model in which individual elements of the $\theta(t)$ change infrequently. When we use sum-of-norms regularization, the whole vector $\theta(t)$ changes infrequently; but when it does change, typically all its elements change. In a statistical linear regression framework, sum-of-norms regularization is called group-lasso (Yuan and Lin, 2006), since it results in estimates in which many groups of variables (in this case, all elements of the parameter change $\theta(t) - \theta(t-1)$) are zero.

2.2 Regularization Path and Critical Parameter Value

The estimated parameter sequence $\theta(t)$ as a function of the regularization parameter λ is called the *regularization path* for the problem. Roughly, larger values of λ correspond to estimated $\theta(t)$ with worse fit, but fewer segments. A basic result from convex analysis tells us that there is a value $\lambda^{\max}$ for which the solution of the problem is constant, *i.e.,* $\theta(t)$ does not vary with t, if and only if $\lambda \geq \lambda^{\max}$. In other words, $\lambda^{\max}$ gives the threshold above which there is only one segment in $\theta(t)$. The critical parameter value $\lambda^{\max}$ is very useful in practice, since it gives a very good starting point in finding a suitable value of λ. Reasonable values are typically on the order of $0.01\lambda^{\max}$ to $\lambda^{\max}$ (which results in no segmentation).

Let θ^{const} be the optimal *constant* parameter vector, *i.e.,* the solution of the normal equations

$$\sum_{t=1}^{N} \left(y(t) - \varphi^T(t)\theta^{\text{const}}\right)\varphi^T(t) = 0. \tag{8}$$

Then we can express $\lambda^{\max}$ as

$$\lambda^{\max} = \max_{t=1,\ldots,N-1} \left\| \sum_{\tau=1}^{t} 2\left(y(\tau) - \varphi^T(\tau)\theta^{\text{const}}\right)\varphi^T(\tau) \right\|_{\text{reg}*}, \tag{9}$$

where $\|\cdot\|_{\text{reg}*}$ is the dual norm associated with $\|\cdot\|_{\text{reg}}$. This is readily computed.

To verify our formula for $\lambda^{\max}$ we use convex analysis (Rockafellar, 1996; Bertsekas et al., 2003; Borwein and Lewis, 2000). The constant parameter $\theta(t) = \theta^{\text{const}}$ solves the problem (7) if and only 0 is in its subdifferential. The fitting term is differentiable, with gradient w.r.t. $\theta(t)$ equal to

$$2\Big(y(t) - \varphi^T(t)\theta^{\text{const}}\Big)\varphi^T(t). \tag{10}$$

Now we work out the subdifferential of the regularization term. The subdifferential of $\|\cdot\|_{\text{reg}}$ at 0 is the unit ball in the dual norm $\|\cdot\|_{\text{reg}*}$. Therefore the subdifferential of the regularization term is any vector sequence of the form

$$g(1) = -z(2), \tag{11a}$$
$$g(2) = z(2) - z(3), \tag{11b}$$
$$\vdots$$
$$g(N-1) = z(N-1) - z(N), \tag{11c}$$
$$g(N) = -z(N), \tag{11d}$$

where $z(2), \ldots, z(N)$ satisfy $\|z(t)\|_{\text{reg}*} \leq \lambda$. We solve these to get

$$z(t) = -\sum_{\tau=1}^{t-1} g(\tau), \quad t = 2, \ldots, N. \tag{12}$$

The optimality condition is

$$g(t) + 2\Big(y(t) - \varphi^T(t)\theta^{\text{const}}\Big)\varphi^T(t) = 0, \quad t = 1, \ldots, N. \tag{13}$$

Combining this with the formula above yields our condition for optimality of $\theta(t) = \theta^{\text{const}}$ as $\lambda \geq \lambda^{\max}$.

2.3 Iterative Refinement

To (possibly) get even fewer changes in the parameter $\theta(t)$, with no or small increase in the fitting term, iterative re-weighting can be used (Candès et al., 2008). We replace the regularization term in (7) with

$$\lambda \sum_{t=2}^{N} w(t)\Big\|\theta(t) - \theta(t-1)\Big\|_{\text{reg}}, \tag{14}$$

where $w(2), \ldots, w(N)$ are positive weights used to vary the regularization over time. Iterative refinement proceeds as follows. We start with all weights equal to one, *i.e.*, $w^{(0)}(t) = 1$. Then for $i = 0, 1, \ldots$ we carry out the following iteration until convergence (which is typically in just a few steps).

1. *Find the parameter estimate.*
   Compute the optimal $\theta^{(i)}(t)$ with weighted regularization using weights $w^{(i)}$.
2. *Update the weights.*
   Set $w^{(i+1)}(t) = 1\Big/\Big(\epsilon + \Big\|\theta^{(i)}(t) - \theta^{(i)}(t-1)\Big\|_{\text{reg}}\Big)$.

Here ϵ is a positive parameter that sets the maximum weight that can occur.

One final step is also useful. From our final estimate of $\theta(t)$, we simply use the set of times at which a model change occurs (*i.e.,* for which $\theta(t) - \theta(t-1)$ is nonzero), and carry out a final least-squares fit over the parameters, which we now require to be piecewise constant over the fixed intervals. This typically gives a small improvement in fitting, for the same number of segments.

2.4 Solution Algorithms and Software

Many standard methods of convex optimization can be used to solve the problem (7) (code used by the authors can be found on `http://www.rt.isy.liu.se/~ohlsson/code.html`). Systems such as CVX (Grant and Boyd, 2010, 2008) or YALMIP (Löfberg, 2004) can readily handle the sum-of-norms regularization, by converting the problem to a cone problem and calling a standard interior-point cone solver. For the special case when the ℓ_1 norm is used as the regularization norm, more efficient special purpose algorithms and software can be used, such as `l1_ls` (Kim et al., 2007). Recently many authors have developed fast first order methods for solving ℓ_1 regularized problems, and these methods can be extended to handle the sum-of-norms regularization used here; see, for example, Roll (2008§2.2). Both interior-point and first-order methods have a complexity that scales linearly with N.

3 Numerical Illustration

We illustrate our method by applying it to a number of segmentation problems. We take $\epsilon = 0.01$ and use the Euclidean norm for regularization throughout the examples. The refinement technique described in Section 2.3 was applied with two refinement iterations and a final refinement by applying least-squares on segments without changes.

Example 1: Changing Time Delay

This example is from `iddemo11` in the System Identification Toolbox, (Ljung, 2007). Consider the system

$$y(t) + 0.9y(t-1) = u(t-n_k) + e(t). \tag{15}$$

The input u is a ± 1 PRBS (Pseudo-Random Binary Sequence) signal and the additive noise has variance 0.1. At time $t = 20$ the time delay n_k changes from 2 to 1. The data are shown in Figure 1. An ARX-model

$$y(t) + ay(t-1) = b_1 u(t-1) + b_2 u(t-2) \tag{16}$$

is used to estimate a, b_1, b_2 with the method described in the previous section. The resulting estimates using $\lambda = 0.1\lambda^{\max}$ are shown in Figure 2. The solid lines show the estimate and dashed the true parameter values. We clearly see that b_1 jumps from 0 to 1, to "take over" to be the leading term around sample 20. The estimate of the parameter a (correctly) does not change notably.

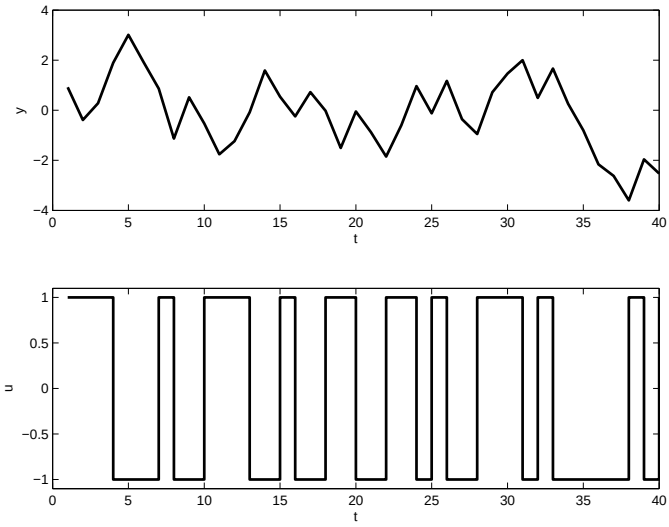

Figure 1: *The data used in Example 1.*

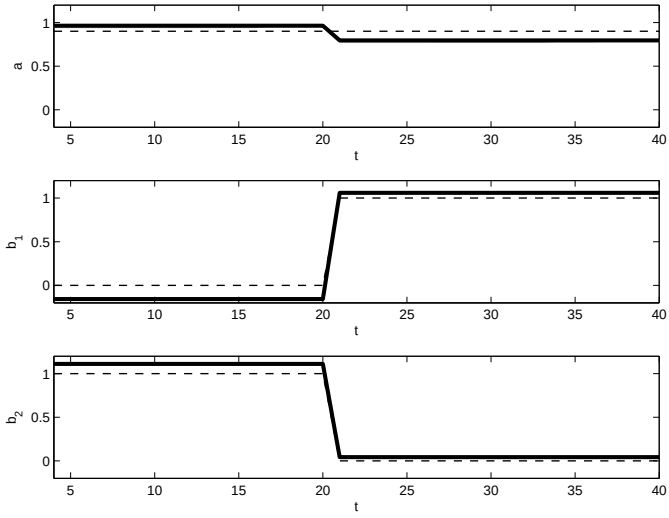

Figure 2: *The parameter estimates in Example 1. Solid lines show the parameter estimates and dashed lines the true parameter values.*

Example 2: Changing Time Series Dynamics

Consider the time series

$$y(t) + ay(t-1) + 0.7y(t-2) = e(t) \tag{17}$$

with $e(t) \sim N(0,1)$. At time $t = 100$ the value of a changes from -1.5 to -1.3. The output data and the estimate of a are shown in Figure 3. $\lambda = 0.01\lambda^{\max}$ was used.

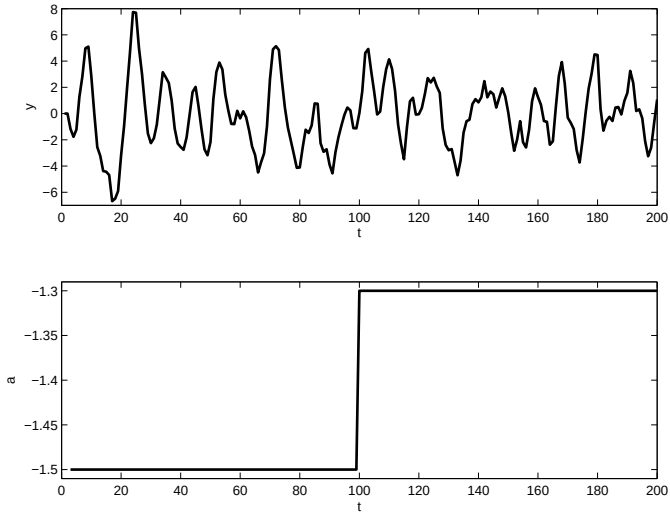

Figure 3: *The time series data (upper plot) and the estimate of a (lower plot) of Example 2.*

To motivate the iterative refinement procedure suggested in Section 2.3, let us see what happens if it is removed. Figure 4 shows the estimate of a (around $t = 100$) with and without the refinement iteration. As shown by the figure, (7) incorrectly estimates the change at $t = 100$ and gives an estimate having a change both at $t = 100$ and $t = 101$. Using iterative refinement, however, this does not occur. Without iterative refinement, a is estimated to -5.1 at $t = 100$.

Example 3: Seismic Signal Segmentation

Let us study the seismic data from the October 17, 1989, Loma Prieta earthquake in the Santa Cruz mountains. (This data is provided with MATLAB as `quake.mat` and discussed in the command `quake.m`). We choose to decimate the 200 Hz measurements of acceleration in the east-west direction ("`e`") by a factor of 100 and segment the resulting signal modeled as an AR process of second order. Here, the regularization constant λ in (7) will really act as a design parameter that controls how many segments will be chosen. For example, $\lambda = 0.15\lambda^{\max}$ gives two segments, $\lambda = 0.12\lambda^{\max}$ gives three segments and $\lambda = 0.1\lambda^{\max}$ gives four segments. The result for $\lambda = 0.15\lambda^{\max}$ is shown in Figure 5.

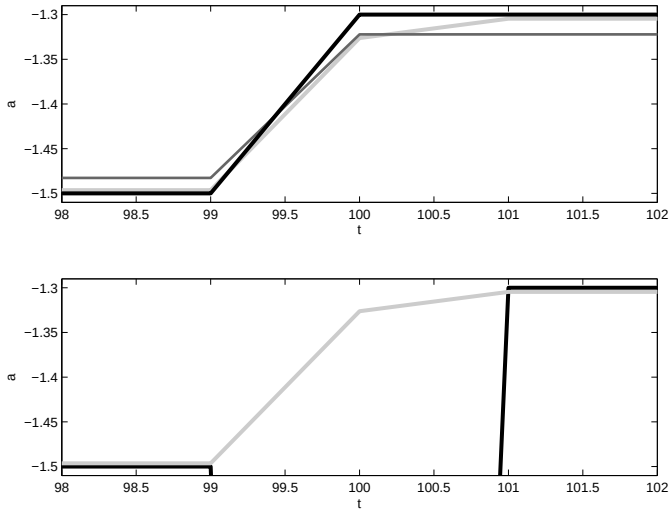

Figure 4: *Estimates of a in Example 2 with (top plot) and without (bottom plot) iterative refinement. Thick black line, estimate after least-squares has been applied to segments without changes in a and light-gray thick line, estimate given by* (7)*. In the top plot, the gray thin lines show estimates of a after one and two iterative refinements (the two lines are not distinguishable). Without iterative refinement (bottom plot) a is estimated to* -5.1 *at* $t = 100$*.*

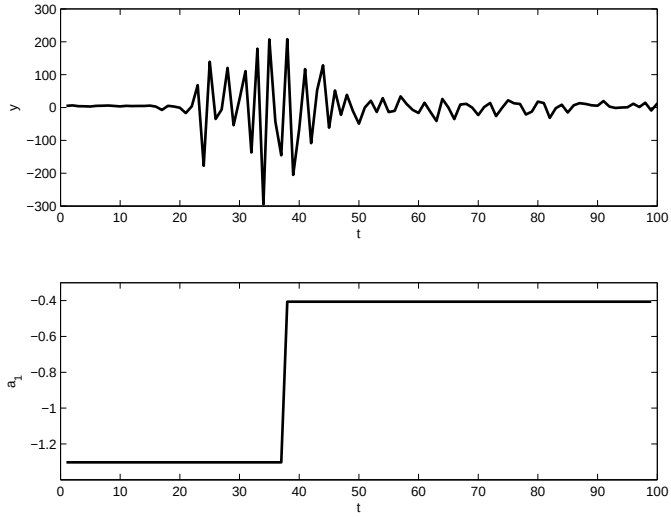

Figure 5: *The seismic signal used in Example 3 is shown in the upper plot. a_1 is shown in the lower plot.*

4 Comparisons with Other Methods for Segmentation

Several methods for model segmentation have been suggested earlier, see *e.g.*, Gustafsson (1992, Chap. 5), Gustafsson (2001), Basseville and Nikiforov (1993). They typically employ either multiple detection algorithms (Segen and Sanderson, 1980), hidden Markov models (HMM) (Blom and Bar-Shalom, 1988) or explicit management of multiple models, AFMM (Adaptive Forgetting by Multiple Models, Andersson (1985)). The latter algorithm is implemented as the method `segment` in the System Identification Toolbox and as the routine `detectM` in the software package `adfilt`, accompanying the book Gustafsson (2001). The idea is to let M Kalman filters for a stochastic system live in parallel. At each sample the M different predictions from the filters are evaluated. The worst performing filter is killed and a new filter is started. The segmentation is formed by the final estimate of each best performing filter. It should also be mentioned that a similar method to the one proposed in this paper has been discussed for set membership identification, and image segmentation, in Ozay et al. (2008).

All algorithms for tracking time-varying systems must have a trade-off between assumed noise level (e) and the tendency and size of system variations, and that may be reflected in the choice of several tuning parameters. In the `segment` algorithm, the user has to select 8 parameters (assumed noise variance R_2, probability of a jump, the process noise covariance matrix R_1, the initial parameter estimates, along with their covariance matrices, the guaranteed life length of each filter, and, if R_2 is estimated, the forgetting factor for estimating it). Even though several parameters can be given default values, it may be tedious work to tune the segmented regression algorithm. At the same time it leads to considerable flexibility. For good choices of these parameters, `segment` often gives performance comparable in quality to the algorithm suggested here. The big advantage of the proposed method is that it has only one scalar design parameter, λ, with the number of segments controlled by λ. Moreover, reasonable starting values of the parameter can be found from $\lambda^{\max}$, which is easily computed.

Most existing methods are local in nature: A jump is hypothesized at each time instant, and the ensuing samples are used to test this hypothesis. In contrast, our method is indeed global in nature: For a given λ (corresponding to a certain number of jumps), the positions of these jumps are determined as those that globally minimize (7). Still, the complexity of the algorithm is linear in the length of the data record. It seems that this should be an advantage for situations with infrequent jumps in noisy environments. That this indeed is the case is illustrated in the following example.

Example 4: Comparison Between `segment` and (7)

Let us compare our method with `segment` in the System Identification Toolbox (Ljung, 2007). Consider the system

$$y(t) + a_1 y(t-1) + 0.7y(t-2) = u(t-1) + 0.5u(t-2) + e(t) \tag{18}$$

with $u(t) \sim N(0,1)$ and $e(t) \sim N(0,9)$. At $t = 400$, a_1 changes from −1.5 to −1.3

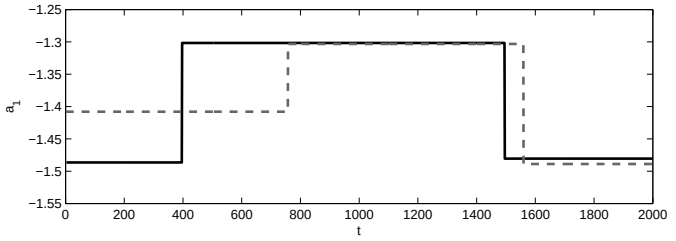

Figure 6: *Estimates of a_1 in the ARX-model used in Example 4 using our method (solid) and* `segment` *(dashed).*

and at $t = 1500$ a_1 returns to -1.5. Both `segment` and our method are provided with the correct ARX structure and asked to estimate all ARX parameters (a_1, a_2, b_1, b_2). With the same design parameters as used to generate the data (the true equation error variance, jump probability, initial ARX parameters and covariance matrix of the parameter jumps) `segment` does not find any changes at all in the ARX parameters. Tuning the design variable R_2 in `segment` so it finds three segments gives the estimate of a_1 shown in Figure 6. It does not seem possible to find values of all the design variables in `segment` that give the correct jump instants.

Using our method with the same choices as in Section 3 and tuning λ so as to obtain three segments gives directly the correct change times. The parameter estimate of our method using $\lambda = 0.025\lambda^{\max}$ is also shown in Figure 6.

5 Ramifications and Conclusions

5.1 Akaike's Criterion and Hypothesis Testing

Model segmentation is really a problem of selecting the number of parameters to describe the data. If the ARX model has n parameters and uses R segments, the segmented model uses $d = Rn$ parameters. The Akaike criterion (AIC), (Akaike, 1973) is a well known way to balance the model fit against the model complexity:

$$\min_{d,\Theta} \left[V(Z^N, \Theta) + \frac{2d}{N} \right] \tag{19}$$

$$d = \dim(\Theta) \tag{20}$$

where V is $2/N$ times the negative log likelihood function and Z^N is the data record with N observations. Comparing with (7), V is the log of 1/N times the first term (if the innovations are Gaussian, see *e.g.,* Ljung (1999, p. 506)), and the regularization term corresponds to the model cardinality term $2d/N$. In fact, sum-of-norms regularization is a common way to approximate cardinality constraints, *e.g.,* Boyd and Vandenberghe (2004). The link to cardinality penalties becomes even more pronounced with the iterative refinement procedure of

Section 2.3. It aims, with iterative replacement of the weights, at a regularization term

$$\lambda \sum_{t=2}^{N} \frac{\|\theta(t) - \theta(t-1)\|_{\text{reg}}}{\epsilon + \|\theta(t) - \theta(t-1)\|_{\text{reg}}}, \tag{21}$$

which essentially counts the number of nonzero terms, *i.e.,* the number of jumps and hence the number of parameters.

A common statistical approach to selecting model size is to use hypothesis testing, *e.g.,* Ljung (1999, p. 507), where the simpler model is the null hypothesis. Using the optimal test, likelihood ratios, is known to correspond to the Akaike criterion at a certain test level (Söderström, 1977). The criterion (7) can thus be interpreted as a simplified likelihood ratio test, where λ sets the test levels.

5.2 General State Space Models

It is well known that ARX-model estimation with varying parameters can be seen as state estimation in a general state space model, see *e.g.,* Ljung (1999, p. 367). Applying the Kalman filter to this time-varying ARX-model gives the recursive least squares algorithm. It works well if the time variation is well described as a Gaussian white noise process. The segmentation problem (5) rather correspond to an assumption that the parameter changes at rare instants, *i.e.,* a "process noise" that as zero most of the time, and nonzero at random time instants with a random amplitude. Our method can therefore also be used for state smoothing for general state space models with such process noise. This includes problems of abrupt change detection, and processes with load disturbances (*cf.* equations (2.10)-(2.11) in Ljung (1999).)

5.3 Summary

We have studied the model segmentation problem and suggested to treat it as least-squares problem with sum-of-norms regularization of the parameter changes. We do not claim that the suggested method necessarily outperforms existing approaches; but being a global method, it certainly has an edge in cases with considerable noise and infrequent jumps. An important benefit is also that it has just one scalar design variable, whose influence on the parameter fit and number of segments is easily understood, and for which a reasonable starting value is readily found.

Acknowledgement

This work was supported by the Strategic Research Center MOVIII, funded by the Swedish Foundation for Strategic Research, SSF, and CADICS, a Linnaeus center funded by the Swedish Research Council.

Bibliography

H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Proceedings of the 2nd International Symposium on Information Theory*, pages 267–281, Akademiai Kiado, Budapest, 1973.

P. Andersson. Adaptive forgetting in recursive identification through multiple models. *International Journal of Control*, 42(5):1175–1193, 1985.

M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes – Theory and Application*. Prentice-Hall, Englewood Cliffs, NJ, 1993.

D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

H. A. P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783, August 1988.

G. Bodenstein and H. M. Praetorius. Feature extraction from the electroencephalogram by adaptive segmentation. *Proceedings of the IEEE*, 65:642–652, 1977.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS Books in Mathematics. Springer, 2000.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, February 2006.

E. J. Candès, M. B. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications, special issue on sparsity*, 14(5):877–905, December 2008.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, August 2010.

F. Gustafsson. *Estimation of Discrete Parameters in Linear Systems*. PhD thesis, Linköping University, Linköping, Sweden, 1992. No 271.

F. Gustafsson. *Adaptive Filtering and Change Detection*. Wiley, New York, 2001.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, December 2007.

G. Lindgren. Markov regime models for mixed distributions and switching regressions. *Scandinavian Journal of Statistics*, 5:81–91, 1978.

L. Ljung. *System Identification - Theory for the User*. Prentice-Hall, Upper Saddle River, N.J., 2nd edition, 1999.

L. Ljung. *The System Identification Toolbox: The Manual*. The MathWorks Inc. 1st edition 1986, 7th edition 2007, Natick, MA, USA, 2007.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL `http://control.ee.ethz.ch/~joloef/yalmip.php`.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010.

N. Ozay, M. Sznaier, C. M. Lagoa, and O. Camps. A sparsification approach to set membership identification of a class of affine hybrid systems. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 123–130, December 2008.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1996.

J. Roll. Piecewise linear solution paths with application to direct weight optimization. *Automatica*, 44:2745–2753, 2008.

J. Segen and A. Sanderson. Detecting changes in a time-series. *IEEE Transactions on Information Theory*, 26:249–255, 1980.

T. Söderström. On model structure testing in system identification. *International Journal of Control*, 26:1–18, 1977.

R. Tibsharani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B (Methodological)*, 58(1):267–288, 1996.

J. K. Tugnait. Detection and estimation for abruptly changing systems. *Automatica*, 18:607–615, 1982.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

Paper B

Identification of Piecewise Affine Systems Using Sum-of-Norms Regularization

Authors: Henrik Ohlsson and Lennart Ljung

Edited version of the paper:

H. Ohlsson and L. Ljung. Piecewise affine system identification using sum-of-norms regularization. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

Identification of Piecewise Affine Systems Using Sum-of-Norms Regularization

Henrik Ohlsson and Lennart Ljung

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{ohlsson,ljung}@isy.liu.se

Abstract

Piecewise affine systems serve as an important approximation of nonlinear systems. The identification of piecewise affine systems is here tackled by overparametrizing and assigning a regressor-parameter to each of the observations. Regressor parameters are then forced to be the same if that not causes a major increase in the fit term. The formulation takes the shape of a least-squares problem with sum-of-norms regularization over regressor parameter differences, a generalization of the ℓ_1-regularization. The regularization constant is used to trade off fit and the number of partitions.

1 Introduction

Hybrid systems is a class of systems having both continuous and discrete dynamics. The continuous dynamics are often ruled by physical principles and the discrete due to discrete decisions or logic devices. But hybrid systems have also proven to be handy approximations of nonlinear continuous systems.

A type of hybrid systems is systems which can be described by a piecewise affine function, denoted *piecewise affine systems*. A *piecewise affine* (PWA) function $f : \mathcal{R}^{n_x} \to \mathcal{R}$ can be written on the form

$$f(x) = \begin{cases} \theta_1^T \begin{bmatrix} x \\ 1 \end{bmatrix}, & \text{if } x \in \mathcal{H}_1, \\ \vdots & \\ \theta_s^T \begin{bmatrix} x \\ 1 \end{bmatrix}, & \text{if } x \in \mathcal{H}_s, \end{cases} \tag{1}$$

where $\theta_k \in \mathcal{R}^{n_x+1}$, $k = 1, \ldots, s$, define the submodels on the *partitions* $\mathcal{H}_k$, $k = 1, \ldots, s$. The partitions are often assumed to be polyhedral. Measurements, or

observations, are noisy versions of $f(x)$ according to

$$y = f(x) + e, \quad E[e] = 0,\ E[ee^T] = \Gamma. \tag{2}$$

A subclass of PWA functions is *piecewise ARX* (PWARX). For a PWARX, x is composed of past system inputs u and outputs y.

1.1 Problem Formulation

Given the observations $\{(y_k, x_k)\}_{k=1}^N$, $y \in \mathcal{R}$, $x \in \mathcal{R}^{n_x}$, estimate a piecewise affine function of the form (1). The number of partitions, s, is a *priori* unknown. Estimation of the shape of the partitions is not treated in this contribution but can be handled by *e.g.,* applying a classification algorithm to the output of the proposed algorithm (see *e.g.,* Bemporad et al. (2005)).

1.2 Background

It is clear that if the partitions, *i.e.,* $\mathcal{H}_i$, $i = 1, \ldots, s$, are known, it is easy to find the regressor parameters of the subsystems. PWA system identification approaches can therefore be classified into groups according to how they find the partitions. Five techniques stand out:

- The parameters giving the partitions and the subsystem models are estimated simultaneously.
- Simple partitions and subsystem models are estimate simultaneously and repeatedly. See *e.g.,* Roll et al. (2004).
- The partitions and submodels are iteratively estimated, alternating between estimating partitions and submodels. See *e.g.,* Bemporad et al. (2003).
- The partitions are first estimated and then the submodels.
- The submodels are estimated and then the partitions (see *e.g.,* Vidal et al. (2003); Bemporad et al. (2005)).

The proposed method belongs to the last category. The underlying idea of methods of the last item is to simultaneously cluster the observed data and fit an affine model to the data of each cluster. It is essential that the clustering and regression are done simultaneously (or possibly alternating between the two) since the distance measure used in the clustering can not only be based on the distance between regressors. It must also depend on how well the measured output fit to the estimated submodels. Having clustered and estimated the submodels, the partitions are estimated.

In this contribution we pose the identification of piecewise affine systems as a sum-of-norms regularized least squares problem. The regularization constant is used to trade off fit and the number of partitions *i.e.,* s, and could preferably be found using cross validation (see *e.g.,* Hastie et al. (2001, pp. 214-217) for different types of cross validation). The proposed formulation takes the form of a convex optimization problem, so the global solution can be computed efficiently.

Relevant previous contributions using the sum-of-norms regularization are given by Kim et al. (2009); Ohlsson et al. (2010c,a,b). See also Ozay et al. (2008).

2 Proposed Method

2.1 Informal Preview

Assume that we are given a data set $Z^N = \{y_k, x_k,\ k = 1, \dots, N\}$ generated by a piecewise function (1), with s partitions $\mathcal{H}_k$, $k = 1, \dots, s$, and submodels defined by θ_k^0, $k = 1, \dots, s$.

1. In a first round we associate each measurement k with a parameter vector $\theta_k \in \mathcal{R}^{n_x+1}$. The goal of the proposed algorithm is to estimate θ_k, $k = 1, \dots, N$, so that if $x_k \in \mathcal{H}_r$ then $\theta_k = \theta_r^0$.
2. Next we cluster the x's into s subsets H_r, $r = 1, \dots, s$, that are suitable to associate with the same vector $\bar{\theta}_r$ *i.e.,* $H_r \triangleq \{x_k | \theta_k = \bar{\theta}_r\}$. This is done by checking which parameter vectors θ_k, θ_j that can be merged. Essentially we should check how much the criterion of fit
$$\min_{\theta_k, k=1,\dots,N} \sum_{k=1}^{N} \left\| \Gamma^{-1/2} \left(y_k - \theta_k^T \begin{bmatrix} x_k \\ 1 \end{bmatrix} \right) \right\|_2^2 \tag{3}$$
increases by setting $\theta_k = \theta_j = \bar{\theta}_r$ and merge if the increase is small enough. Then check if any of the $\bar{\theta}_r$s can be merged. And so on. This combinatorial problem is impractical to solve and we will here find an approximative solution using an optimization formulation. Note that we simultaneously estimate the submodels defined by $\bar{\theta}_r$, $r = 1, \dots, s$ and cluster the observation data (or the θ_ks).
3. The point sets H_r can now be used to partition the x-space into s partitions $\mathcal{H}_r$. This is a standard pattern recognition/classification problem that can be solved by several established technique (*e.g.,* support vector machines (Vapnik, 1995)) and will not be discussed here. See also Bemporad et al. (2005) for a discusses of this problem for a PWA system identification setting.

2.2 Clustering and Estimation Algorithm

We solve step (2) by the following technique: Let
$$K(x_k, x_j) : \mathcal{R}^{n_x} \times \mathcal{R}^{n_x} \to \mathcal{R} \tag{4}$$
be a kernel. We will give some examples of suitable choices of K shortly.

Given a data set Z^N, a choice of kernel K, p and λ, minimize
$$\sum_{k=1}^{N} \left\| \Gamma^{-1/2} \left(y_k - \theta_k^T \begin{bmatrix} x_k \\ 1 \end{bmatrix} \right) \right\|_2^2 + \lambda \sum_{k,j=1}^{N} K(x_k, x_j) \|\theta_k - \theta_j\|_p \tag{5}$$

with respect to θ_k, $k = 1, \ldots, N$, where Γ is defined in (2).

The first term of (5),

$$\sum_{k=1}^{N} \left\| \Gamma^{-1/2} \left(y_k - \theta_k^T \begin{bmatrix} x_k \\ 1 \end{bmatrix} \right) \right\|_2^2 \tag{6}$$

measures the fit to observations. The second term

$$\sum_{k=1}^{N} \sum_{j=1}^{N} K(x_k, x_j) \|\theta_k - \theta_j\|_p \tag{7}$$

is a regularization term, a *sum-of-norms* regularization. The sum-of-norms regularization has strong similarities to the ℓ_1-regularization, which has been very popular recently, *e.g.,* in the much used *lasso* method (Tibsharani, 1996) or *compressed sensing* (Donoho, 2006; Candès et al., 2006). In fact, if we define $\Delta\theta$ as the vector in $\mathcal{R}^{N^2}$ of stacked terms $K(x_k, x_j)\|\theta_k - \theta_j\|_p$, $k, j = 1 \ldots, N$,

$$\begin{aligned} \Delta\theta \triangleq \Big[& K(x_1, x_1)\|\theta_1 - \theta_1\|_p \;\; K(x_1, x_2)\|\theta_1 - \theta_2\|_p \;\ldots \\ & K(x_N, x_{N-1})\|\theta_N - \theta_{N-1}\|_p \;\; K(x_N, x_N)\|\theta_N - \theta_N\|_p \Big]^T, \end{aligned} \tag{8}$$

one could see (5) as a ℓ_1-regularized problem with the ℓ_1 regularization acting on the vector $\Delta\theta$. The ℓ_1-regularization makes sure that the vector $\Delta\theta$ becomes sparse. We will in general use $p = 2$, but other choices are of course possible. However, to get the properties discussed below, p should be chosen greater than one. We will come back to this shortly.

There are three key reasons why the criterion (5) is attractive:

- It is a convex optimization problem, so the global solution can be computed efficiently.
- The sum-of-norms-regularization will cause θ_k to be identical to θ_j, if that not causes a major increase in the fit term (6). In this case, this implies that many of the regularized variables come out as exactly zero. λ is a design parameter which regulates the number of clusters or partitions found.
- It is easy to include constraints without destroying convexity.

The kernel can be used to stress that θ's associated with closed-by x's are more probable to have identical θ-values. It can be seen as a prior for the clustering. We will use the following kernel in our examples:

$$K(x_k, x_j) \triangleq \begin{cases} 1 & \text{if } x_j \text{ is one of the 9 closest neighbors} \\ & \text{of } x_k \text{ among all the observations,} \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

Since the number of parameters in (5) equals the number of observations, the regularization is necessary to prevent overfitting to the noisy observations. Using (7)

we prevent overfitting by penalizing the number of distinct θ-values, essentially s, used in in (5).

Remark 1. Undesirable, also the cardinalities of H_r, $r = 1, \ldots, s$, play a role in the regularization (7). Our experience is that this effect is minor and that λ controls the trade-off between fit and the number of partitions s.

We should comment on the difference between using $p = 1$ and some $p > 1$ in (5). With $p = 1$, we obtain an estimate of the regularization variable having many of its elements equal to zero, we obtain a sparse vector. When we use $p > 1$, the whole estimated regularization variable vector often becomes zero; but when it is nonzero, typically all its elements are nonzero. $p > 1$ is clearly to be preferred here since we desire the whole parameter vectors θ to be the same if they are not needed to be different. In a statistical linear regression framework, sum-of-norms regularization ($p > 1$) is called *group-lasso* (Yuan and Lin, 2006), since it results in estimates in which many groups of variables are zero.

We can now define (with θ_k, $k = 1, \ldots, N$, minimizing (5)):

- s as the number of distinct θ-values in $\{\theta_k,\ k = 1, \ldots, N\}$.
- $\bar{\theta}_r$, $r = 1, \ldots, s$, to be the s distinct θ-values of $\{\theta_k,\ k = 1, \ldots, N\}$.
- H_r, $r = 1, \ldots, s$, as $H_r \triangleq \{x_k | \theta_k = \bar{\theta}_r\}$.
- $r(k)$ as the function

$$r(k) \triangleq r | k \in H_r. \tag{10}$$

2.3 Iterative Refinement

To (possibly) get even more zeros in the estimate of the regularized variables, with no or small increase in the fitting term, iterative re-weighting can be used (Candès et al., 2008). We modify the regularization term in (5) and consider

$$\sum_{k=1}^{N} \left\| \Gamma^{-1/2} \left(y_k - \theta_k^T \begin{bmatrix} x_k \\ 1 \end{bmatrix} \right) \right\|_2^2 + \lambda \sum_{k=1}^{N} \sum_{j=1}^{N} \alpha(k, j) K(x_k, x_j) \|\theta_k - \theta_j\|_p \tag{11}$$

where $\alpha(1, 1), \ldots, \alpha(N, N)$ are positive weights used to vary the regularization over indices k and j. Iterative refinement proceeds as follows. We start with all weights equal to one *i.e.,* $\alpha^{(0)}(k, j) = 1$, $k, j = 1, \ldots, N$. Then for $i = 0, 1, \ldots$ we carry out the following iteration until convergence (which is typically in just a few steps).

1. *Find the θ estimates.*
   Compute the optimal $\theta_k^{(i)}$ using (11) with the weighted regularization using weights $\alpha^{(i)}$.
2. *Update the weights.*
   For $j = 1, \ldots, N$, set $\alpha^{(i+1)}(k, j) = 1/(\epsilon + K(x_k, x_j)\|\theta_k - \theta_j\|_p)$.

Here ϵ is a positive parameter that sets the maximum weight that can occur.

One final step is also useful. From our final estimate of $\bar{\theta}$, we simply define the mapping $r(k)$ (see (10)) from the last iteration. Then carry out a constrained least squares optimization over $\bar{\theta}_r$

$$\min_{\bar{\theta}_r, r=1,\dots,s} \sum_{k=1}^{N} \left\| \Gamma^{-1/2} \left(y_k - \bar{\theta}_{r(k)}^T \begin{bmatrix} x_k \\ 1 \end{bmatrix} \right) \right\|_2^2. \tag{12}$$

The algorithm is summarized in Algorithm 1.

Algorithm 1 PWA System Identification Using Sum-of-Norms Regularization (PWASON)

Given $\{(y_t, x_t)\}_{t=1}^N$. Let ϵ be a positive parameter, set $\alpha^{(0)}(k,j) = 1$ for $k, j = 1, \dots, N$ and $i = 0$. Then, for a chosen kernel K, $p > 1$ and regularization parameter λ:

1. Compute the optimal $\theta_k^{(i)}$ using (11) with $\alpha = \alpha^{(i)}$.
2. Set $\alpha^{(i+1)}(k,j) = 1/(\epsilon + K(x_k, x_j)\|\theta_k - \theta_j\|_p)$.
3. If convergence, go to the next step, otherwise set $i = i + 1$ and return to (1).
4. Compute a final estimate of $\bar{\theta}_r$ using (12).

2.4 Solution Algorithms and Software

Many standard methods of convex optimization can be used to solve the problem (5). Systems such as CVX (Grant and Boyd, 2010, 2008) or YALMIP (Löfberg, 2004) can readily handle the sum-of-norms regularization, by converting the problem to a cone problem and calling a standard interior-point method. For the special case when the ℓ_1 norm is used as the regularization norm, more efficient special purpose algorithms and software can be used, such as `l1_ls` (Kim et al., 2007). Recently many authors have developed fast first order methods for solving ℓ_1 regularized problems, and these methods can be extended to handle the sum-of-norms regularization used here; see, for example, Roll (2008§2.2).

3 Numerical Illustrations

Example 1: A One Dimensional Example

Consider the one-dimensional PWARX system (introduced in Ferrari-Trecate et al. (2003))

$$y_k = \begin{cases} u_{k-1} + 2 + e_k, & -4 \le u_{k-1} \le -1, \\ -u_{k-1} + e_k, & -1 < u_{k-1} < 2, \\ u_{k-1} + 2 + e_k, & 2 \le u_{k-1} \le 4. \end{cases} \tag{13}$$

Generate $\{u_k\}_{k=1}^{50}$ by sample a uniform distribution $U(-4,4)$. Let $e_k \sim N(0, 0.05)$. Figure 1 shows the dataset $\{(y_k, u_k)\}_{k=1}^{50}$. Let the kernel K be defined by (9), set

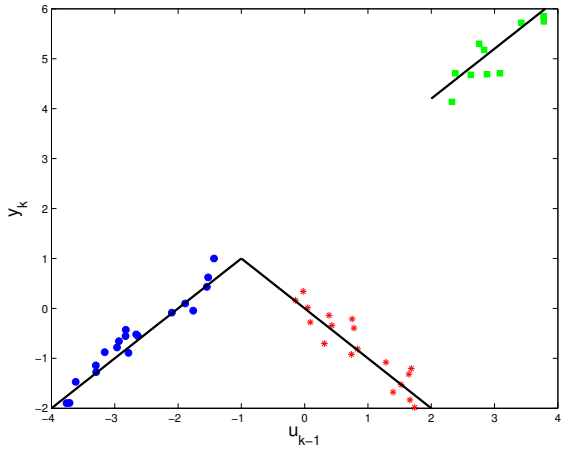

Figure 1: *Data used in Example 1. Solid line shows the true PWA function of the PWARX system.*

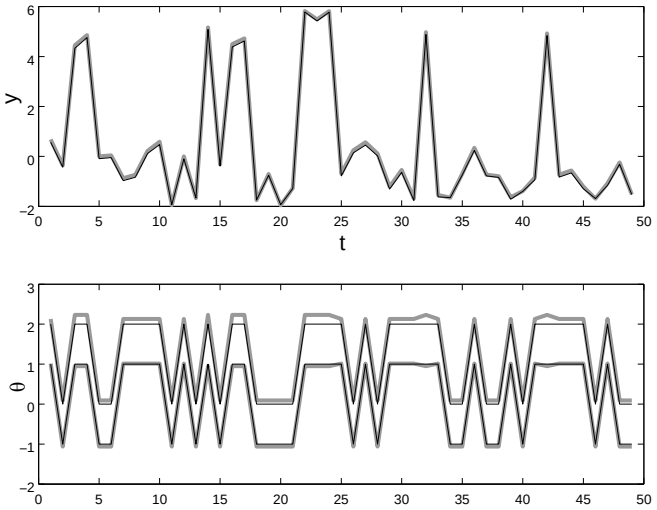

Figure 2: *Top plot, true (thin black line) and estimated (thick gray line) y (underneath the black line so hardly visible) for $k = 1, \ldots, 50$. Bottom plot, true (thin black line) and estimated (thick gray line) θ for $k = 1, \ldots, 50$.*

$x_k = u_{k-1}$, $\Gamma = 1$ and chose $p = 2$. $\lambda = 2$ then produces the result shown in Figure 2. The obtained $\bar{\theta}$-values were:

$$\begin{bmatrix} -1.0 \\ 0.1 \end{bmatrix}, \begin{bmatrix} 1.0 \\ 2.2 \end{bmatrix}, \begin{bmatrix} 1.0 \\ 2.1 \end{bmatrix}. \tag{14}$$

The results compare well with the result reported in Ferrari-Trecate et al. (2003).

Example 2: A Multi-Dimensional Example

Consider the multi-dimensional PWARX system (introduced in Bemporad et al. (2003), see also Nakada et al. (2005); Bemporad et al. (2005))

$$y_k = \begin{cases} -0.4y_{k-1} + u_{k-1} + 1.5 + e_k, & \text{if } 4y_{k-1} - u_{k-1} + 10 < 0 \\ 0.5y_{k-1} - u_{k-1} - 0.5 + e_k, & \text{if } 4y_{k-1} - u_{k-1} + 10 \geq 0 \text{ and} \\ & \quad 5y_{k-1} + u_{k-1} - 6 < 0 \\ -0.3y_{k-1} + 0.5u_{k-1} - 1.7 + e_k, & \text{if } 5y_{k-1} + u_{k-1} - 6 \geq 0. \end{cases} \tag{15}$$

Generate $\{u_k\}_{k=1}^{200}$ by sample a uniform distribution $U(-5, 5)$ and let $e_k \sim U(-0.1, 0.1)$. Figure 3, left plot, shows the dataset $\{(y_k, u_k)\}_{k=1}^{200}$. Define the kernel K as in (9), set $x_k = [y_{k-1}\ u_{k-1}]^T$, $\Gamma = 1$, $p = 2$ and $\lambda = 1$. The obtained $\bar{\theta}$-values were:

$$\begin{bmatrix} -0.40 \\ 1 \\ 1.50 \end{bmatrix}, \begin{bmatrix} 0.50 \\ -1 \\ -0.50 \end{bmatrix}, \begin{bmatrix} 0.57 \\ -1 \\ -0.50 \end{bmatrix}, \begin{bmatrix} -0.30 \\ 0.50 \\ -1.7 \end{bmatrix}, \begin{bmatrix} -1.60 \\ 1.92 \\ -4.7 \end{bmatrix}. \tag{16}$$

Most of the observations obtained a θ equal to one of the four first $\bar{\theta}$-values in (16). Three observations got a θ-estimate equal to the fifth $\bar{\theta}$-value. Increasing λ ($\lambda = 1.2$) causes the third θ-estimate to disappear and the observations previously associated with it to change to the second $\bar{\theta}$-value. The estimate for $\lambda = 1.2$ is visualized in the right of Figure 3, Figures 4, 5, 6 and 7. s is 4. Setting $\lambda = 1.5$ makes $s = 3$ and by that, all observations were correctly assigned to their partitions.

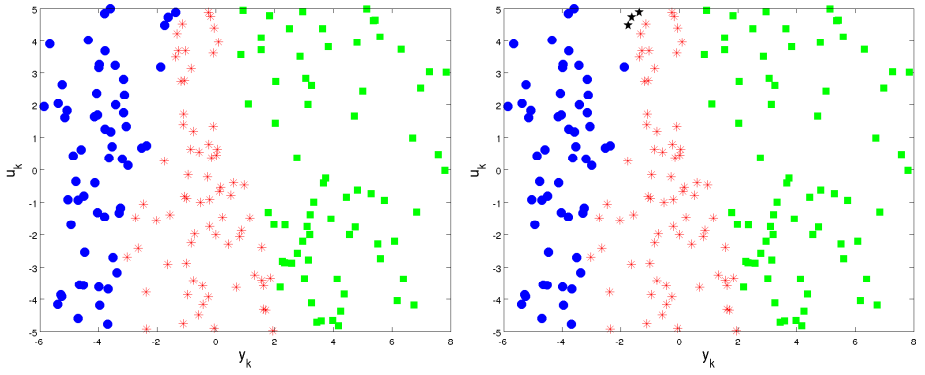

Figure 3: *Left plot, generated data ('o', '+' and '□'-symbols are used to show $\mathcal{H}_r$, $r = 1, \ldots, 3$). Right plot, estimated clusters ('o', '+', '□' and '★'-symbols are used to show H_r, $r = 1, \ldots, 4$).*

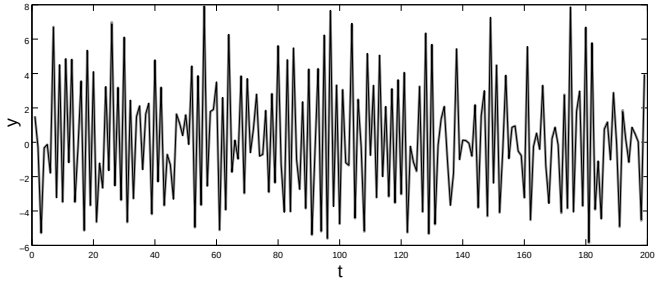

Figure 4: *Noise-free y (black thin) and estimated y (thick gray line) for $k = 1, \ldots, 200$. The black line is on top of the gray line. See Figure 6 for the difference between noise-free and estimated y.*

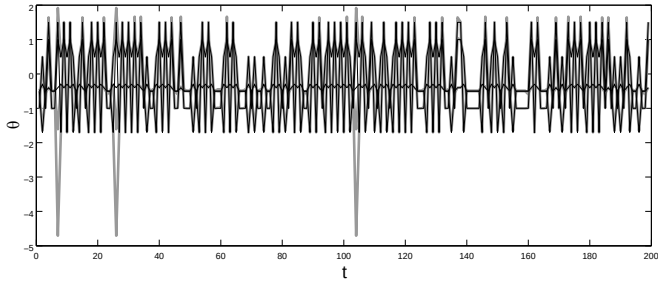

Figure 5: *True θ (black thin) and estimated θ (thick gray line) for $k = 1, \ldots, 200$.*

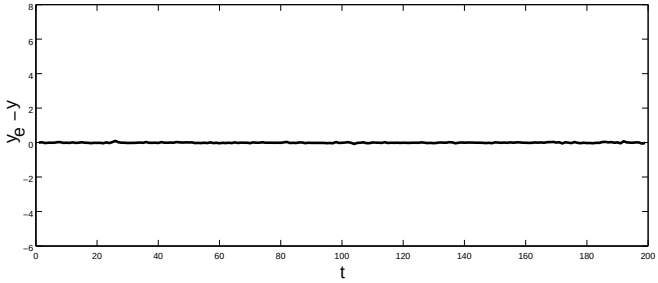

Figure 6: *Difference between noise-free y and estimated y for $k = 1, \ldots, 200$.*

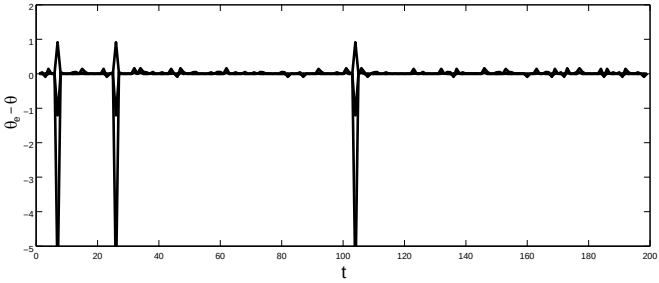

Figure 7: *Difference between true θ and estimated θ for $k = 1, \ldots, 200$.*

Example 3: Approximation of a Nonlinear Function

Consider

$$y_t = f(u_t) + e_t, \qquad f(u_t) = e^{-u_t},\ e_t \sim N(0, 0.001). \tag{17}$$

Generate 100 observations by letting $u \sim U(0, 5)$. The observations are shown in Figure 8. Let us use the proposed method to generate a piecewise affine approximation of $f(u_t) = e^{-u_t}$. λ here controls the trade-off between the fit and the number of segments. $\lambda = 0.01$ gives the result given in the left of Figure 9 and $\lambda = 0.05$ gives the result given in the right of Figure 9. In both cases, the kernel defined by (9), $\Gamma = 1$ and $p = 2$ were used.

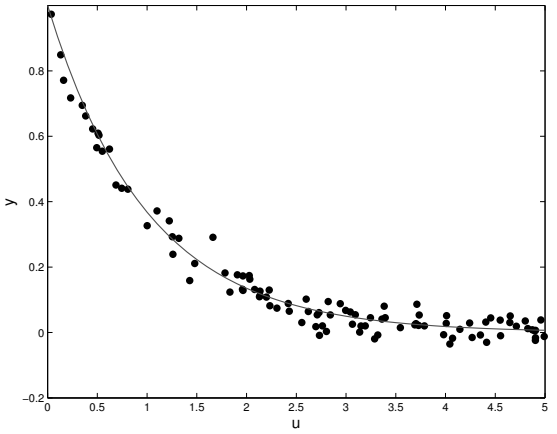

Figure 8: *Observed y's and f (thin gray line).*

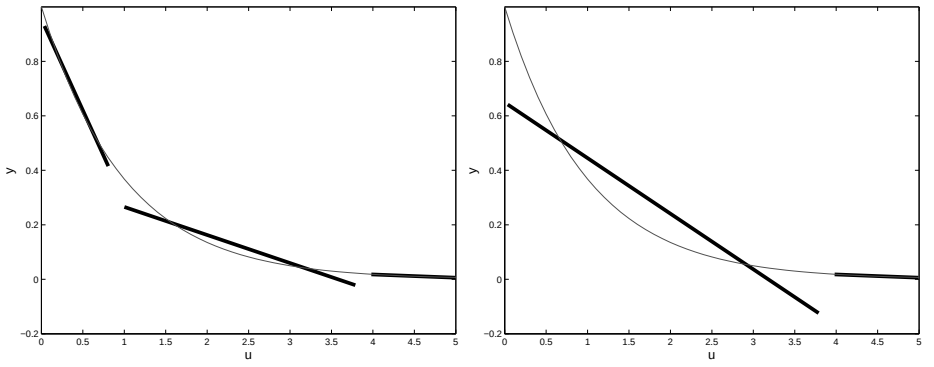

Figure 9: *Approximated f (thick black line) and f (thin gray line). In the left plot $\lambda = 0.01$ and in the right plot $\lambda = 0.05$.*

4 Conclusion

A method for piecewise affine system identification has been presented. The formulation takes the shape of a least-squares problem with sum-of-norms regularization over regressor parameter differences, a generalization of ℓ_1-regularization. The regularization constant is used to trade off fit and the number of partitions. Numerical illustrations on previously known examples from the literature shows that the proposed method performs well in comparison to know piecewise affine systems identification methods.

There are several interesting extensions of proposed scheme. For example, a piecewise nonlinear function could be estimated by applying a regularization as in (7) to *Support Vector Regression* (SVR, see *e.g.,* Suykens and Vandewalle (1999)).

Acknowledgment

Partially supported by the Swedish foundation for strategic research in the center MOVIII and by the Swedish Research Council in the Linnaeus center CADICS.

Bibliography

A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino. A greedy approach to identification of piecewise affine models. In *Proceedings of the 6th international conference on Hybrid systems (HSCC'03)*, pages 97–112, Prague, Czech Republic, 2003. Springer-Verlag.

A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino. A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10):1567–1580, October 2005.

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, February 2006.

E. J. Candès, M. B. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications, special issue on sparsity*, 14(5):877–905, December 2008.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

G. Ferrari-Trecate, M. Muselli, D Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, August 2010.

T. Hastie, R Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, December 2007.

S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL `http://control.ee.ethz.ch/~joloef/yalmip.php`.

H. Nakada, K. Takaba, and T. Katayama. Identification of piecewise affine systems based on statistical clustering technique. *Automatica*, 41(5):905–913, 2005.

H. Ohlsson and L. Ljung. Piecewise affine system identification using sum-of-norms regularization. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State smoothing by sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010b. To appear.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010c.

N. Ozay, M. Sznaier, C. M. Lagoa, and O. Camps. A sparsification approach to set membership identification of a class of affine hybrid systems. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 123–130, December 2008.

J. Roll. Piecewise linear solution paths with application to direct weight optimization. *Automatica*, 44:2745–2753, 2008.

J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004.

J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999.

R. Tibsharani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B (Methodological)*, 58(1):267–288, 1996.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

R. Vidal, S. Soatto, Y. Ma, and S. Sastry. An algebraic geometric approach to the identification of a class of linear hybrid systems. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, volume 1, pages 167–172, December 2003.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

Paper C

Smoothed State Estimates Under Abrupt Changes Using Sum-of-Norms Regularization

Authors: Henrik Ohlsson, Fredrik Gustafsson, Lennart Ljung and Stephen Boyd

Edited version of the paper:

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State estimation under abrupt changes using sum-of-norms regularization. *Automatica*, 2010c. Submitted, under revision.

Parts of the theory presented in this paper have also been presented in:

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State smoothing by sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

Smoothed State Estimates Under Abrupt Changes Using Sum-of-Norms Regularization

Henrik Ohlsson*, Fredrik Gustafsson*, Lennart Ljung* and Stephen Boyd**

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{ohlsson,fredrik,
ljung}@isy.liu.se

**Dept. of Electrical Engineering,
Stanford University,
Stanford, CA 94305 USA
boyd@stanford.edu

Abstract

The presence of abrupt changes, such as impulsive disturbances and load disturbances, make state estimation considerably more difficult than the standard setting with Gaussian process noise. Nevertheless, this type of disturbances is commonly occurring in applications which makes it an important problem. An abrupt change often introduces a jump in the state and the problem is therefore readily treated by change detection techniques. In this paper, we take a rather different approach. The state smoothing problem for linear state space models is here formulated as a least-squares problem with sum-of-norms regularization, a generalization of the ℓ_1-regularization. A nice property of the suggested formulation is that it only has one tuning parameter, the regularization constant which is used to trade off fit and the number of jumps. An extension to nonlinear state space models is also given.

1 Introduction

We consider the problem of state estimation in linear state space models, where impulsive disturbances occur in the process model. There are several conceptually different ways to handle disturbances in state estimation. One possibility is to model the disturbance as a sequence of Gaussian random variables with known second order moment, so the optimal solution is provided by the Kalman filter (KF, Kalman (1960), see also Kailath et al. (2000)). Another, quite different, possibility is to assume that the disturbance is a deterministic arbitrary sequence, and apply subspace projections. See Hui and Zak (2005) for details and more references on that approch. The case of impulsive process noise occurs frequently in at least three different application areas:

- In automatic control, impulsive noise is often used to model load disturbances.
- In target tracking, impulsive noise is used to model force disturbances, corresponding to maneuvers for the tracked object.
- In fault detection and isolation (FDI) literature impulsive noise is used to model additive faults. Usually, this is done in a deterministic setting (Patton et al., 1989), but a stochastic framework is also common (Basseville and Nikiforov, 1993; Gustafsson, 2001).

We formulate the problem in a probabilistic framework where the KF is the best linear unbiased estimator (BLUE), and the interacting multiple model (IMM, Blom and Bar-Shalom (1988)) algorithm provides an approximation to the exact problem. In contrast to IMM, we here propose a method that solves an approximate problem in an optimal way.

Our approach is based on convex optimization. It is well-known that the KF solves an optimization problem where the sum of squared two-norms of the process and measurement noises is minimized. Inspired by the recent progress of sum-of-norms regularization in the statistical literature (Kim et al. (2009), see also related contribution in the control community, Ohlsson et al. (2010c)), we suggest to change the squared two-norm of the process noise to a sum-of-norms, to capture the impulse character of the process disturbance. The consequence of this is that a sparse sequence of process noise is automatically obtained in contrast to the KF. The algorithm solves the smoothing problem in linear complexity, and a further advantage compared to KF and IMM algorithms is that convex constraints of the state sequence are easily handled in the same framework.

We start with a brief introduction to dynamical systems and stochastic disturbances. This is followed up by a discussion on the smoothing problem in Section 3. In particular, we care about the optimization formulation of the Kalman smoother. Section 4 contains the main contribution of the paper, the proposed method for state smoothing with impulsive process disturbances. We call the method state smoothing by sum-of-norms regularization (STATESON). In Section 5 a comparison with popular methods for state smoothing with impulsive process noise is given. A justification by some numerical illustrations is given in Section 6. Section 7 presents an extension of the presented framework to nonlinear models. The paper is ended by a conclusion in Section 8.

2 Introduction: Dynamic Systems with Stochastic Disturbances

The standard linear state space model with stochastic disturbances is well known to be

$$\begin{aligned} x(t+1) &= A_t x(t) + B_t u(t) + G_t v(t) \\ y(t) &= C_t x(t) + e(t). \end{aligned} \tag{1a}$$

Here, v and e are white noises: sequences of independent random vectors

$$\begin{aligned} E[v(t)] &= 0, & E[e(t)] &= 0 \quad \forall t \\ E[v(t)v^T(s)] &= 0, & E[e(t)e^T(s)] &= 0 \quad \text{if } t \neq s \\ E[v(t)v^T(t)] &= R_1(t), & E[e(t)e^T(t)] &= R_2(t). \end{aligned} \tag{1b}$$

The independence of the noise sequences is required in order to make $x(t)$ a Markov process.

The model (1) with the "process noise" v being Gaussian is a standard model for control applications. v then represents the combined effect of all those non-measurable inputs that in addition to u affect the states. This is the common model used both for state estimation and control design based on LQG (linear quadratic Gaussian).

But, an equally common situation is that v corresponds to an *unknown input*. It could be

- a *load disturbance e.g.,* a step change in moment load in an electric motor, a (up or down) hill for a vehicle, etc. (Sometimes, the term load disturbance is used only for the case $B_t = G_t$.)
- an event that causes the state to jump, *a change*, see *e.g.,* Gustafsson (2001).

Such unknown inputs are not naturally modeled as Gaussian noise. Instead it is convenient to capture their unpredictable nature by the distribution (*cf.* eq (2.10)-(2.11) in Ljung (1999).)

$$v(t) = \delta(t)\eta(t), \tag{2a}$$

where

$$\delta(t) = \begin{cases} 0 & \text{with probability } 1-\mu \\ 1 & \text{with probability } \mu \end{cases} \qquad \eta(t) \sim N(0, Q) \tag{2b}$$

This makes $R_1(t) = \mu Q$. The matrices A_t and G_t may further model the waveform of the disturbance as a response to the pulse in v.

We shall in this contribution discuss efficient ways of estimating the states under this assumption of the process noise v.

3 State Estimation (Smoothing)

A natural and common problem is to estimate the states $x(t)$ of the system (1) from measurements of u and y. For the case of Gaussian noise sources, this problem is of course solved by the Kalman filter and the Kalman smoother, *e.g.,* Kailath et al. (2000). For our current purpose it is of interest to view this as an explicit minimization problem. For given $x(1)$ the states $x(t), t = 2, \ldots, N$ can be computed from $v(t), t = 1, \ldots, N-1$. The quality of these state estimates could

be measured by the criterion of fit to observations:

$$\sum_{t=1}^{N} \left\| R_2^{-1/2}(t)\Big(y(t) - C_t x(t)\Big)\right\|_2^2 \tag{3}$$

where, for a vector $z = [z_1\, z_2\, \dots\, z_{n_z}]^T$, $\|z\|_p \triangleq (\sum_{i=1}^{n_z} |z_i|^p)^{1/p}$. This should be minimized w.r.t. $x(1), v(t);\ t = 1, \dots, N-1$. At the same time, the use of jumps in the states, corresponding to v should be constrained in some sense, so as to avoid over-fitting to the noisy y-measurements.

The most common way is to use a quadratic regularization

$$\min_{x(1), v(t), 1\le t\le N-1} \sum_{t=1}^{N} \left\| R_2^{-1/2}(t)\Big(y(t) - C_t x(t)\Big)\right\|_2^2 + \sum_{t=1}^{N-1} \left\| R_1^{-1/2}(t) v(t)\right\|_2^2 \tag{4}$$

which gives the classical Kalman smoothing estimate, *e.g.,* Kailath et al. (2000). In the case of Gaussian process noise v, this is also the maximum likelihood estimate and gives the conditional mean of $x(t)$ given the observations. It is a pure least squares problem, and the solution is usually given in various recursive filter forms, see *e.g.,* Ljung and Kailath (1976).

Since $x(t)$ is a given function of $x(1), v(t)$ and the known sequence $u(t)$, it may seem natural to do the minimization directly over $x(t)$, *i.e.,* to write

$$\begin{aligned}\min_{x(t), 1\le t\le N} & \sum_{t=1}^{N} \left\| R_2^{-1/2}(t)\Big(y(t) - C_t x(t)\Big)\right\|_2^2 \\ & + \sum_{t=1}^{N-1} \left\| R_1^{-1/2}(t) G_t^{\dagger}\Big(x(t+1) - A_t x(t) - B_t u(t)\Big)\right\|_2^2\end{aligned} \tag{5a}$$

where $G^{\dagger}$ is the pseudo inverse of G. However, if G is not full rank, nothing constrains the state evolution in the null space of G, so (5a) must be complemented with the constraint

$$G_t^{\perp}(x(t+1) - A_t x(t) - B_t u(t)) = 0 \tag{5b}$$

where $G^{\perp}$ is the projection onto the null-space of G,

$$G^{\perp} \triangleq I - GG^{\dagger}. \tag{6}$$

However, since several approaches can be interpreted as explicit methods to estimate $v(t)$ (or $\delta(t)$ in (2)), we shall adhere to the (equivalent) formulation (4).

4 The Proposed Method: State Smoothing by Sum-of-Norms Regularization

The type of process noise that we are interested in (see (2)) motivates a rather different regularization term than the one used in (5a).

4.1 Sum-of-Norms Regularization

To penalize state changes over time, we use a penalty or regularization term (see *e.g.,* Boyd and Vandenberghe (2004, p. 308)) that is a sum of norms of the estimated extra inputs $v(t)$:

$$\min_{x(1),v(t),1\le t\le N-1} \sum_{t=1}^{N} \left\| R_2^{-1/2}(t)\Big(y(t) - C_t x(t)\Big)\right\|_2^2 + \lambda \sum_{t=1}^{N-1} \left\| Q^{-1/2} v(t)\right\|_p \tag{7a}$$

subject to

$$x(t+1) = A_t x(t) + B_t u(t) + G_t v(t) \tag{7b}$$

where the ℓ_p-norm is used for regularization, and λ is a positive constant that is used to control the trade-off between the fit to the observations $y(t)$ (the first term) and the size of the state changes caused by $v(t)$ (the second term). The regularization norm could be any ℓ_p-norm, like ℓ_1 or ℓ_2, but it is crucial that it is a sum of norms, and not a sum of squared norms, which was used in (5a).

When the regularization norm is taken to be the ℓ_1 norm, *i.e.,* $\|z\|_1 = \sum_{i=1}^{n_z} |z_i|$, the regularization in (7a) is a standard ℓ_1 regularization of the least-squares criterion. Such regularization has been very popular recently, *e.g.,* in the much used lasso method (Tibsharani, 1996) or compressed sensing (Donoho, 2006; Candès et al., 2006).

There are two key reasons why the criterion (7a) is attractive:

- It is a convex optimization problem, so the global solution can be computed efficiently. In fact, its special structure allows it to be solved in $O(N)$ operations, so it is quite practical to solve it for a range of values of λ, even for large values of N.
- The sum-of-norms form of the regularization favors solutions where "many" (depending on λ) of the regularized variables come out as exactly zero in the solution. In this case, this implies that many of the estimates of $v(t)$ become zero (with the number of $v(t)$s becoming zero controlled roughly by λ).

A third advantage is that

- It is easy to include realistic convex state constraints without destroying convexity.

We should comment on the difference between using an ℓ_1 regularization and some other type of sum-of-norms regularization, such as sum-of-Euclidean norms. With ℓ_1 regularization, we obtain an estimate of v having many of its elements equal to zero. When we use sum-of-norms regularization, the whole estimated vector $v(t)$ often becomes zero; but when it is nonzero, typically all its elements are nonzero. In a statistical linear regression framework, sum-of-norms regularization is called group-lasso (Yuan and Lin, 2006), since it results in estimates in which many groups of variables are zero.

Remark 1. A criterion (7a) handles the process noise as described in (2) well. In some situations it may however be more accurate to assume a Gaussian noise component in the process noise as well, *i.e.,*

$$\begin{aligned} x(t+1) &= A_t x(t) + B_t u(t) + G_t v(t) + H_t w(t) \\ y(t) &= C_t x(t) + e(t) \end{aligned} \tag{8}$$

with $w \sim N(0, S)$ and v as defined in (2). It is then motivated to use a criterion

$$\min_{\substack{x(1), v(t), w(t) \\ 1 \le t \le N-1}} \sum_{t=1}^{N} \left\| R_2^{-1/2}(t)\big(y(t) - C_t x(t)\big) \right\|_2^2 + \sum_{t=1}^{N-1} \lambda \left\| Q^{-1/2} v(t) \right\|_p + \left\| S^{-1/2} w(t) \right\|_2^2 \tag{9a}$$

$$\text{s.t.} \quad x(t+1) = A_t x(t) + B_t u(t) + G_t v(t) + H_t w(t) \tag{9b}$$

rather than (7).

4.2 Regularization Path and Critical Parameter Value

The estimated sequence $v(t)$ as a function of the regularization parameter λ is called the *regularization path* for the problem. Roughly, larger values of λ correspond to estimated $x(t)$ with worse fit, but an estimate of $v(t)$ with many zero elements. A basic result from convex analysis tells us that there is a value $\lambda^{\max}$ for which the estimated $v(t)$ is identically zero if and only if $\lambda \geq \lambda^{\max}$. In other words, $\lambda^{\max}$ gives the threshold above which $v(t) = 0$, $t = 1, ..., N$. The critical parameter value $\lambda^{\max}$ is very useful in practice, since it gives a very good starting point in finding a suitable value of λ. Reasonable values are typically in the order of $0.01\lambda^{\max}$ to $\lambda^{\max}$.

Proposition 1. *Introduce ε_t for the (process) noise free residual*

$$\varepsilon_t \triangleq y(t) - C_t \left(\sum_{r=1}^{t-1} \prod_{s=r+1}^{t-1} A_s B_r u(r) + \left(\prod_{s=1}^{t-1} A_s \right) x(1) \right) \tag{10}$$

and take $\bar{\varepsilon}_t$ to be ε_t evaluated at

$$x(1) = \arg\min_{x(1)} \sum_{t=1}^{N} \left\| R_2^{-1/2}(t) \varepsilon_t \right\|_2^2. \tag{11}$$

We can then express $\lambda^{\max}$ as

$$\lambda^{\max} = \max_{k=1,\ldots,N-1} \left\| 2 \sum_{t=k+1}^{N} \left(R_2^{-1/2}(t) C_t \left(\prod_{s=k+1}^{t-1} A_s \right) G_k Q^{1/2} \right)^T R_2^{-1/2}(t) \bar{\varepsilon}_t \right\|_q. \tag{12}$$

with $\|\cdot\|_q$ the dual norm ($\frac{1}{p} + \frac{1}{q} = 1$) associated with $\|\cdot\|_p$ used in (7a).

The proof is given in the appendix.

4.3 Iterative Refinement

To (possibly) get even more zeros in the estimate of $v(t)$, with no or small increase in the fitting term, iterative re-weighting can be used (Candès et al., 2008). We modify the regularization term in (7a) and consider

$$\min_{x(1),v(t),1\le t\le N-1} \sum_{t=1}^{N} \left\| R_2^{-1/2}(t)\Big(y(t) - C_t x(t)\Big)\right\|_2^2 + \lambda \sum_{t=1}^{N-1} \alpha(t)\left\|Q^{-1/2}v(t)\right\|_p \qquad (13)$$

where $\alpha(1),\dots,\alpha(N-1)$ are positive weights used to vary the regularization over time. Iterative refinement proceeds as follows. We start with all weights equal to one *i.e.,* $\alpha^{(0)}(t) = 1$. Then for $i = 0, 1, \dots$ we carry out the following iteration until convergence (which is typically in just a few steps).

1. *Find the state estimate.*
   Compute the optimal $v^{(i)}(t)$ using (13) with the weighted regularization using weights $\alpha^{(i)}$.
2. *Update the weights.*
   For $t = 1,\dots,N-1$, set $\alpha^{(i+1)}(t) = 1/(\epsilon + \|Q^{-1/2}v^{(i)}\|_p)$.

Here ϵ is a positive parameter that sets the maximum weight that can occur.

One final step is also useful. From our final estimate of $v(t)$, we simply define set of times T at which an estimated load disturbance occurs (*i.e.,* $T = \{t|v(t) \neq 0\}$) and carry out a final optimization over just $v(t), t \in T$. The algorithm is summarized in Algorithm 1.

Algorithm 1 State Estimation by Sum-of-Norms Regularization (STATESON)

Given A_t, B_t, C_t, G_t, Q, $R_2(t)$ and $\{(y(t), u(t))\}_{t=1}^N$. Let ϵ be a positive parameter, set $\alpha^{(0)}(t) = 1$ for $t = 1,\dots,N-1$ and $i = 0$. Then, for a chosen λ:

1. Compute the optimal $v^{(i)}(t)$ using (13) with $\alpha = \alpha^{(i)}$.
2. Set $\alpha^{(i+1)}(t) = 1/(\epsilon + \|Q^{-1/2}v^{(i)}\|_p)$.
3. If convergence, go to the next step, otherwise set $i = i + 1$ and return to (1).
4. Compute a final estimate of $v(t)$ using (13) by just performing the minization over those $v(t)$ for which $v^{(i+1)}(t)$ is non-zero.

Remark 2. If the jump covariance Q in (2) is known or can be given a good value, the final optimization step (step (4) in Algorithm 1) should be replaced by a Kalman smoother with the time-varying process noise

$$R_1(t) = \begin{cases} 0 & \text{for } t \text{ such that } v(t) = 0 \\ Q & \text{otherwise.} \end{cases}$$

It should be noticed that if the correct jump-times and Q have been found, this is actually optimal in the sense that no other smoother (linear or nonlinear) can achieve an unbiased estimate with a lower error covariance.

4.4 Solution Algorithms and Software

Many standard methods of convex optimization can be used to solve the problem (7a). Systems such as CVX (Grant and Boyd, 2010, 2008) or YALMIP (Löfberg, 2004) can readily handle the sum-of-norms regularization, by converting the problem to a cone problem and calling a standard interior-point method. For the special case when the ℓ_1 norm is used as the regularization norm, more efficient special purpose algorithms and software can be used, such as `l1_ls` (Kim et al., 2007). Recently many authors have developed fast first order methods for solving ℓ_1 regularized problems, and these methods can be extended to handle the sum-of-norms regularization used here; see, for example, Roll (2008§2.2). Both interior-point and first-order methods have a complexity that scales linearly with N.

5 Other Approaches

Many methods for state estimation with non-Gaussian noise as in (2) are suggested in the literature, both in connection with change detection, *e.g.,* Gustafsson (2001), and target tracking, *e.g.,* Bar-Shalom et al. (2001). Many of them can be seen as ways to explicitly estimate $v(t)$ or $\delta(t)$ in (2).

If the δ-sequence was known, the problem could be treated as a Kalman smoother with known, time varying $R_1(t)$ ($R_1(t_k) = Q$ for those t_k with $\delta(t_k) = 1$ and $R_1(t_\ell) = 0$ otherwise.) This is sometimes known as the *clairvoyant filter* (or filter with an oracle). The (time-varying) smoothed state error covariance matrix can readily be computed for this case. Clearly this gives a lower bound for any possible estimate, which no other (linear or nonlinear) filter can beat.

Based on the model (2), a number of approximative methods have been developed. If $\delta(t)$ is not known, we could hypothesize in each time step that it is either 0 or 1. This leads to a large bank (2^N) of Kalman filters as the optimal solution. The posterior probability of each filter can be estimated from this bank, which consists of a weighted sum of the state estimates from each filter. In practice, the number of filters in the bank must be limited, and there are two main options (see Chapter 10 in Gustafsson (2010)):

- Merging trajectories of different $\delta(t)$ sequences. This includes the well-known IMM filter, see Blom and Bar-Shalom (1988).
- Pruning, where unlikely sequences are deleted from the filter bank.

Change detection techniques can also be used to detect the time instances when $\delta(t) = 1$. In the linear case, *e.g.,* a change detection algorithm can be applied to the innovations of a Kalman filter to detect jumps in the process noise. If a jump is detected, the process noise covariance in the Kalman filter is made *e.g.,* 10 times larger to adapt to the abrupt state change.

6 Numerical Illustration

Consider the discrete time model of a DC motor (see *e.g.*, Ljung (1999, pp. 95-97), $T_s = 0.1$ s, $\tau = 0.286$, $\beta = 40$)

$$\begin{aligned} x(t+1) &= \begin{bmatrix} 0.7047 & 0 \\ 0.08437 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 11.81 \\ 0.6250 \end{bmatrix} \Big(u(t) + v(t) \Big) \\ s(t) &= \begin{bmatrix} 0 & 1 \end{bmatrix} x(t) \\ y(t) &= s(t) + e(t), \end{aligned} \tag{14}$$

with $u(t)$ a ± 0.1 PRBS signal (pseudo-random binary sequence), $e(t) \sim N(0,1)$ and $x(1) \sim N(0, I)$. The system was simulated with $v(t)$ distributed according to (2) with $\mu = 0.015$ and $Q = 0.5$, which gave the particular load disturbance sequence

$$v(t) = \begin{cases} -0.6 & \text{for } t = 55, \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

and $y(t)$, $t = 1, \ldots, 100$, was observed. The resulting estimate of the angle $s(t)$ using $\lambda = 0.1\lambda^{\max}$, two refinement iterations, $Q = 0.5$, $R_2 = 1$ and an Euclidean norm in the regularization is shown in Figure 1. Figure 1 also shows the measurement $y(t)$ and the true sequence $s(t)$ that was used to generate the $y(t)$ measurements. $v(t)$ was estimated to

$$v(t) = \begin{cases} -0.55 & \text{for } t = 55, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

The mean squared error (MSE) for the state estimate was for this particular setup 0.28. Since the jump-time was correctly found, the estimate almost coincide with the estimate of the clairvoyant estimator. If a Kalman smoother is applied with the true measurement and process noise variances ($R_1 = \mu Q = 0.0075$, $R_2 = 1$) a MSE of 1.0 was obtained. The result is summarized in Table 1.

Table 1: *MSE for the state estimate obtained by the Kalman smoother (BLUE), state estimation by sum-of-norms regularization (STATESON) and clairvoyant smoother.*

Algorithm	MSE
BLUE	1.0
STATESON	0.28
Clairvoyant smoother	0.12

Let us now compare the state smoothing by sum-of-norms regularization (7) with some other methods that have been suggested in the literature.

For the same sequence $v(t)$ we run Monte-Carlo simulations over realizations of the measurement noise $e(t)$ with $R_2 = 1$. We run 2000 simulations and compute the smoothed state squared error over the runs.

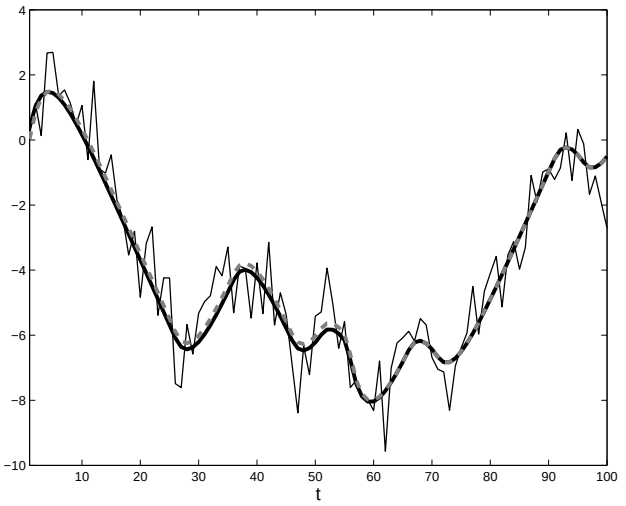

Figure 1: *The resulting estimate of $s(t)$ showed with a solid thick line; dashed line, true sequence $s(t)$; solid thin black line, measurements $y(t)$. The jump in $v(t)$ at $t = 55$ is hardly visible.*

We compare:

1. The proposed method state smoothing by sum-of-norms regularization.
2. Conventional Kalman smoother with $R_1 = 0.0075$ and $R_2 = 1$.
3. Kalman smoother together with CUSUM (cumulative sum, Page (1954), see also Algorithm 2). First, CUSUM was applied to both the whitened innovations and the negative whitened innovations of a Kalman filter with $R_2 = 1$. R_1 was set to 0.5 when $g(t+1)$ exceeded h but was otherwise 0. In a second step, a Kalman smoother was applied with $R_1 = 0.5$ at the time instances of detected changes and $R_1 = 0$ otherwise. $h = 10$ and $\gamma = 1$ gave good performance.
4. IMM smoother with two modes, $R_2 = 1$ for both and $R_1 = 0$ and 0.5, respectively, with probabilities 0.985 and 0.015. The IMM smoothing implementation of Särkkä and Hartikainen (2007) was used.
5. The lower bound according to the clairvoyant smoother.

Algorithm 2 CUSUM

Set $g(1) = 0$. For a chosen γ and h, a change in the signal $r(t)$ is detected by observing when

$$g(t+1) = \max(g(t) + r(t) - \gamma, 0) \tag{17}$$

exceeds h. After a change has been detected, g is reset to zero. The time of the change is taken as the previous time instance for which $g(t) = 0$.

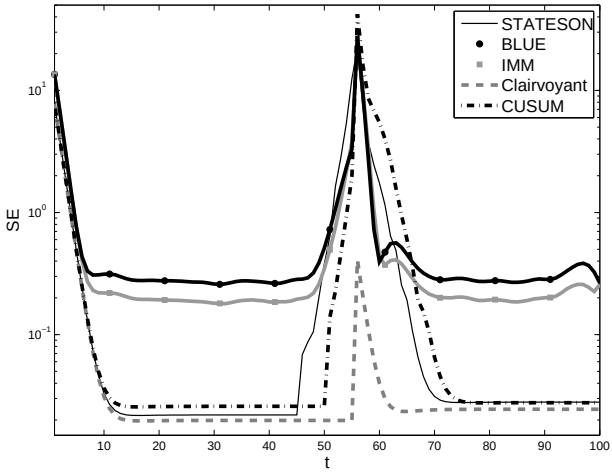

Figure 2: *Mean (over Monte Carlo runs) squared errors (SE) versus time. All the sample SE means were taken over Monte Carlo runs and not time. 2000 Monte Carlo runs were used.*

See Figure 2. We see that state smoothing by sum-of-norms regularization outperforms the Kalman smoother and the IMM smoother and that it gets fairly close to the lower bound given by the clairvoyant smoother. Kalman smoother together with CUSUM does almost as good as state smoothing by sum-of-norms regularization. Figure 3 shows a plot of estimated v values by STATESON for the 2000 runs. Figure 4 shows the estimated v by the Kalman smoother together with CUSUM.

Let us also investigate how the methods perform under varying signal-to-noise conditions (the design parameters held fixed over the different SNRs, tuned for a SNR of 5.7). 200 Monte Carlo runs were performed for a number of different R_2 to produce the plot shown in Figure 5. The mean of the squared errors were taken both over time and the 200 Monte Carlo runs with the same signal-to-noise ration (SNR). The SNR was computed as

$$SNR = \left(\frac{\sum_{t=1}^{N} |s(t)|}{\sum_{t=1}^{N} |e(t)|} \right)^2 \tag{18}$$

where $s(t)$ is the signal given in (14) if $u(t) \equiv 0$ is feeding (14) and $e(t)$ is the measurement noise. The plot shows that state smoothing by sum-of-norms regularization does well in comparison with the compared methods. It may seem surprising that the Kalman smoother together with CUSUM dose not do better. CUSUM detects the changes in most cases, however, it does not give an accurate estimate of the time of the changes (not even in high SNR). CUSUM also suffers of varying SNR and would do better if it was retuned for each new SNR value.

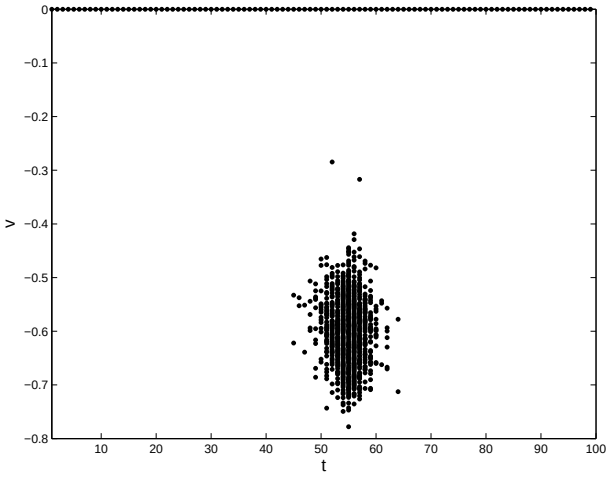

Figure 3: *The STATESON estimates of v that gave the squared errors in Figure 2.*

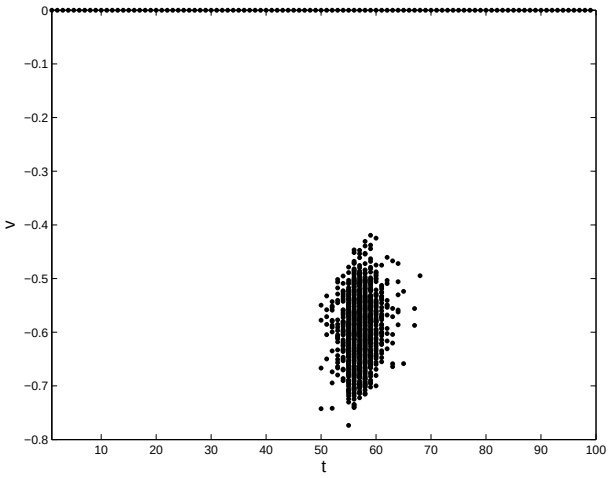

Figure 4: *The estimates of v that gave the squared errors in Figure 2 for the Kalman smoother together with CUSUM.*

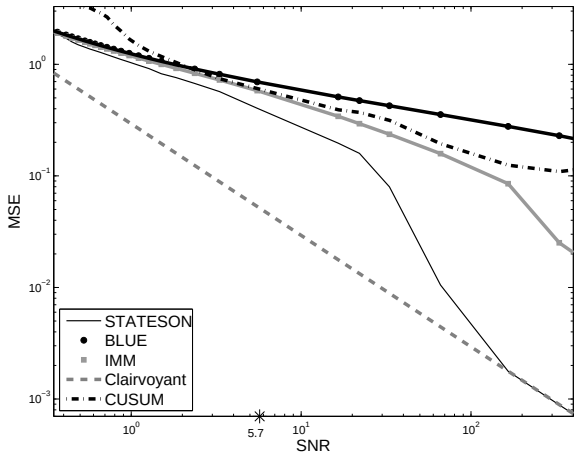

Figure 5: *MSE versus SNR. All the means were taken over Monte Carlo runs and time. 200 Monte Carlo runs were used for each SNR value. The simulations shown in Figure 2 had a SNR of* 5.7 *(in average).*

7 Extension to Nonlinear Models

An extension to nonlinear systems is of interest since many systems are poorly described by linear approximations. We do this in an extended-Kalman-filter-like fashion and approximate the nonlinear system by a time-varying linear model. To get an initial state trajectory estimate, we use an extended Kalman filter. The algorithm is summarized in Algorithm 3.

Algorithm 3 State Estimation by Sum-of-Norms Regularization Using Nonlinear Models

Given a nonlinear state space model and $\{(y(t), u(t))\}_{t=1}^{N}$.

1. Find an initial state trajectory estimate by applying an extended Kalman filter.
2. Create a time-varying approximation of the nonlinear system by linearizing around the computed state trajectory.
3. Apply Algorithm 1 to obtain a new state estimate.
4. Return to step (2) if necessary.

Example 1: A Nonlinear Example – A Pendulum

Consider the pendulum shown in Figure 6. Its dynamical behavior using a mass $m = 1$ and a pole length $L = 1$ is described by the nonlinear relation

$$\frac{d}{dt}\begin{bmatrix}\theta \\ d\theta/dt\end{bmatrix} = \begin{bmatrix}d\theta/dt \\ -g\sin\theta\end{bmatrix} + \begin{bmatrix}0 \\ F\end{bmatrix}. \tag{19}$$

g is the gravitational constant ($g = 9.81$ was used in the simulations). Using Euler

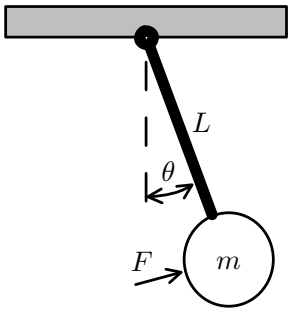

Figure 6: *Notation for the Pendulum in Example 1.*

integration with a time step of 0.05, we obtain the time-discrete representation ($x_1 = \theta, x_2 = d\theta/dt$)

$$\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} x_1(t-1) + 0.05x_2(t-1) \\ x_2(t-1) - 0.05g \sin x_1(t-1) \end{bmatrix} + \begin{bmatrix} 0 \\ F(t) \end{bmatrix}. \tag{20}$$

Let us assume that we can measure the quantity

$$y(t) = \sin x_1(t) + e(t), \quad e(t) \sim N(0, 0.05) \tag{21}$$

and that the system is driven by the process noise $F(t) = w(t) + v(t)$, $w(t) \sim N(0, 0.0005)$ and

$$v(t) = \begin{cases} 1 & \text{for } t = 500, \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

A realization of $y(t)$ is given in Figure 7. $x_1(1) = \pi/3$ and $x_2(1) = 0$ were used

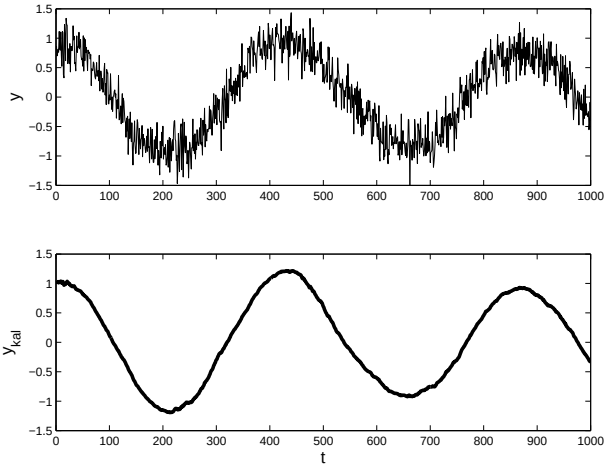

Figure 7: *A plot of the data (top figure) and the initial estimate obtained from the EKF (bottom figure).*

to initialize the system. The result obtained by applying an EKF is also given in Figure 7. We now proceed as described in Algorithm 3 (the criterion (9) was used, see Remark 1, due to the Gaussian component in the process noise) and use the EKF estimate to obtain an initial linear time-varying representation of the pendulum around the trajectory. Using $\lambda = 0.05$ and two iterations in Algorithm 3, the result given in Figure 8 was obtained. As seen, the impulse at $t = 500$ was correctly detected.

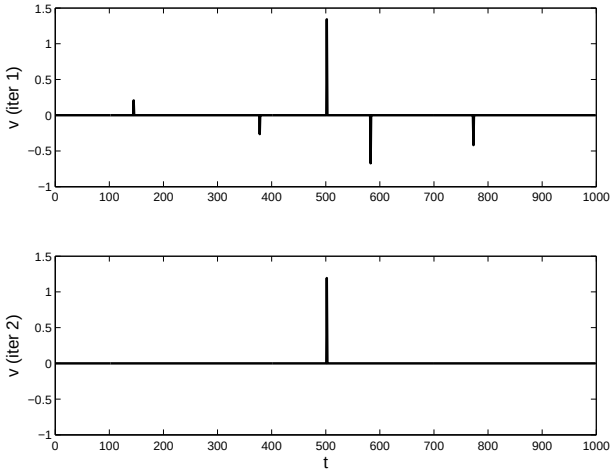

Figure 8: *Estimates of v. Top plot, iteration 1 and bottom plot, iteration 2.*

8 Conclusion

A novel formulation of the state estimation problem in the presence of abrupt changes has been presented. The proposed approach treats the state smoothing problem as a constrained least-squares problem with a sum-of-norms regularization. Some numerical illustrations have been given. The approach can be seen an an extension of the technique used for segmentation of ARX-models in Ohlsson et al. (2010c). The extension to nonlinear models was also discussed and exemplified.

Acknowledgment

This work was supported by the Strategic Research Center MOVIII, funded by the Swedish Foundation for Strategic Research, SSF, and CADICS, a Linnaeus center funded by the Swedish Research Council.

A Appendix

A.1 Proof of Proposition 1

To verify our formula for $\lambda^{\max}$ we use convex analysis (Rockafellar, 1996; Bertsekas et al., 2003; Borwein and Lewis, 2000). Fist note that

$$x(t) = G_{t-1}v(t-1) + A_{t-1}x(t-1) + B_{t-1}u(t-1) \tag{23a}$$

$$= \sum_{r=1}^{t-1}\left(\prod_{s=r+1}^{t-1} A_s\right)\Big(G_r v(r) + B_r u(r)\Big) + \left(\prod_{s=1}^{t-1} A_s\right)x(1). \tag{23b}$$

Introduce

$$\varepsilon_t \triangleq y(t) - C_t\Big(\sum_{r=1}^{t-1}\prod_{s=r+1}^{t-1} A_s B_r u(r) + \left(\prod_{s=1}^{t-1} A_s\right)x(1)\Big) \tag{24}$$

and let $\bar{\varepsilon}_t$ be ε_t evaluated at the $x(1)$ minimizing

$$\min_{x(1)} \sum_{t=1}^{N} \left\|R_2^{-1/2}(t)\varepsilon_t\right\|_2^2. \tag{25}$$

(7) can then be written as

$$\min_{x(1),\bar{v}(t),t=1,\ldots,N-1} \sum_{t=1}^{N}\left\|R_2^{-1/2}(t)\left(\varepsilon_t - C_t\sum_{r=1}^{t-1}\left(\prod_{s=r+1}^{t-1} A_s\right)G_r Q^{1/2}\bar{v}(r)\right)\right\|^2 + \lambda\sum_{t=1}^{N-1}\|\bar{v}(t)\|_p \tag{26}$$

with $\bar{v}(t) \triangleq Q^{-1/2}v(t)$ and using (24). It is clear that the subdifferential of $\|\bar{v}(t)\|_p$ evaluated at $\bar{v}(t) = 0$ is the unit ball in the dual norm $\|\cdot\|_q$, $1/p + 1/q = 1$. $\lambda^{\max}$ must therefore satisfy

$$\lambda^{\max} = \max_k \left\|\nabla_{\bar{v}(k)}\sum_{t=1}^{N}\left\|R_2^{-1/2}(t)\left(\bar{\varepsilon}_t - C_t\sum_{r=1}^{t-1}\left(\prod_{s=r+1}^{t-1} A_s\right)G_r Q^{1/2}\bar{v}(r)\right)\right\|^2\Big|_{\bar{v}\equiv 0}\right\|_q \tag{27a}$$

$$= \max_k \left\|2\sum_{t=k+1}^{N}\left(R_2^{-1/2}(t)C_t\left(\prod_{s=k+1}^{t-1} A_s\right)G_k Q^{1/2}\right)^T R_2^{-1/2}(t)\bar{\varepsilon}_t\right\|_q. \tag{27b}$$

Bibliography

Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. John Wiley & Sons, 2001.

M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes – Theory and Application*. Prentice-Hall, Englewood Cliffs, NJ, 1993.

D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

H. A. P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783, August 1988.

J. M. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS Books in Mathematics. Springer, 2000.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, February 2006.

E. J. Candès, M. B. Wakin, and S. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications, special issue on sparsity*, 14(5):877–905, December 2008.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, August 2010.

F. Gustafsson. *Adaptive Filtering and Change Detection*. Wiley, New York, 2001.

F. Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur AB, 2010.

S. Hui and S. H. Zak. Observer design for systems with unknown inputs. *International Journal of Applied Mathematics and Computer Science (AMCS)*, 15 (4), 2005.

T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice-Hall, Englewood Cliffs, NJ, 2000.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, December 2007.

S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

L. Ljung. *System Identification - Theory for the User*. Prentice-Hall, Upper Saddle River, N.J., 2nd edition, 1999.

L. Ljung and T. Kailath. A unified approach to smoothing formulas. *Automatica*, 12(2):147–157, 1976.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL `http://control.ee.ethz.ch/~joloef/yalmip.php`.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State smoothing by sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. State estimation under abrupt changes using sum-of-norms regularization. *Automatica*, 2010b. Submitted, under revision.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010c.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

R. Patton, P. Frank, and R. Clark. *Fault Diagnosis in Dynamic Systems – Theory and Application*. Prentice Hall, 1989.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1996.

J. Roll. Piecewise linear solution paths with application to direct weight optimization. *Automatica*, 44:2745–2753, 2008.

S. Särkkä and J. Hartikainen. EKF/UKF toolbox for Matlab 7.x, November 2007. Version 1.2.

R. Tibsharani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B (Methodological)*, 58(1):267–288, 1996.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

Paper D

Trajectory Generation Using Sum-of-Norms Regularization

Authors: Henrik Ohlsson, Fredrik Gustafsson, Lennart Ljung and Stephen Boyd

Edited version of the paper:

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010b. To appear.

Trajectory Generation Using Sum-of-Norms Regularization

Henrik Ohlsson*, Fredrik Gustafsson*, Lennart Ljung* and Stephen Boyd**

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{ohlsson,fredrik,
ljung}@isy.liu.se

**Dept. of Electrical Engineering,
Stanford University,
Stanford, CA 94305 USA
boyd@stanford.edu

Abstract

Many tracking problems are split into two sub-problems, first a smooth reference trajectory is generated that meet the control design objectives, and then a closed loop control system is designed to follow this reference trajectory as well as possible. Applications of this kind include (autonomous) vehicle navigation systems and robotics. Typically, a spline model is used for trajectory generation and another physical and dynamical model is used for the control design. Here we propose a direct approach where the dynamical model is used to generate a control signal that takes the state trajectory through the waypoints specified in the design goals. The strength of the proposed formulation is the methodology to obtain a control signal with compact representation and that changes only when needed, something often wanted in tracking. The formulation takes the shape of a constrained least-squares problem with sum-of-norms regularization, a generalization of the ℓ_1-regularization. The formulation also gives a tool to, *e.g.,* in model predictive control, prevent chatter in the input signal, and also select the most suitable instances for applying the control inputs.

1 Introduction

Consider a dynamic system with output $y(t)$ where the design objectives can be formulated as

$$y(t_k) \approx W(t_k), \quad t_k \in T. \tag{1}$$

These points, $W(t_k)$, will be referred to as *waypoints*. The conventional approach is based on the following two steps:

1. Generate spline functions between the waypoints to get a smooth trajectory $y_{ref}(t)$.

2. Design a control system that generates a control input $u(t)$ that makes the system output as close as possible to $y_{ref}(t)$.

A potential problem is that the reference trajectory $y_{ref}(t)$ may not be feasible for the system in that it varies too fast to allow the dynamics to follow the trajectory with given input saturations. Conversely, it may vary too slowly, giving a conservative control performance.

The suggested approach circumvents this problem by generating the reference trajectory based on the system dynamics, rather than spline functions. The result is an input sequence $\{u(t)\}$, with a compact representation, that gives the output sequence $\{y(t)\}$ that passes (within a given distance to) the waypoints. In the presence of disturbances and process noise, a feedback control is still necessary, but this provide small corrections to the already computed input sequence.

The advantages of the method include:

- The complexity is linear in time.
- Control signal saturations can be incorporated.
- Different constraints on input smoothness can be incorporated, as piecewise linear or constant inputs. This is relevant in applications for which a change in control signal is associated with a cost (maybe communicating a change is costly) or the storage is limited.
- State constraints can be added for forbidden regions in the state space.
- The trajectory can be forced to pass the waypoints or any specified distance to them.

Example 1: Two Dimensional Tracking Problem

Considering a two dimensional tracking problem. Assume that we would like the system output to take the values

$$\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}10\\-10\end{bmatrix}, \begin{bmatrix}20\\0\end{bmatrix}, \begin{bmatrix}30\\0\end{bmatrix}, \begin{bmatrix}30\\10\end{bmatrix}, \begin{bmatrix}20\\10\end{bmatrix}, \begin{bmatrix}10\\10\end{bmatrix}, \begin{bmatrix}0\\0\end{bmatrix} \tag{2}$$

at

$$t = 0,\ 1,\ 2,\ 3,\ 4,\ 4.5,\ 5,\ 6. \tag{3}$$

(2) and (3) here define our waypoints. Assume a linear state space description of our system and a limitation on the control signal $\|u\|_2^2 < 40$. We may also assume that its impractical to communicate or to store an output reference as a look-up table with a value for each sample time. This is often the case for industrial robots and autonomous vehicle navigation systems. The solution is to fit a spline to the waypoints and use this as a reference trajectory. In this particular example, the spline would have 8 breakpoints (one for each waypoint). A feedback controller is then applied to follow the spline reference trajectory. Since the spline fitting problem is commonly seen only as a geometrical task and no consideration to the dynamical system and input constraint are taken, the spline reference may be impossible to follow. We will come back to this problem later, in Section 4.

2 Problem Formulation

Given a system

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{4}$$

$x \in \mathcal{R}^n$, $u \in \mathcal{R}^m$ and a set of so called *waypoints*

$$W = \{W(t), t \in T\},\ T = \{t_k, k = 1, \ldots, M\}. \tag{5}$$

The problem is to find u so that the output of (4) closely follows the waypoints:

$$y(t_k) \approx W(t_k), \quad t_k \in T \tag{6}$$

This should be solved under a number of constraints:

- Given a time grid T_{grid}, based on the sampling time T_s : $T_{grid} = \{t_s = sT_s, s = 1, \ldots, N\}$. Assume that T is a subset of T_{grid}.
- The pth derivative of u is an impulse train on the grid T_{grid}:

$$u^{(p)}(t) = \sum_{k=1}^{N} v_k \delta(t - kT_s), \quad v_k \in \mathcal{R}^{n_u} \tag{7}$$

- As many as possible of the terms v_k in (7) should be zero.
- There are input and state constraints (on the grid T_{grid}):

$$\begin{aligned} x(kT_s) &\in \mathcal{K} \\ u(kT_s) &\in \mathcal{U} \end{aligned} \tag{8}$$

We should comment on (7). The common way to impose a smooth output, which often is desirable (see *e.g.,* Gulati and Kuipers (2008), Teruya et al. (2008)), is by using a smooth spline as a reference. By using (7) we impose a smoothness constrain on the control input and also implicitly on the output.

That many of the v_ks are zero imply a compact representation of the input signal. This may be advantageous when storage is limited. It also means few changes in the control signal. This may save money by reduced communication but may also save actuators (see discussions on *chattering* in model predictive control (MPC), see *e.g.,* Wojsznis et al. (2003) or Naus et al. (2008)).

3 Proposed Method

Let the upper part of the extended vector $X(t)$ be the state $x(t)$ of (4), while the lower part is made up of the $p-1$ derivatives of u,

$$X(t) = \begin{bmatrix} x(t) & u(t) & \dot{u}(t) & \ddot{u}(t) & \ldots & u^{(p-1)}(t) \end{bmatrix}^T. \tag{9}$$

Let us accordingly extend the model (4) by (I_r is the $r \times r$ unit matrix):

$$\dot{X}(t) = \bar{A}X(t) + \bar{B}u^{(p)}(t) \tag{10a}$$

$$y(t) = \bar{C}X(t) \tag{10b}$$

$$\bar{A} = \begin{bmatrix} A & B & 0 & 0 & \dots & & 0 \\ 0 & 0 & I_m & 0 & \dots & & 0 \\ 0 & 0 & 0 & & & & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ & & & & & & 0 \\ & & & & & 0 & I_m \\ 0 & 0 & 0 & \dots & & 0 & 0 \end{bmatrix} \tag{10c}$$

$$\bar{B} = \begin{bmatrix} 0 & \dots & 0 & I_m \end{bmatrix}^T \tag{10d}$$

$$\bar{C} = \begin{bmatrix} C & 0 & \dots & 0 \end{bmatrix} \tag{10e}$$

Then by sampling this system with a pulse-train as input (`c2d(syst,Ts,'imp')` in MATLAB) we obtain

$$X(kT_s + T_s) = FX(kT_s) + Gv_k \tag{11a}$$

$$x(kT_s) = PX(kT_s) \tag{11b}$$

$$y(kT_s) = HX(kT_s) \tag{11c}$$

$$u(kT_s) = RX(kT_s) \tag{11d}$$

with

$$F = e^{\bar{A}T_s} \tag{12a}$$

$$G = F\bar{B} \tag{12b}$$

$$H = \bar{C} \tag{12c}$$

$$P = \begin{bmatrix} I_n & 0 & 0 & \dots & 0 \end{bmatrix} \tag{12d}$$

$$R = \begin{bmatrix} 0 & I_m & 0 & \dots & 0 \end{bmatrix} \tag{12e}$$

Let us assume that $x(0)$ is known and $u(0) = \dot{u}(0) = \cdots = u^{(p-1)}(0) = 0$ for simplicity. We can now phrase the problem as

$$\min \sum_{t \in T} \left\| y(t) - W(t) \right\|_2^2 \tag{13}$$

w.r.t. v_k under the constraints (11a), (8) and trying to have as many v_k as possible equal to zero. To capture the latter we wish to use *sum-of-norms regularization*:

$$\min_{v_k, k=1,\dots,N} \sum_{t \in T} \left\| y(t) - W(t) \right\|_2^2 + \lambda \sum_{k=1}^{N} \left\| v_k \right\|_{\mathsf{p}} \tag{14a}$$

$$u(kT_s) \in \mathcal{U} \tag{14b}$$

$$x(kT_s) \in \mathcal{X} \tag{14c}$$

where $y(kT_s)$, $x(kT_s)$ and $u(kT_s)$ are generated from v_k and $X(0)$. $\|\cdot\|_p$ is defined as $\|z\|_p \triangleq (\sum_{i=1}^{n_z} |z_i|^p)^{1/p}$ for a vector $z = [z_1\ z_2 \dots z_{n_z}]^T$.

The last term in the cost is a regularization term, and λ is a positive constant that is used to control the trade-off between the fit to the waypoints $W(t)$ (the first term) and the size of the state changes caused by v_k (the second term). The regularization norm could actually be any ℓ_p-norm, like ℓ_1 or ℓ_2, but it is crucial that it is a sum of norms, and not a sum of squared norms.

When the regularization norm is taken to be the ℓ_1 norm, *i.e.*, $\|z\|_1 = \sum_{k=1}^{n} |z_k|$, the regularization in (14a) is a standard ℓ_1 regularization of the least-squares criterion. Such regularization has been very popular recently, *e.g.*, in the much used lasso method (Tibsharani, 1996) or compressed sensing (Donoho, 2006; Candès et al., 2006). See also Kim et al. (2009) and Ohlsson et al. (2010b) for relevant contributions.

There are two key reasons why the criterion (14a) is attractive:

- It is a convex optimization problem, so the global solution can be computed efficiently. In fact, its special structure allows it to be solved in $O(N)$ operations, so it is quite practical to solve it for a range of values of λ, even for large values of N.
- The sum-of-norms form of the regularization favors solutions where "many" (depending on λ) of the regularized variables come out as exactly zero in the solution. In this case, this implies that many of the estimates of v_k become zero (with the number of v_ks becoming zero controlled roughly by λ).

A third advantage is that

- It is easy to include realistic state constraints without destroying convexity.

We should comment on the difference between using an ℓ_1 regularization and some other type of sum-of-norms regularization, such as sum-of-Euclidean norms. With ℓ_1 regularization, we obtain an estimate of v_k having many of its elements equal to zero. When we use sum-of-norms regularization, the whole estimated vector v_k often becomes zero; but when it is nonzero, typically all its elements are nonzero. In a statistical linear regression framework, sum-of-norms regularization is called group-lasso (Yuan and Lin, 2006), since it results in estimates in which many groups of variables are zero.

One final step is also useful. From our estimate of v_k from (14a), we simply carry out a final least-squares fit over the nonzero v_k (fixing the other v_k to zero). This typically gives a small improvement in fit to the waypoints.

3.1 Solution Algorithms and Software

Many standard methods of convex optimization can be used to solve the problem (14a). Systems such as CVX (Grant and Boyd, 2010, 2008) or YALMIP (Löfberg, 2004) can readily handle the sum-of-norms regularization, by converting

the problem to a cone problem and calling a standard interior-point method. For the special case when the ℓ_1 norm is used as the regularization norm, more efficient special purpose algorithms and software can be used, such as `l1_ls` (Kim et al., 2007). Recently many authors have developed fast first order methods for solving ℓ_1 regularized problems, and these methods can be extended to handle the sum-of-norms regularization used here; see, for example, Roll (2008§2.2). Both interior-point and first-order methods have a complexity that scales linearly with N.

4 Numerical Illustration

Example 2: Two Dimensional Tracking Problem, Cont'd

Let us return to Example 1. Assume that we chose to model our system with the model (see Chapter 13 of Gustafsson (2010), there called a constant acceleration model)

$$\begin{aligned}\dot{x}(t) &= \begin{bmatrix}0 & I_2 & 0\\ 0 & 0 & I_2\\ 0 & 0 & 0\end{bmatrix} x(t) + \begin{bmatrix}0\\ 0\\ I_2\end{bmatrix} u(t),\\ y(t) &= \begin{bmatrix}I_2 & 0 & 0\end{bmatrix} x(t).\end{aligned} \tag{15}$$

The two first elements of the state x are the x- and y-position, the third and fourth, velocity in the x- and y-direction and the two last elements, the acceleration in x- and y-direction. u is the jerk (the derivative of the acceleration). Assume now that we believe that a piecewise constant input u in (15), (that means that $p = 1$ in (7)) gives a smooth enough output. The requirement of a piecewise constant input implies that we have to extend our model with a integrator. We obtain the extended model

$$\begin{aligned}\dot{x}(t) &= \begin{bmatrix}0 & I_2 & 0 & 0\\ 0 & 0 & I_2 & 0\\ 0 & 0 & 0 & I_2\\ 0 & 0 & 0 & 0\end{bmatrix} x(t) + \begin{bmatrix}0\\ 0\\ 0\\ I_2\end{bmatrix} \sum_k v_k \delta(t - kT_s),\\ y(t) &= \begin{bmatrix}I_2 & 0 & 0 & 0\end{bmatrix} x(t),\end{aligned} \tag{16}$$

and seek a "sparse" pulse train $\{v_k\}$. Discretizing (16) under the assumption that v_k is a pulse train and with $T_s = 0.1$ gives (see (11) and (12))

$$\begin{aligned}X(kT_s + T_s) &= \begin{bmatrix}I_2 & 0.1I_2 & 0.005I_2 & 0.0002I_2\\ 0 & I_2 & 0.1I_2 & 0.005I_2\\ 0 & 0 & I_2 & 0.1I_2\\ 0 & 0 & 0 & I_2\end{bmatrix} X(kT_s) + \begin{bmatrix}0.0002I_2\\ 0.005I_2\\ 0.1I_2\\ I_2\end{bmatrix} v_k,\\ y(kT_s) &= \begin{bmatrix}I_2 & 0 & 0 & 0\end{bmatrix} X(kT_s).\end{aligned} \tag{17}$$

Let $X(0) = 0$ and use (14a) (with (17) as a constraint) to compute $\{v_k\}$. Finally carry out a least-squares fit over the nonzero v_k (fixing the other v_k to zero). The trajectory generated by this last estimate $\{v_k\}$ is shown in Figure 1 for

$\lambda = 0.05$, 0.1 and 0.5 and ℓ_1-norm. The associated pulse train is given in Figure 2. Feeding a system consisting of a single integrator gives the searched piecewise constant input signal that should be used in (15) to make the output as in Figure 1. Notice that it is only 10, 9 respective 6 v_k-values that are needed to represent the control input. A known control saturation would also be easy to implement and impossible reference trajectories (as may occur when using splines) are not a problem. Let us also compute the trajectory using a pulse train, piecewise

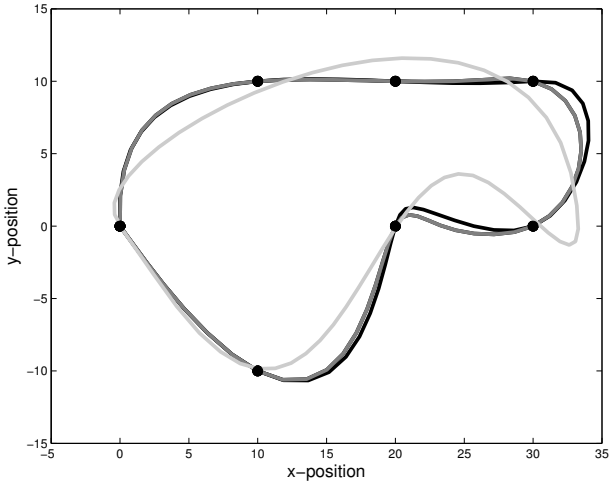

Figure 1: *The computed trajectory (*x*- and* y*-position) of Example 2 shown for $\lambda = 0.05$, 0.1 and 0.5 (black, gray resp. light gray line). Waypoints are shown with filled circles.*

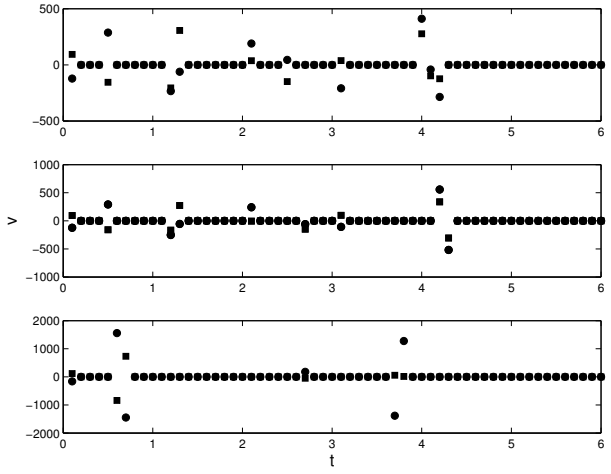

Figure 2: *The pulse train used to generate the trajectory shown in Figure 1. From top to bottom, $\lambda = 0.05$, 0.1 and 0.5. Filled circles and squares are used to symbolize the two dimensional v_k.*

constant and piecewise linear input ($p = 0$, $p = 1$ resp. $p = 2$ in (7)). The result using $\lambda = 0.05$ is shown in Figures 3 and 4. Note that the black line in Figure 1 thus coincides with the gray line in Figure 3.

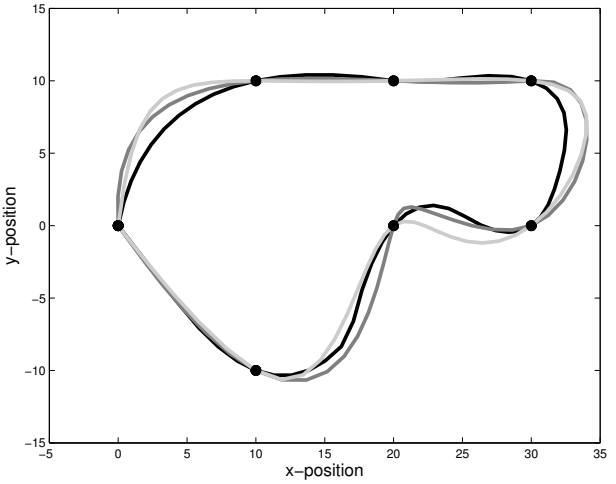

Figure 3: *The computed trajectory (*x- *and* y*-position) of Example 2 shown using a pulse train as an input in* (15) *(black thick line), a piecewise constant input (gray line) and a piecewise linear input (light gray line). Waypoints are shown with filled circles.*

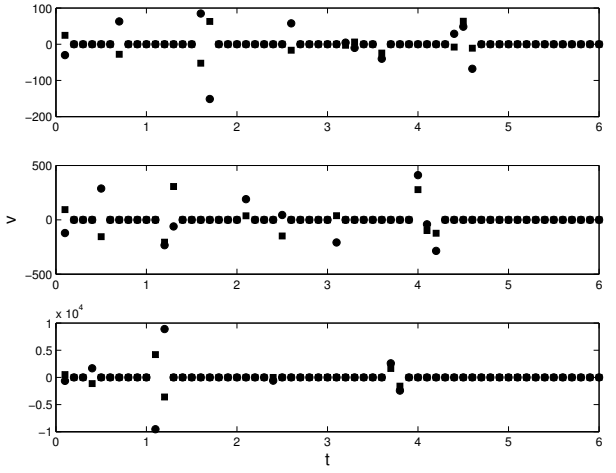

Figure 4: *The pulse train used to generate the trajectory shown in Figure 3. From top to bottom, v_k used to generate a pulse train, a piecewise constant and a piecewise linear input to* (15)*. Filled circles and squares are used to symbolize the two dimensional v_k.*

Example 3: Selecting Time Instances for Control Inputs

Consider the DC-motor model

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 0 & 1 \end{bmatrix} x(t). \end{aligned} \tag{18}$$

Let W contain the reference values

$$\{0,\ 0,\ 0,\ 0,\ 10,\ 10,\ 0,\ 0,\ 10,\ 10,\ 10,\ 10,\ 10\} \tag{19}$$

and let the associated t_k be

$$\{0.75, 2.25, 3, 3.75, 5.25, 7.5, 7.8, 8.25, 9, 10.5, 12, 13.5, 15\}.$$

Extend now (18) by adding an extra state to be able to impose a piecewise constant input $u(t)$. Set the sampling time $T_s = 0.15$ and discretize. The extended discretized model takes the form

$$\begin{aligned} X(kT_s + T_s) &= \begin{bmatrix} 0.8607 & 0 & 0.1393 \\ 0.1393 & 1 & 0.0107 \\ 0 & 0 & 1 \end{bmatrix} X(kT_s) + \begin{bmatrix} 0.1393 \\ 0.0107 \\ 1 \end{bmatrix} v_k \\ y(kT_s) &= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} X(kT_s). \end{aligned} \tag{20}$$

The computed $y(t)$ using $\lambda = 0.5$, $\|v_k\|_2^2 < 40$ and an initial state $x(0) = \begin{bmatrix} 0 & 2 \end{bmatrix}^T$ is shown in Figure 5. The associated pulse train is given in Figure 6.

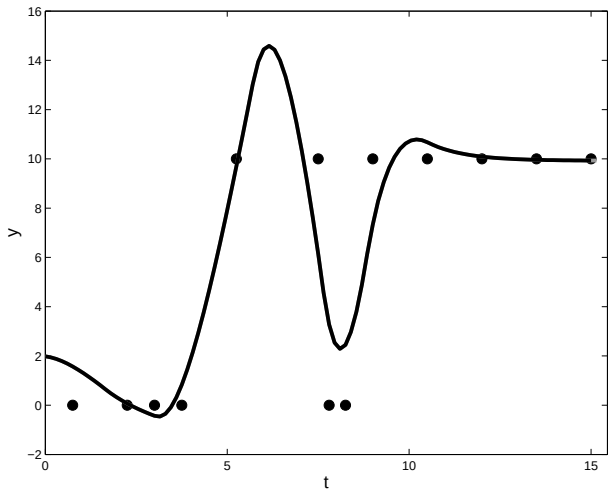

Figure 5: *The computed trajectory of Example 3 shown using black thick line. Waypoints are shown with filled circles.*

Let us now make the example a bit more interesting by applying the technique repeatedly, optimizing over a horizon of mT_s and applying the control for nT_s seconds before re-optimizing. Assume the same waypoints as above. With $m = 14$,

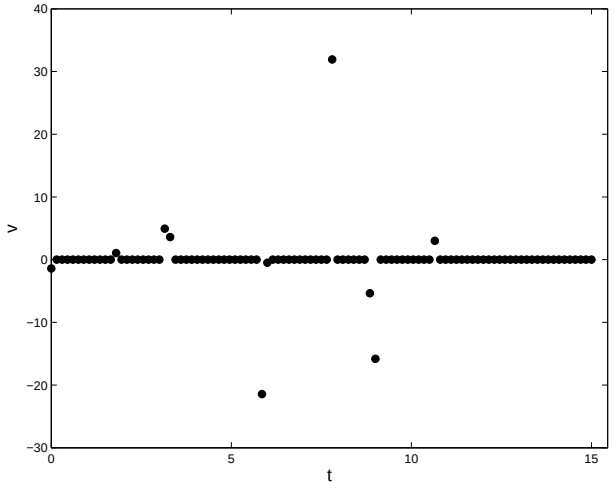

Figure 6: *The pulse train used to generate the trajectory shown in Figure 5.*

$n = 4$ and $\lambda = 1$ the result shown in Figures 7 and 8 was obtained. To recursively compute control signals like this is done in optimal control and MPC. To avoid changing the control signal, if not necessary, like above, may help prevent chattering in MPC (see *e.g.,* Wojsznis et al. (2003) or Naus et al. (2008) for relevant contributions discussing the problem of chattering in applications).

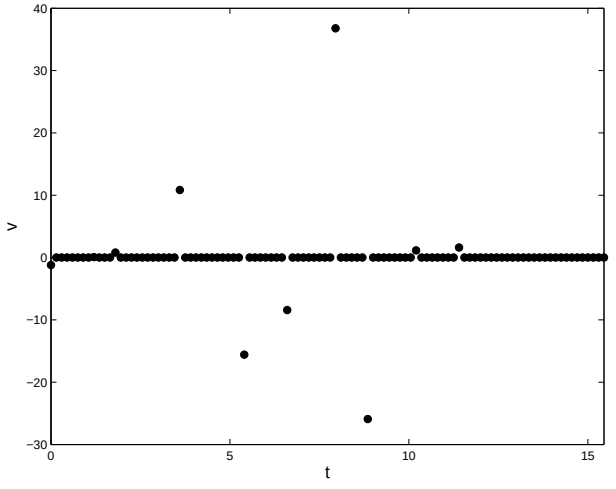

Figure 8: *The pulse train used to generate the trajectory shown in Figure 7.*

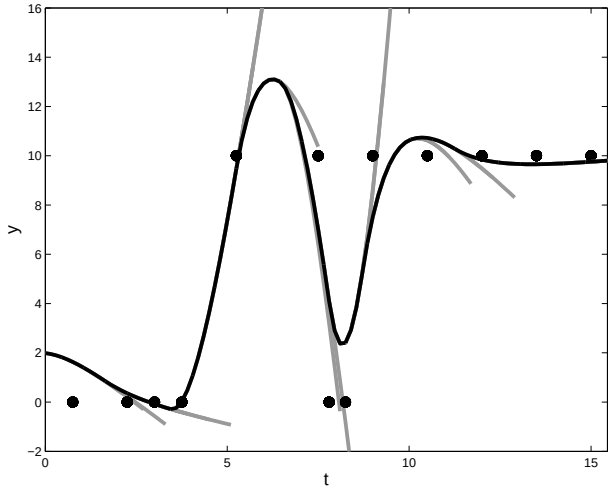

Figure 7: *The computed trajectory of Example 3 shown using black thick line. Waypoints are shown with black filled circles. Gray is used to show the recursively computed full trajectories (the first nT_s seconds of these trajectories were painted black since this was how long a computed control sequence was applied).*

5 Conclusion

A spline representation is often chosen for the reference signal in tracking application. The motivation for this is twofold:

- Using splines a certain degree of smoothness can be guaranteed.
- Splines can be compactly represented.

A spline representation has two disadvantages:

- A spline is a piecewise polynomial function with the different pieces often glued together at the waypoints (see for example Sun et al. (2000) for an attempt to remove this constraint). A more flexible approach would be to not restrict the breakpoints of the spline to the waypoints. This could for example lead to a smoother reference trajectory with a more compact representation.
- A second disadvantage is that it is difficult to guarantee that the generated reference is physically possible for the system to follow.

The proposed method generates a control input which could be fed through the system model to give a spline. We see no reason for computing this spline,

however. The output of the method is rather the sequence $\{v_k\}_{k=1}^N$. The sparse sequence $\{v_k\}_{k=1}^N$ can be stored or communicated using limited resources to the system. The system can then generate an input by integrating the pulse train defined by $\{v_k\}_{k=1}^N$ which takes the system through the specified waypoints. A feedback control is still necessary to reduce the effect of noise and model errors.

The proposed method can hence guarantee a smooth system output. It does not have the problem of generating infeasible reference trajectories. And, since the output sequence is optimized to have few changes but at suitable instances, the representation may be considerably more compact then using splines as a reference.

The proposed method has an optimization formulation. It is convex and the complexity grows linear with N. Constraints, such as control signal saturations, can easily be incorporated. Relations to optimal control and MPC have also been discussed. There is also a relation to Lebesgue sampling, even-triggered sampling and control (see *e.g.,* Åström and Bernhardsson (2003)). This has not been discussed and is seen as future work.

Acknowledgment

Partially supported by the Swedish foundation for strategic research in the center MOVIII and by the Swedish Research Council in the Linnaeus center CADICS. The authors also want to thank Professor Bo Bernhardsson for useful comments on the manuscript.

Bibliography

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509, February 2006.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. D. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, August 2010.

S. Gulati and B. Kuipers. High performance control for graceful motion of an intelligent wheelchair. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3932–3938, Pasadena, CA, USA, May 2008.

F. Gustafsson. *Statistical Sensor Fusion*. Studentlitteratur AB, 2010.

S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, December 2007.

S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. URL `http://control.ee.ethz.ch/~joloef/yalmip.php`.

G. Naus, R. van den Bleek, J. Ploeg, B. Scheepers, R. van de Molengraft, and M. Steinbuch. Explicit MPC design and performance evaluation of an ACC Stop-&-Go. In *Proceedings of the American Control Conference, 2008*, pages 224–229, June 2008.

H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd. Trajectory generation using sum-of-norms regularization. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, USA, December 2010a. To appear.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111, 2010b.

J. Roll. Piecewise linear solution paths with application to direct weight optimization. *Automatica*, 44:2745–2753, 2008.

S. Sun, M. Egerstedt, and C. F. Martin. Control theoretic smoothing splines. *IEEE Transactions on Automatic Control*, 45(12):2271–2279, December 2000.

Y. Teruya, H. Seki, and S. Tadakuma. Driving trajectory generation of electric powered wheelchair using spline curve. In *Proceedings of the 10th IEEE International Workshop on Advanced Motion Control (AMC'08)*, pages 516–519, Trento, Italy, March 2008.

R. Tibsharani. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B (Methodological)*, 58(1):267–288, 1996.

W. Wojsznis, J. Gudaz, T. Blevins, and A. Mehta. Practical approach to tuning MPC. *ISA Transactions*, 42(1):149–162, 2003.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

K. Åström and B. Bernhardsson. Systems with lebesgue sampling. In A. Rantzer and C. Byrnes, editors, *Directions in Mathematical Systems Theory and Optimization*, volume 286 of *Lecture Notes in Control and Information Sciences*, pages 1–13. Springer Berlin / Heidelberg, 2003.

Paper E

Weight Determination by Manifold Regularization

Authors: Henrik Ohlsson and Lennart Ljung

Edited version of the paper:

H. Ohlsson and L. Ljung. Weight determination by manifold regularization. In *Distributed Decision-Making and Control*, Lecture Notes in Control and Information Sciences. Springer Verlag, 2010b. Submitted.

Parts of the theory presented in this paper have also been presented in:

H. Ohlsson, J. Roll, and L. Ljung. Manifold-constrained regressors in system identification. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008b.

H. Ohlsson and L. Ljung. Semi-supervised regression and system identification. In X. Hu, U. Jonsson, B. Wahlberg, and B. Ghosh, editors, *Three Decades of Progress in Control Sciences*. Springer Verlag, December 2010a. To appear.

Weight Determination by Manifold Regularization

Henrik Ohlsson and Lennart Ljung

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden
{ohlsson,ljung}@isy.liu.se

Abstract

A new type of linear kernel smoother is derived and studied. The smoother, referred to as weight determination by manifold regularization, is the solution to a regularized least squares problem. The regularization avoids overfitting and can be used to express prior knowledge of an underlying smooth function. An interesting property of the kernel smoother is that it is well suited for systems govern by the semi-supervised smoothness assumption. Several examples are given to illustrate this property. We also discuss why these types of techniques can have a potential interest for the system identification community.

1 Introduction

A central problem in many scientific areas is to link certain observations to each other and build *models* for how they relate. In loose terms, the problem could be described as relating y to φ in

$$y = f_0(\varphi) \tag{1}$$

where φ is a vector of observed variables, a *regressor* vector, and y is a characteristic of interest, an *output*. In system identification φ could be observed past behavior of a dynamical system, and y the predicted next output.

Observations are often imperfect or noisy, and we are therefore led to consider

$$y = f_0(\varphi) + e, \quad e \sim N(0, \sigma^2). \tag{2}$$

Assume now that a set of observations, $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$, of how f_0 transforms φ is available. $f_0 : \mathcal{R}^{n_\varphi} \to \mathcal{R}$ is itself unknown. The conventional approach within system identification is to make use of a parametric expression $f(\varphi_t, \theta)$, which is hopefully flexible enough to imitate the transformation f_0. $f(\varphi_t, \theta)$ is adjusted to

the observations by choosing θ as

$$\hat{\theta} = \arg\min_{\theta} \sum_{t=1}^{N_e} l(y_t - f(\varphi_t, \theta)), \tag{3}$$

where $l : \mathcal{R} \to \mathcal{R}$ is used to measure how well the model predict the estimation data $\{(\varphi_t, y_t)\}_{t=1}^{N_e}$. l could *e.g.,* be chosen as a norm.

There are a number of parametric expressions and of varying flexibility, and to chose a model structure just flexible enough is crucial when using (3). Let *e.g.,*

$$f(\varphi, \theta) = \theta \tag{4}$$

and $l(\cdot) = (\cdot)^2$ in (3). $\hat{\theta}$, and $f(\varphi, \hat{\theta})$, then become the mean of the observed outputs $\sum_{t=1}^{N_e} y_t / N_e$. This model has of course very good predictive abilities if f_0 is constant but has otherwise rather limited abilities to produce satisfying predictions.

The other extreme, and of particular interest in this chapter, would be to use a parameter for each of the φ_t we will work with. Let $\mathcal{D}$ denote that set of regressors. $\mathcal{D}$ is typically larger than the set of regressors in the estimation set. (If nothing else, we have occasion to compute the response value $f_0(\varphi)$ at new points). Let Θ be a parameter vector of the same size as the number of elements in $\mathcal{D}$:

$$\text{card}(\mathcal{D}) = \dim \Theta \tag{5}$$

We can then associate each parameter value in Θ with a response value $f_0(\varphi_t)$ for any $\varphi_t \in \mathcal{D}$: for convenience denote the elements of Θ by f_t. This particular model hence takes the form

$$f(\varphi_t, \Theta) = f_t, \quad \forall \varphi_t \in \mathcal{D}. \tag{6}$$

Remark 1 (Nonparametric Model). Somewhat miss-leading, a model for which the number of parameters grows with the number of estimation data is called a *nonparametric model*. The model given in (6) is hence a nonparametric model.

2 Supervised, Semi-Supervised and Unsupervised Learning

Before continuing it is useful to introduce the notion of *supervised*, *semi-supervised* and *unsupervised learning*.

The term *supervised learning* is used for algorithms for which the construction of $f(\varphi, \hat{\theta})$ is "supervised" by the measured information in y. In contrast to this, *unsupervised learning* only has the information of the regressors $\{\varphi_t, t = 1, \ldots, N_e\}$. In unsupervised classification, *e.g.,* Kohonen (1995), the classes are constructed by various clustering techniques. *Manifold learning*, *e.g.,* Tenenbaum et al. (2000); Roweis and Saul (2000) deals with unsupervised techniques to construct a manifold in the regressor space that houses the observed regressors.

Semi-supervised algorithms are less common. In semi-supervised algorithms, both (y, φ)-pairs and φs, for which no output has been observed, are used to construct the model $f(\varphi, \theta)$. This is particularly interesting if extra effort is required to obtain y. Thus costly (y, φ)-pairs are supported by less costly regressors to improve the result. It is clear that unsupervised and semi-supervised algorithms are of interest only if the regressors have a pattern that is unknown *a priori*.

Semi-supervised learning is an active area within classification and machine learning (see Chapelle et al. (2006); Zhu (2005) and references therein). The main reason that semi-supervised algorithms are not often seen in regression and system identification may be that it is less clear when only regressors can be of use. We will try to bring some clarity to this through this chapter. Generally it could be said that regression problems having regressors constrained to rather limited regions in the regressor space may be suitable for a semi-supervised regression algorithm. It is also important that regressors are available and comparably "cheap" to get as opposed to the (y, φ)-pairs.

3 Cross Validation and Regularization

Let us now return to the model given in (6). If we let $l(\cdot) = (\cdot)^2$ again and $\mathcal{D} = \{\varphi_1, \ldots, \varphi_{N_e}\}$, the criterion of fit (3) now takes the form

$$\hat{\Theta} = (\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_{N_e}) = \underset{f_1, f_2, \ldots, f_{N_e}}{\arg\min} \sum_{t=1}^{N_e} (y_t - f_t)^2 = (y_1, y_2, \ldots, y_{N_e}). \tag{7}$$

$f(\varphi_t, \hat{\Theta})$ hence "succedes" to perfectly fit to the estimation data. If there was no measurement noise polluting the observations, this would be a good thing. However, with measurement noise present, obtaining a perfect fit is not desirable and termed *overfitting*. Overfitting is a problem for flexible models and to chose a model structure just flexible enough to imitate f_0 (and not flexible enough to being able to imitate the noise) would be ideal.

There are a number of approaches to find what is "just flexible enough". Most approaches can be seen belonging to either *cross validation* or *regularization*.

In *cross validation*, a new data set $\{(\varphi_t, y_t)\}_{t=1}^{N_v}$ is utilized to avoid overfitting. The data set $\{(\varphi_t, y_t)\}_{t=1}^{N_v}$ is denoted the *validation data set*. Since measurement noise e of the validation data set is impossible to predict, the best possible would be to perfectly predict the outcome of the deterministic part of (2) *i.e.,* $f_0(\varphi)$. Therefore, for a number of candidate structures $f_i(\varphi, \hat{\theta}_i)$, $i = 1, \ldots, m$ ($\hat{\theta}$ found using (3)), a model is chosen by

$$\underset{f_i(\varphi, \hat{\theta}_i), i=1,\ldots,m}{\arg\min} \sum_{t=1}^{N_v} l\Big(y_t - f_i(\varphi_t, \hat{\theta}_i)\Big). \tag{8}$$

To evaluate (8) we need to compute predictions for f_0 at regressors not included in the estimation data set *i.e.,* $f_i(\varphi_t, \hat{\theta}_i)$, $t = 1, \ldots, N_v$. For the model

$f(\varphi_t, \theta) = \theta$, see (4), this is straight forward. $f(\varphi_t, \hat{\theta})$, $t = 1, \ldots, N_\mathrm{v}$ are simply equal to $\sum_{t=1}^{N_\mathrm{e}} y_t / N_\mathrm{e}$. For the model given in (6), however, it is not trivial and we will discuss this in the next section.

In *regularization*, a cost on flexibility is added to the criterion of fit. $f(\varphi_t, \theta)$ is now adjusted to the observations by choosing θ as

$$\hat{\theta} = \arg\min_{\theta} \sum_{t=1}^{N_\mathrm{e}} l\Big(y_t - f(\varphi_t, \theta) big) + \lambda J(\theta, \varphi_t), \tag{9}$$

rather than using (3). $J(\theta, \varphi_t)$ serves as a cost on flexibility and is often used to penalize non-smooth estimates. λ is seen as a design parameter and regulates the trade-off between fit to the estimation data and smoothness. Choosing the "just flexible enough" model structure is now transformed to choosing the right λ-value.

For the model proposed in (6), a suitable regularizer is

$$J(\Theta, \varphi_t) = \sum_{t=1}^{N_\mathrm{e}} \left(f_t - \sum_{s=1}^{N_\mathrm{e}} \frac{k(\varphi_t, \varphi_s) f_s}{\sum_{r=1}^{N_\mathrm{e}} k(\varphi_t, \varphi_r)} \right)^2, \tag{10}$$

where $k : \mathcal{R}^{n_\varphi} \times \mathcal{R}^{n_\varphi} \to \mathcal{R}$ is a *kernel*. The regularizer (10) makes sure that close-by regressors are transformed in a similar way by $f(\varphi, \Theta)$ and therefore reassures smoothness. This remedies the overfitting problem that (6) was previously suffering of.

There are a number of different kernels that could be of interest to use in (10). Some of the most interesting kernels are the *squared exponential*, *KNN* and *LLE* kernel. The details of these kernels are outlined in Appendix A.1.

4 Generalization

For most practical purposes it is not enough to find a model $f(\varphi, \theta)$ that well imitates f_0 at $\{(\varphi_t, y_t)\}_{t=1}^{N_\mathrm{e}}$. Generalization to non-observed data is often of more importance. This is denoted the model's ability to *generalize* to unseen data.

Let φ_* be an unseen regressor, *i.e.*, $\varphi_* \neq \varphi_t, t = 1, \ldots, N_\mathrm{e},$. For the simple model (4),

$$f(\varphi, \hat{\theta}) = \sum_{t=1}^{N_\mathrm{e}} y_t / N_\mathrm{e}, \tag{11}$$

generalization is trivial since the prediction does not depend on the regressor. The estimate for $f(\varphi_*)$, $\varphi_* \neq \varphi_t, t = 1, \ldots, N_\mathrm{e}$, is simply taken as $\sum_{t=1}^{N_\mathrm{e}} y_t / N_\mathrm{e}$.

For the model (6),

$$f(\varphi_t, \hat{\Theta}) = \hat{f}_t, \quad t = 1, \ldots, N_\mathrm{e}, \tag{12}$$

with

$$\hat{\Theta} = \underset{f_1,f_2,\ldots,f_{N_e}}{\arg\min} \sum_{t=1}^{N_e}(y_t - f_t)^2 + \lambda \sum_{t=1}^{N_e}\left(f_t - \sum_{s=1}^{N_e}\frac{k(\varphi_t,\varphi_s)f_s}{\sum_{r=1}^{N_e}k(\varphi_t,\varphi_r)}\right)^2, \tag{13}$$

generalization is a bit more involved. The most natural way is to introduce a new parameter f_* for the estimate of $f_0(\varphi_*)$ and let the smoothness implied by the regularization give an estimate

$$f(\varphi_*, \hat{\Theta}) = \hat{f}_* \tag{14}$$

with

$$\hat{\Theta} = \underset{f_1,f_2,\ldots,f_{N_e},f_*}{\arg\min} \sum_{t=1}^{N_e}(y_t - f_t)^2 + \lambda \sum_{\varphi_t\in\mathcal{D}}\left(f_t - \sum_{\varphi_s\in\mathcal{D}}\frac{k(\varphi_t,\varphi_s)f_s}{\sum_{\varphi_r\in\mathcal{D}}k(\varphi_t,\varphi_r)}\right)^2. \tag{15}$$

$\mathcal{D}$ now contains both the estimation data and φ_*, *i.e.*, $\mathcal{D} = \{\varphi_1, \varphi_2, \ldots, \varphi_{N_e}, \varphi_*\}$. Since (15) is quadratic in the optimization variables, an explicit solution can be computed. Introduce first the notation

$$\mathbf{J} \triangleq [I_{N_e\times N_e}\ 0_{N_e\times 1}],\ \mathbf{y} \triangleq [y_1\ y_2 \ldots y_{N_e}]^T, \tag{16a}$$

$$\hat{\mathbf{f}} \triangleq [\hat{f}_1\ \hat{f}_2 \ldots \hat{f}_{N_e}\ \hat{f}_*]^T,\ \bar{k}(\varphi_t,\varphi_s) \triangleq \frac{k(\varphi_t,\varphi_s)}{\sum_{\varphi_r\in\mathcal{D}}k(\varphi_t,\varphi_r)}, \tag{16b}$$

$$\mathbf{K} \triangleq \begin{bmatrix} \bar{k}(\varphi_1,\varphi_1) & \bar{k}(\varphi_1,\varphi_2) & \ldots & \bar{k}(\varphi_1,\varphi_{N_e}) & \bar{k}(\varphi_1,\varphi_*) \\ \bar{k}(\varphi_2,\varphi_1) & \bar{k}(\varphi_2,\varphi_2) & & \bar{k}(\varphi_2,\varphi_{N_e}) & \bar{k}(\varphi_2,\varphi_*) \\ \vdots & & \ddots & & \vdots \\ \bar{k}(\varphi_{N_e},\varphi_1) & \bar{k}(\varphi_{N_e},\varphi_2) & \ldots & \bar{k}(\varphi_{N_e},\varphi_{N_e}) & \bar{k}(\varphi_{N_e},\varphi_*) \\ \bar{k}(\varphi_*,\varphi_1) & \bar{k}(\varphi_*,\varphi_2) & \ldots & \bar{k}(\varphi_*,\varphi_{N_e}) & \bar{k}(\varphi_*,\varphi_*) \end{bmatrix}. \tag{16c}$$

(15) can then be written as

$$(\mathbf{y} - \mathbf{J}\hat{\mathbf{f}})^T(\mathbf{y} - \mathbf{J}\hat{\mathbf{f}}) + \lambda(\hat{\mathbf{f}} - \mathbf{K}\hat{\mathbf{f}})^T(\hat{\mathbf{f}} - \mathbf{K}\hat{\mathbf{f}}) \tag{17}$$

which expands into

$$\hat{\mathbf{f}}^T\left(\lambda(\mathbf{I} - \mathbf{K})^T(\mathbf{I} - \mathbf{K}) + \mathbf{J}^T\mathbf{J}\right)\hat{\mathbf{f}} + 2\hat{\mathbf{f}}^T\mathbf{J}^T\mathbf{y} + \mathbf{y}^T\mathbf{y}. \tag{18}$$

Setting the derivative with respect to $\hat{\mathbf{f}}$ to zero and solving gives

$$\hat{f}_* = \mathbf{e}_*\left(\lambda(\mathbf{I} - \mathbf{K})^T(\mathbf{I} - \mathbf{K}) + \mathbf{J}^T\mathbf{J}\right)^{-1}\mathbf{J}^T\mathbf{y}, \quad \mathbf{e}_* \triangleq [0_{1\times N_e}1]. \tag{19}$$

It is straight forward to do the generalization for more than one unobserved regressor at a time. $\mathcal{D}$ used in (15) then indexes all regressors, the N_e estimation regressors and the regressors for which we seek an estimate of the function-value but have not observed the output. We will refer to the method outlined in (19) to as *Weight Determination by Manifold Regularization* (WDMR, Ohlsson et al. (2008), see also Ohlsson (2008); Ohlsson and Ljung (2010a)). The reason for the name will become clear later.

Proposition 1 (Linear Kernel Smoother). *The estimate given in* (19) *can be rewritten in the form*

$$f(\varphi_*, \hat{\Theta}) = \sum_{t=1}^{N_e} w_t y_t \tag{20}$$

and is therefore a linear estimator, since it is linear in the estimation outputs. The estimate given in (19) *is also a kernel smoother since it is constructed using kernels. These two combined makes WDMR a linear kernel smother (see* e.g., *Hastie et al. (2001), p. 129). For WDMR* w_t *is given by*

$$w_t = \mathbf{e}_* \left(\lambda (\mathbf{I} - \mathbf{K})^T (\mathbf{I} - \mathbf{K}) + \mathbf{J}^T \mathbf{J} \right)^{-1} \mathbf{J}^T \mathbf{y} \mathbf{e}_t^T \tag{21}$$

with $\mathbf{e}_t = [0_{1\times t-1}\ 1\ 0_{1\times N_e - t}]$. *The expression for constructing the weights* w *in* (20) *is referred to as the equivalent kernel in the literature (see* e.g., *Hastie et al. (2001) p. 170).*

Notice that the resulting estimates coming from estimating the function-value of unobserved regressors one-by-one and all at the same time will not be the same. This is a property of semi-supervised regression approaches. The regularization will make sure that the estimated f_t varies smoothly on regressor-dense regions. We will return to this property later and discuss when it can be useful.

5 WDMR and the Nadaraya-Watson Smoother

In the linear kernel smoother WDMR the kernel was used to provide a smoothness prior. A kernel can also "direct" be used to obtain an estimate for $f_0(\varphi_*)$ using

$$f(\varphi_*) = \sum_{t=1}^{N_e} \frac{k(\varphi_*, \varphi_t) y_t}{\sum_{r=1}^{N_e} k(\varphi_*, \varphi_r)}, \tag{22}$$

which also is a linear kernel smoother. This is referred to as the *Nadaraya-Watson smoother* or *estimator* (Nadaraya, 1964; Watson, 1964). It may seem a bit overcomplicated to, as in WDMR, use a kernel as *e.g.,* smoothness prior but in the end anyway end up with a linear kernel smoother. What is achieved by using a kernel in WDMR compared to using the kernel direct in the Nadaraya-Watson smoother as in (22)?

Remark 2. Note that the Nadaraya-Watson smoother weight together noisy observations $\{y_t\}_{t=1}^{N_e}$ to obtain an estimate $f(\varphi_*)$. To reduce the influence of noise, y_t in (22) could itself be replaced by an estimate of $f_0(\varphi_t)$ by using (22) a second time. *i.e.,*

$$f(\varphi_t) = \frac{1}{\sum_{r=1}^{N_e} k(\varphi_t, \varphi_r) + k(\varphi_t, \varphi_*)} \Big(\sum_{s=1}^{N_e} k(\varphi_t, \varphi_s) y_s + k(\varphi_t, \varphi_*) f(\varphi_*) \Big). \tag{23}$$

If all noisy observations are replaced, we obtain the system of equations

$$f(\varphi_t) = \sum_{\varphi_s \in \mathcal{D}} \frac{k(\varphi_t, \varphi_s) f(\varphi_s)}{\sum_{\varphi_r \in \mathcal{D}} k(\varphi_t, \varphi_r)}, \quad \forall \varphi_t \in \mathcal{D},\ \mathcal{D} = \{\varphi_1, \varphi_2, \ldots, \varphi_{N_e}, \varphi_*\}, \tag{24}$$

which takes a familiar form (see the regularization in (15)). The regularization in WDMR expresses the desire to obtain an estimate satisfying this system of equations.

Remark 3. The resulting estimates coming from estimating the function-value at unobserved regressors one-by-one and all at the same time will not be the same for WDMR. This is a property of *semi-supervised* regression approaches. The regularization will make sure that the estimated f_ts vary smoothly on regressor-dense regions. The Nadaraya-Watson smoother is not a semi-supervised approach and estimating one function-value at a time or all at the same time would be the same.

To start examine the advantages and disadvantages of WDMR compared to the Nadaraya-Watson smoother, let us look at an example.

Example 1: Nadaraya-Watson Smoother Versus WDMR

Let us now consider a standard test example from Narendra and Li (1996), "the Narendra-Li example":

$$x_{t+1} = \left(\frac{x_t}{1 + x_t^2} + 1\right) \sin(z_t) \tag{25a}$$

$$z_{t+1} = z_t \cos(z_t) + x_t \exp\left(-\frac{x_t^2 + z_t^2}{8}\right) + \frac{u_t^3}{1 + u_t^2 + 0.5\cos(x_t + z_t)} \tag{25b}$$

$$y_t = \frac{x_t}{1 + 0.5\sin(z_t)} + \frac{z_t}{1 + 0.5\sin(x_t)} + e_t \tag{25c}$$

This dynamical system was simulated with 2000 samples using a random binary input, giving input output data $\{y_t, u_t, t = 1, \ldots, 2000\}$. A separate set of 200 validation data, see Figure 1, were also generated with a sinusoidal input. The chosen regression vector was

$$\varphi_t = \begin{bmatrix} y_{t-1} & y_{t-2} & y_{t-3} & u_{t-1} & u_{t-2} & u_{t-3} \end{bmatrix}^T. \tag{26}$$

Let us use the squared exponential kernel (see Appendix A.1) and apply the Nadaraya-Watson smoother and WDMR ($\lambda = 0.0001$ was used in (21)) to estimate the function-values at the validation regressors. The result is given in Table 1. A length scale (see Appendix A.1 for the definition of length scale) of 0.6 and 0.7 gave the best performing Nadaraya-Watson smoother respective WDMR. The table also give the fit for a neural network (a single layer sigmoid network with 23 units in the System Identification Toolbox (Ljung, 2007) gave the best performance) and the prediction given by guessing that the next f_0-value will equal the previous observation.

The direct usage of the squared exponential kernel in the Nadaraya-Watson smoother is doing very well compare to the neural network and guessing that the next f_0-value will be equal the previous observation. However, even better

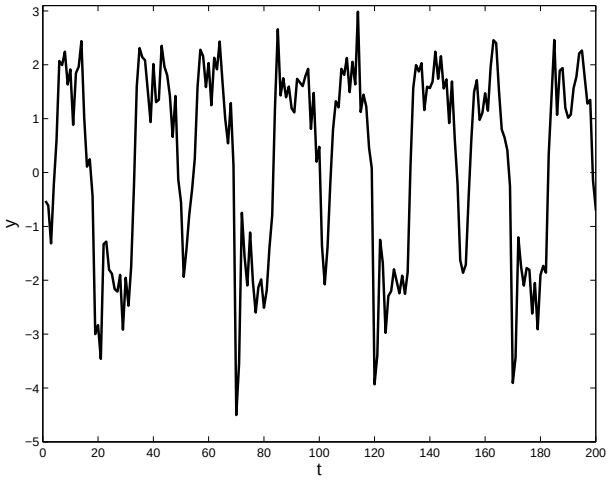

Figure 1: *Validation data for the Narendra-Li example.*

does WDMR. As mentioned earlier (see Remark 2), WDMR has a hierarchical scheme for denoising the observations. One may therefore wonder if enlarging the bandwidth/length scale in the Nadaraya-Watson smoother would have the same denoising effect. Figure 2 shows that it is not that easy and that enlarging the bandwidth/length scale does not help the Nadaraya-Watson smoother.

Table 1: *Mean fit over 20 noise and input realization for the Nadaraya-Watson smoother and WDMR using a squared exponential kernel, a neural network and the estimate obtained by simply taking the previous output as an estimate for the next function value.*

Algorithm	Mean fit (%)
Nadaraya-Watson (squared exponential, $\ell = 0.6$)	68
WDMR (squared exponential, $\ell = 0.7$, $\lambda = 10^{-4}$)	71
Neural network (23 units)	66
Last measurement	47

It is also interesting to examine what happens if the measurement noise changes. Table 2 gives the result from an experiment where the noise level was decreased in three steps. We see that as the noise level decrease the difference in performance between the Nadaraya-Watson smoother and WDMR disappears.

So far we have only applied WDMR to a batch of data. We claimed earlier that applying WDMR to a batch of data is not the same as applying it to the regressors one-by-one. And unfortunately, the positive result seen for the batch tends to disappear when this is done. WDMR and the Nadaraya-Watson smoother then give similar performance. There are possibly many reasons for this. One possible reason is the following. In the batch setting, the regularization in WDMR makes sure that the estimate varies smoothly over regions of regressors. So called

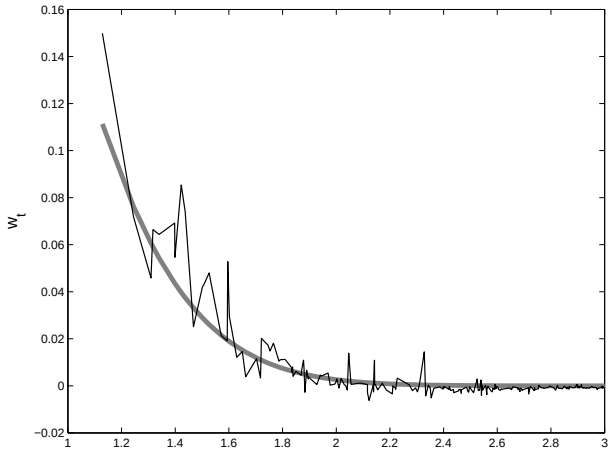

Figure 2: *w_t of* (21) *(thin black line) plotted as a function of $|\varphi_i - \varphi_t|$, $i = 1 \ldots, N_e$ and φ_t being one of the validation regressors. Thick gray line shows the corresponding weights of the Nadaraya-Watson smoother.*

Table 2: *The Nadaraya-Watson smoother and WDMR's performance for three different noise levels. Both algorithms used a squared exponential kernel and were tuned for each of the noise levels for optimal performance. $l = 0.8, 0.6, 0.6, 0.5$ were used in the Nadaraya-Watson estimator and $(l, \lambda) = (0.8,\ 10^{-4})$, $(0.7,\ 10^{-4})$, $(0.7,\ 0.8 \cdot 10^{-4})$, $(0.7,\ 0.5 \cdot 10^{-4})$ in WDMR.*

Algorithm	Fit $\sigma^2 = 0.5$	Fit $\sigma^2 = 0.1$	Fit $\sigma^2 = 0.05$	Fit $\sigma^2 = 0.01$
Nadaraya-Watson	56	69	72	74
WDMR	60	71	73	74

boundary effects have however been observed. This means that the estimates for regressors at the end of dense regions often are worse than estimates for regressors surrounded by many other regressors. Boundary effects are a known issue of kernel smoothers, see *e.g.,* Hastie et al. (2001, p. 168). This supports that batch would do better than one-by-one.

This unfortunately means that WDMR is less interesting for other than batch and nonlinear FIR models. We will in the subsequent only discuss a batch approach.

6 The Semi-Supervised Smoothness Assumption

WDMR does not only have good denoising properties, it can also provide desirable properties when it comes to problems having regressors confined to limited regions *e.g.,* manifolds, in the regressor space. Let us illustrate this by a pictorial example.

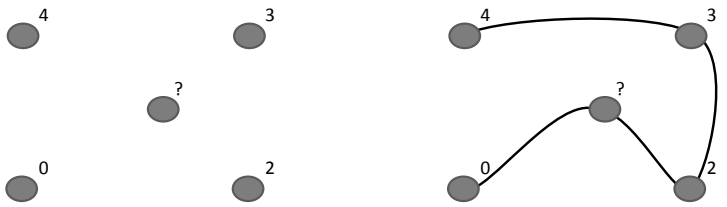

Figure 3: *The left side shows 5 regressors, four with measured outputs and one with unknown output. Desiring an estimate of the function value at the regressor with a "?", we could simply weight together the two closest regressors' outputs and get 2.5. Say now that the process that generated our regressors, traced out the path shown in the right part of the figure. Would we still guess 2.5?*

Consider the five regressors shown in the left of Figure 3. For four of the regressors the output has been observed and their outputs are written out next to them. One of the regressors' output is unknown. To estimate the function value at that regressor, we could use the Nadaraya-Watson smoother and compute the average of the two closest regressors' outputs, which would give an estimate of 2.5. Let us now add the information that the regressors and the outputs were sampled from an in time continuous process and that the value of the regressor was evolving along the curve shown in the right part of Figure 3. Knowing this, a better estimate of the function value would probably be 1. The knowledge of that the regressors are restricted to a certain region in the regressor space can hence make us reconsider our estimation strategy.

We are in regression interested in finding estimates for the conditional distribution $p(f|\varphi)$. For the regressors without observed output to be useful, it is required that the regressor distribution $p(\varphi)$ brings information concerning the conditional $p(f|\varphi)$. We saw from the pictorial example that one situation for which this is the case is when we make the assumption that the sought function value changes continuously along high-density areas in the regressor space. This assumption is referred to as the *semi-supervised smoothness assumption* (Chapelle et al., 2006):

Assumption E.1 (Semi-Supervised Smoothness). If two regressors φ_1, φ_2 in a high-density region are close, then so should $f_0(\varphi_1)$ and $f_0(\varphi_2)$ be.

"High density region" is a somewhat loose term: In many cases it corresponds to a manifold in the regressor space, such that the regressors for the application in question are confined to this manifold. That two regressors are "close" then means that the distance between them along the manifold (the geodesic distance) is small.

In classification, this smoothness assumption is interpreted as that the class labels should be the same in the high-density regions. In regression, we interpret this as a slowly varying function along high-density regions. Note that in regression, it is common to assume that the function value varies smoothly in the regressor space; the semi-supervised smoothness assumption is less conservative since it only assumes smoothness in the high-density regions in the regressor space. Two regressors could be close in the regressor space metric, but far apart along the high density region (the manifold): think of the region being a spiral in the regressor space.

One may discuss how common it is in system identification that the regressors are constrained to a manifold. The input signal part of the regression vector should according to identification theory be "persistently exciting" which is precisely the opposite of being constrained. However, in many biological applications and in DAE (Differential Algebraic Equation) modeling such structural constraints are frequently occurring.

6.1 A Comparison Between the Nadaraya-Watson Smoother and WDMR Using the KNN Kernel

To illustrate the advantage of WDMR under the semi-supervised smoothness assumption, we continue to discuss the pictorial example previously discussed. We now add 5 regressors with unobserved output to the 5 previously considered. Hence we have 10 regressors, 4 with observed outputs and 6 with unobserved outputs, and we desire an estimate of the output marked with a question mark in Figure 4. The left part of Figure 4 shows how the Nadaraya-Watson smoother solves the estimation problem if the KNN kernel (see Appendix A.1) is used. The kernel will cause the searched function value to be similar to the observed outputs of the K closest regressors. In the right part of Figure 4, WDMR with the KNN kernel is used. This kernel grants estimates of the K closest regressors (observed or unobserved output) to be similar. Since the closest regressors, to the regressor for which we search the function value, are unobserved, information is propagated from the observed regressors towards the one for which we search a function value estimate along the chain of unobserved regressors. The shaded regions in both the left and right part of the figure symbolize the way information is propagated using the Nadaraya-Watson smoother and WDMR. In the left part of the figure we will therefore obtain an estimate equal to 2.5 while in the right we get an estimate equal to 1.

The ability of WDMR to account for manifolds in the regressor space and the semi-supervised smoothness assumption is a rather unique property of a kernel smoother and the reason for the name *Weight Determination by Manifold Regularization*.

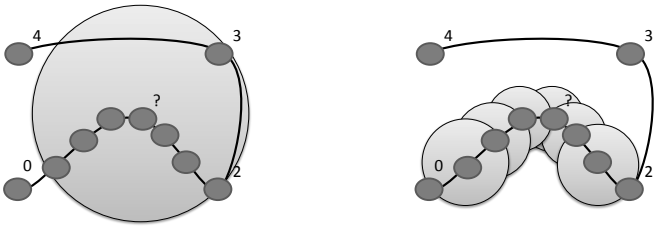

Figure 4: *An illustration of the difference of using the Nadaraya-Watson smoother (left part of the figure) and WDMR (right part of the figure) with the KNN kernel.*

7 Related Approaches

Semi-supervised learning has been around since the 1970s (some earlier attempts exist). *Fisher's linear discriminant rule* was then discussed under the assumption that each of the class conditional densities was Gaussian. *Expectation maximization* was applied using both regressor-output-pairs and regressors to find the parameters of the Gaussian densities (Hosmer, 1973). During the 1990s the interest for semi-supervised learning increased, mainly due to its application to text classification, see *e.g.,* Nigam et al. (1998). The first usage of the word *semi-supervised* learning, as it is used today, was not until 1992 (Merz et al., 1992).

The boost in the area of manifold learning in the 1990s brought with it a number of semi-supervised methods. *Semi-supervised manifold learning* is a type of semi-supervised learning in which the map found by an unsupervised manifold learning algorithm is restricted by giving a number of regressor-output-pairs as examples for how that map should be. Most of the algorithms are extensions of unsupervised manifold learning algorithms, see among others Belkin et al. (2006); Yang et al. (2006); Navaratnam et al. (2007); de Ridder et al. (2003); de Ridder and Duin (2002); Ohlsson et al. (2008); Zhao and Zhang (2009). Another interesting contribution is the developments by Rahimi in Rahimi et al. (2007). A time series of regressors, some with measured outputs and some not, are considered there. The series of estimates best fitting the given outputs and at the same time satisfying some temporal smoothness assumption is then computed.

Most of the references above are to semi-supervised classification algorithms. They are however relevant since most semi-supervised classification methods can, with minor modifications, be applied to regression problems. The modification or the application to regression problems are however almost never discussed or exemplified. For more historical notes on semi-supervised learning, see Chapelle et al. (2006).

Similar methods to WDMR has also been discussed previously, see *e.g.,* Goldberg and Zhu (2006); Ohlsson et al. (2007); Yang et al. (2006); Bengio et al. (2006); Belkin et al. (2006); Wang and Zhang (2008). Yang et al. (2006) discusses manifold learning and construct a semi-supervised version of the manifold learning

technique *Locally Linear Embedding* (LLE, Roweis and Saul (2000)) which coincides with a particular choice of kernel in (15). Combining LLE with system identification was also discussed in Ohlsson et al. (2007). Goldberg and Zhu (2006) studies graph based semi-supervised methods for classification and derives a similar objective function as (15). Bengio et al. (2006); Wang and Zhang (2008) discuss a classification method called *label propagation* which is an iterative approach converging to (15). In Belkin et al. (2006), support vector machines is extended to work under the semi-supervised smoothness assumption. There is also a huge literature on kernel smoothers, see *e.g.*, Hastie et al. (2001).

8 Examples

We give in the following two examples of regression problem for which the semi-supervised smoothness assumption is motivated.

8.1 fMRI

functional Magnetic Resonance Imaging, fMRI is a technique to measure brain activity. The fMRI measurements give a measure of the degree of oxygenation in the blood, it measures the *Blood Oxygenation Level Dependent* (BOLD) response. The degree of oxygenation reflects the neural activity in the brain and fMRI is therefore an indirect measure of brain activity.

Measurements of brain activity can with fMRI be acquired as often as once a second and are given as an array, each element giving a scalar measure of the average activity in a small volume element of the brain. These volume elements are commonly called *voxels* (short for *volume pixel*) and they can be as small as one cubic millimeter. The fMRI measurements are heavily affected by noise.

In this example, we consider measurements from an $8 \times 8 \times 2$ array covering parts of the visual cortex gathered with a sampling period of 2 seconds. To remove noise, data was prefiltered by applying a spatial and temporal filter with a squared exponential kernel. The filtered fMRI measurements at each time t were vectorized into the regression vector φ_t. fMRI data was acquired during 240 seconds (giving 120 samples, since the sampling period was 2 seconds) from a subject that was instructed to look away from a flashing checkerboard covering 30% of the field of view. The flashing checkerboard moved around and caused the subject to look to the left, right, up and down. The direction in which the person was looking was seen as the output. The output was chosen to 0 when the subject was looking to the right, $\pi/2$ when looking up, π when looking to the left and $-\pi/2$ when looking down.

The direction in which the person was looking is described by its angle, a scalar. The fMRI data should hence be constrained to a one-dimensional closed manifold residing in the 128 dimensional regressor space (since the regressors can be parameterized by the angle). If we assume that the semi-supervised smoothness assumption holds, WDMR therefore seems like a good choice.

The 120 regressors with observed output were separated into two sets, a training set consisting of 80 regressors and a test set consisting of 40 regressors. The training set was further divided into an estimation set and a validation set, both of the same size. The estimation set and the regressors of the validation set were used in WDMR. The estimated outputs of the validation regressors were compared to the measured outputs and used to determine the design parameters. λ in (15) was chosen as 0.8 and K (using the kernel determined by LLE, see Appendix A.1) as 6. The tuned WDMR regression algorithm was then used to predict the direction in which the person was looking. The result from applying WDMR to the 40 regressors of the test set is shown in Figure 5.

The result is satisfactory but it is not clear to what extent the one-dimensional manifold has been found. The number of regressors with unobserved output used are rather low and it is therefore not surprising that the Nadaraya-Watson smoother with the KNN kernel can be shown to do almost as good as WDMR in this example. One would expect that adding more regressors with unobserved output would improve the result obtained by WDMR. The estimates of the Nadaraya-Watson smoother would however stay unchanged since the Nadaraya-Watson smoother is a supervised method and therefore not affected by regressors with unobserved output.

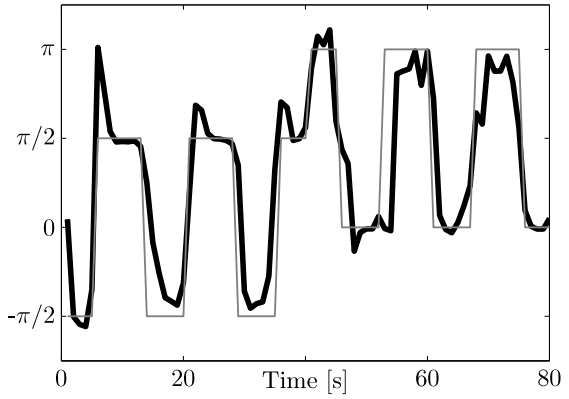

Figure 5: *WDMR applied to brain activity measurements (fMRI) of the visual cortex in order to tell in what direction the subject in the MR scanner was looking. Thin gray line shows the direction in which the subject was looking and thick black line, the estimated direction by WDMR.*

8.2 Climate Reconstruction

There exist a number of climate recorders in nature from which the past temperature can be extracted. However, only a few natural archives are able to record climate fluctuations with high enough resolution so that the seasonal variations can be reconstructed. One such archive is a bivalve shell. The chemical composition of a shell of a bivalve depends on a number of chemical and physical parameters of the water in which the shell was composed. Of these parameters,

the water temperature is probably the most important one. It should therefore be possible to estimate the water temperature for the time the shell was built, from measurements of the shell's chemical composition. This would *e.g.*, give climatologists the ability to estimate past water temperatures by analyzing ancient shells.

In this example, we used 10 shells grown in Belgium. Since the temperature in the water had been monitored for these shells, this data set provides excellent means to test the ability to predict water temperature from chemical composition measurements. For these shells, the chemical composition measurements had been taken along the growth axis of the shells and paired up with temperature measurements. Between 30 and 52 measurement were provided from each shell, corresponding to a time period of a couple of months. The 10 shells were divided into an estimation set and a validation set. The estimation set consisted of 6 shells (a total of 238 regressors with observed output) grown in Terneuzen in Belgium. Measurements from five of these shells are shown in Figure 6. The figure shows measurements of the relative concentrations of Sr/Ca, Mg/Ca and Ba/Ca (Pb/Ca is also measured but not shown in the figure). The line shown between measurements connects the measurements coming from a shell and gives the chronological order of the measurements (two in time following measurements are connected by a line).

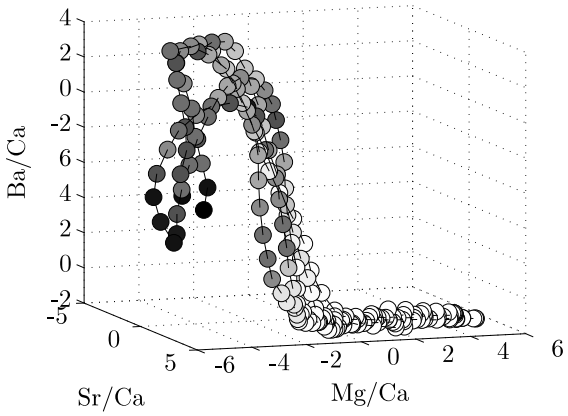

Figure 6: *A plot of the Sr/Ca, Mg/Ca and Ba/Ca concentration ratio measurements from five shells. Lines connects measurements (ordered chronologically) coming from the same shell. The temperatures associated with the measurements were color coded and are shown as different gray scales on the measurement points.*

As seen in the figure, measurements are highly restricted to a small region in the measurement space. Also, the water temperature (gray level coded in Figure 6) varies smoothly in the high-density regions. This together with that it is a biological process generating data, motivates the semi-supervised smoothness assumption when trying to estimate water temperature from chemical composition measurements (4-dimensional regressors).

The four shells in the validation set came from four different sites (Terneuzen, Breskens, Ossenisse, Knokke) and from different time periods. The estimated temperatures for the validation data obtained by using WDMR with the kernel determined by LLE (see Appendix A.1) are shown in Figure 7. For comparison purpose, it could be mentioned that the Nadaraya-Watson smoother using the LLE kernel had a *Mean Absolute Error* (MAE) nearly twice as high as WDMR.

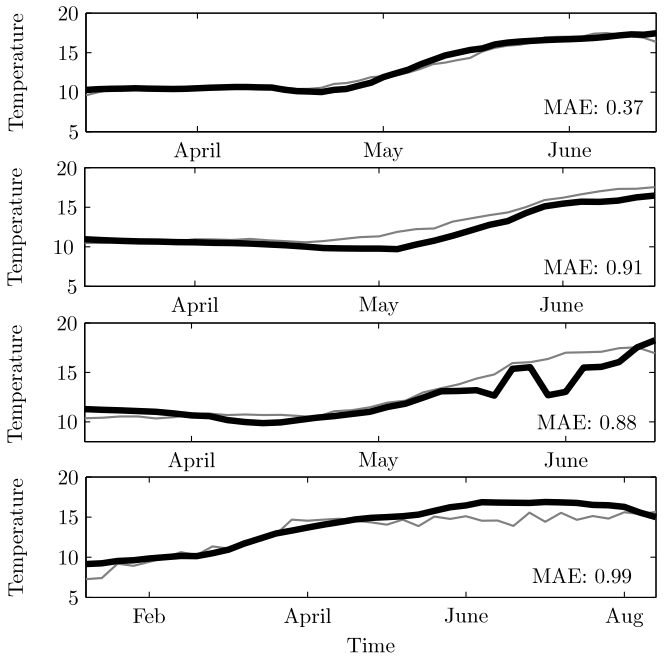

Figure 7: *Water temperature estimations using WDMR for validation data (thick line) and measured temperature (thin line). From top to bottom figure, shells from: Terneuzen, Breskens, Ossenisse, Knokke.*

A more detailed discussion of this exampled is presented in Bauwens et al. (2009). The data sets used were provided by Vander Putten and colleagues (Vander Putten et al., 1999) and Gillikin and colleagues (Gillikin et al., 2006a,b).

9 Conclusion

This chapter presents and discusses a novel linear kernel smoother, weight determination by manifold regularization. The regression method is of particular interest when regressors are confined to limited regions in the regressor space and under the semi-supervised smoothness assumption. Examples of this type of problems were given.

Acknowledgment

This work was supported by the Strategic Research Center MOVIII, funded by the Swedish Foundation for Strategic Research, SSF, and CADICS, a Linnaeus center funded by the Swedish Research Council.

A Appendix

A.1 Kernels

This section presents kernels referred to in the chapter. The convention that $k(\varphi_i, \varphi_j) = 0$ if $i = j$ is always used. See Chapter 4 in Rasmussen and Williams (2005) for more on kernels.

The KNN Kernel

Define the *K-nearest neighbor kernel* as

$$k(\varphi_i, \varphi_j) \triangleq \begin{cases} \frac{1}{K}, & \text{if } \varphi_j \text{ is one of the } K \text{ closest neighbors,} \\ 0, & \text{otherwise.} \end{cases} \tag{27}$$

The Squared Exponential Kernel

Define the *squared exponential kernel* (sometimes called a Gaussian kernel) as

$$k(\varphi_i, \varphi_j) \triangleq e^{-\|\varphi_i - \varphi_j\|_2^2 / 2\mathfrak{l}^2}. \tag{28}$$

$\mathfrak{l}$ is a parameter of the kernel and denoted the *length scale*.

The LLE Kernel

Locally Linear Embedding (LLE, Roweis and Saul (2000)), is a technique to find lower dimensional manifolds to which an observed collection of regressors belong. A brief description of it is as follows:

Let $\{\varphi_i, i = 1, \ldots, N\}$ belong to $U \subset \mathcal{R}^{n_\varphi}$ where U is an unknown manifold of dimension n_z. A coordinatization z_i, ($z_i \in \mathcal{R}^{n_z}$) of U is then obtained by first minimizing the cost function

$$\varepsilon(l) = \sum_{i=1}^{N} \left\| \varphi_i - \sum_{j=1}^{N} l_{ij} \varphi_j \right\|_2^2 \tag{29a}$$

under the constraints

$$\begin{cases} \sum_{j=1}^{N} l_{ij} = 1, \\ l_{ij} = 0 \text{ if } \|\varphi_i - \varphi_j\|_2 > C_i(\kappa) \text{ or if } i = j. \end{cases} \tag{29b}$$

Here, $C_i(\kappa)$ is chosen so that only κ weights l_{ij} become nonzero for every i. κ is a design variable. It is also common to add a regularization to (29a) not to get degenerate solutions.

Then for the determined l_{ij} find z_i by minimizing

$$\sum_{i=1}^{N} \left\| z_i - \sum_{j=1}^{N} l_{ij} z_j \right\|_2^2 \tag{30}$$

wrt $z_i \in R^{n_z}$ under the constraint

$$\frac{1}{N} \sum_{i=1}^{N} z_i z_i^T = I_{n_z \times n_z} \tag{31}$$

z_i will then be the coordinate for φ_i in the lower dimensional manifold. Define now the *LLE-kernel* as

$$k(\varphi_i, \varphi_j) \triangleq l_{ij} \tag{32}$$

where l_{ij} is defined in (29).

Note that the LLE kernel is invariant to translation, rotation, and rescaling of the regressors φ. By using the LLE kernel in (19) we hence assume that the map f_0 is a linear combination of some coordinates that are invariant to translation, rotation, and rescaling of the regressors.

Bibliography

M. Bauwens, H. Ohlsson, K. Barbé, V. Beelaerts, F. Dehairs, and J. Schoukens. On climate reconstruction using bivalve shells: Three methods to interpret the chemical signature of a shell. In *Proceedings of the 7th IFAC Symposium on Modelling and Control in Biomedical Systems*, Aalborg, Denmark, August 2009.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Y. Bengio, O. Delalleau, and N. Le Roux. Label propagation and quadratic criterion. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*, pages 193–216. MIT Press, 2006.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

D. de Ridder and R. P.W. Duin. Locally linear embedding for classification, 2002. Technical Report, PH-2002-01, Pattern Recognition Group, Dept. of Imaging Science & Technology, Delft University of Technology, Delft, The Netherlands.

D. de Ridder, O. Kouropteva, O. Okun, M. Pietikäinen, and R. Duin. Supervised locally linear embedding. In O. Kaynak, E. Alpaydin, E. Oja, and L. Xu, editors, *Artificial Neural Networks and Neural Information Processing – ICANN/ICONIP 2003*, volume 2714 of *Lecture Notes in Computer Science*, pages 333–341. Springer Berlin / Heidelberg, 2003.

D. P. Gillikin, F. Dehairs, A. Lorrain, D. Steenmans, W. Baeyens, and L. André. Barium uptake into the shells of the common mussel (Mytilus edulis) and the potential for estuarine paleo-chemistry reconstruction. *Geochimica et Cosmochimica Acta*, 70(2):395–407, 2006a.

D. P. Gillikin, A. Lorrain, S. Bouillon, P. Willenz, and F. Dehairs. Stable carbon isotopic composition of Mytilus edulis shells: Relation to metabolism, salinity, $\delta^{13}C_{DIC}$ and phytoplankton. *Organic Geochemistry*, 37(10):1371–1382, 2006b.

A. B. Goldberg and X. Zhu. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In *Proceedings of TextGraphs: The First Workshop on Graph Based Methods for Natural Language Processing (TextGraphs'06)*, pages 45–52, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

T. Hastie, R Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

Jr. Hosmer, D. W. A comparison of iterative maximum likelihood estimates of

the parameters of a mixture of two normal distributions under three different types of sample. *Biometrics*, 29(4):761–770, 1973.

T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, 1995.

L. Ljung. *The System Identification Toolbox: The Manual*. The MathWorks Inc. 1st edition 1986, 7th edition 2007, Natick, MA, USA, 2007.

C. J. Merz, D. C. St. Clair, and W. E. Bond. SeMi-supervised adaptive resonance theory (SMART2). In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, volume 3, pages 851–856, June 1992.

E. A Nadaraya. On estimating regression. *Theory of Probability and its Applications*, 9:141–142, 1964.

K. S. Narendra and S.-M. Li. Neural networks in control systems. In P. Smolensky, M. C. Mozer, and D. E. Rumelhard, editors, *Mathematical Perspectives on Neural Networks*, chapter 11, pages 347–394. Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1996.

R. Navaratnam, A. W. Fitzgibbon, and R. Cipolla. The joint manifold model for semi-supervised multi-valued regression. *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV 2007)*, pages 1–8, October 2007.

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of the fifteenth national/-tenth conference on Artificial intelligence/Innovative applications of artificial intelligence (AAAI '98/IAAI '98)*, pages 792–799, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

H. Ohlsson. *Regression on manifolds with implications for system identification*. Licentiate thesis no. 1382, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, December 2008.

H. Ohlsson and L. Ljung. Semi-supervised regression and system identification. In X. Hu, U. Jonsson, B. Wahlberg, and B. Ghosh, editors, *Three Decades of Progress in Control Sciences*. Springer Verlag, December 2010a. To appear.

H. Ohlsson and L. Ljung. Weight determination by manifold regularization. In *Distributed Decision-Making and Control*, Lecture Notes in Control and Information Sciences. Springer Verlag, 2010b. Submitted.

H. Ohlsson, J. Roll, T. Glad, and L. Ljung. Using manifold learning for nonlinear system identification. In *Proceedings of the 7th IFAC Symposium on Nonlinear Control Systems (NOLCOS)*, Pretoria, South Africa, August 2007.

H. Ohlsson, J. Roll, and L. Ljung. Manifold-constrained regressors in system identification. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008.

A. Rahimi, B. Recht, and T. Darrell. Learning to transform time series with a few examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29 (10):1759–1775, 2007.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, December 2005.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

E. Vander Putten, F. Dehairs, L. André, and W. Baeyens. Quantitative in situ microanalysis of minor and trace elements in biogenic calcite using infrared laser ablation - inductively coupled plasma mass spectrometry: A critical evaluation. *Analytica Chimica Acta*, 378(1):261–272, 1999.

F. Wang and C. Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, January 2008.

G. S. Watson. Smooth regression analysis. *Sankhyā Ser.*, 26:359–372, 1964.

X. Yang, H. Fu, H. Zha, and J. Barlow. Semi-supervised nonlinear dimensionality reduction. In *Proceedings of the 23rd international conference on Machine learning (ICML '06)*, pages 1065–1072, New York, NY, USA, 2006. ACM.

L. Zhao and Z. Zhang. Supervised locally linear embedding with probability-based distance for classification. *Computers & Mathematics with Applications*, 57(6):919–926, 2009.

X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

Paper F

On the Estimation of Transfer Functions, Regularizations and Gaussian Processes – Revisited

Authors: Tianshi Chen, Henrik Ohlsson and Lennart Ljung

Edited version of the paper:

T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes – Revisited. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

On the Estimation of Transfer Functions, Regularizations and Gaussian Processes – Revisited

Tianshi Chen, Henrik Ohlsson and Lennart Ljung

Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden

Abstract

Intrigued by some recent results on impulse response estimation by kernel and nonparametric techniques, we revisit the old problem of transfer function estimation from input-output measurements. We formulate a classical approach, focused on finite impulse response (FIR) models, and find that regularization is necessary to cope with the high variance problem. This basic, regularized Least Squares estimate is then a focal point for interpreting other techniques, including Gaussian Process Regression. The role of the kernels – or regularization matrices – is illustrated by numerical experimentation on a data bank of many systems. The consequences for estimating a model of given complexity are illuminated.

1 Introduction

Estimation of the transfer function, or impulse response, of a linear system is a problem that we feel that we have known "everything about" for at least a quarter of a century, *e.g.,* Ljung (1985), based on well established theory and algorithms in statistics and the system identification community. Nevertheless, papers on the problem are still appearing. A recent, very inspiring, and thought provoking, contribution is Pillonetto and De Nicolao (2010a) (see also the follow-up, Pillonetto et al. (2010)), which shows rather remarkable results based on Gaussian Processes and Spline Kernels. That has prompted the current wish to revisit the transfer function estimation problem from scratch.

The problem

Suppose we are given a batch of input-output data (single-input single-output (SISO)) $Z^N = \{u(t), y(t), t = 1, \ldots, N\}$. We have no information about the data,

except that it is collected from a linear system with additive noise. The task is to

a) Estimate, as well as possible, the impulse response of the unknown system.

b) Estimate a model of given order that has an impulse response as close as possible to the unknown system.

The standard answers to these questions are

for b) to use a prediction error/maximum likelihood (PEM/ML) estimate for the given model structure.

for a) to try several models of different orders, apply **b)** and use model order/-model selection techniques to pick the right model order.

We shall revisit these two problems with an emphasis on high order FIR (finite impulse response) models, that are simple, safe and robust ways of building linear models, directly focusing on the impulse response.

2 Problem Formulation

Consider a linear system

$$y(t) = G_0(q)u(t) + v(t) \tag{1}$$

Here q is the shift operator, $qu(t) = u(t+1)$, $v(t)$ is additive noise, independent of the input $u(t)$, and the transfer function is

$$G_0(q) = \sum_{k=1}^{\infty} g_k^0 q^{-k} \tag{2}$$

The coefficients g_k^0 form the *impulse response* of the system. The corresponding frequency function is defined as

$$G_0(e^{i\omega}) = \sum_{k=1}^{\infty} g_k^0 e^{-i\omega k} \tag{3}$$

We measure the sequences $y(t)$ and $u(t)$, $t = 1, 2, \ldots, N$ and the goal is to find an estimate $\hat{G}_N(e^{i\omega})$ of $G_0(e^{i\omega})$ that is as good as possible in the sense of the mean square error (MSE, see (6)). A related goal is to assess and quantify the error in the estimate.

The traditional way is to postulate a finite-dimensional parameterization

$$G(q, \theta) \tag{4}$$

in terms of θ and then estimate θ in some suitable way and deliver the estimate $\hat{G}_N(e^{i\omega}) = G(e^{i\omega}, \hat{\theta}_N)$. Many such parameterizations have been suggested and tested in the literature, *e.g.*, Ljung (1999). A distinct difficulty is to determine the "size" of the parameter vector θ and to assess the error that stems from G_0 being outside the set of functions that is covered within the parameterization. Partly for that reason, alternative approaches based on other ideas, like "Gaussian Pro-

cess Regression", and non-parametric descriptions of the function $G_0(e^{i\omega})$ (or the impulse response) have recently been suggested, *e.g.*, Pillonetto and De Nicolao (2010a); Pillonetto et al. (2010). Related methods for assessing the quality of $\hat{G}_N(e^{i\omega})$ have been discussed in the 1990's and early 2000's (Goodwin et al., 1992, 2002) in connection with bias quantification.

The purpose of the current contribution is to give a simplistic perspective of what can be done to deal with this problem and the different interpretations that can be associated with the solutions.

3 A Data-Bank of Test Data

To test different techniques we generated a data-bank of 5000 systems and data sets. They should be representative of real-life data sets, in that the underlying system is not of low order (but could allow good low order approximations) and should correspond to different signal-to-noise ratios (SNR). We have done as follows:

- A number of 30th order random SISO continuous-time systems were generated in MATLAB using the command `rss`.
- These continuous-time systems were sampled at 3 times the bandwidth to yield the discrete-time systems using the following commands
  ```
  bw=bandwidth(m)
  f = bw*3*2*pi
  md=c2d(m,1/f,'zoh')
  ```
  where `m` is the continuous-time system and `md` is the corresponding discrete-time system.
- These discrete-time systems were split into 2500 "fast" systems `S1` that have all their poles inside a circle with radius 0.95 and 2500 "slow" systems `S2` which have at least on pole outside the circle with radius 0.95 (but inside the unit circle).
- The 5000 systems were simulated with an input which was white Gaussian noise with unit variance, and output additive white Gaussian noise with different variances:
  - low SNR: SNR=1. The additive output noise has the same variance as the noise-free output. The number of data in these records were 375.
  - high SNR: SNR=10. The additive output noise has a variance which is a tenth of the variance of the noise-free output. The number of data in these records were 500.
- This gives four collections of data sets.
  - S1D1: Fast systems with high SNR.
  - S2D1: Slow systems with high SNR.
  - S1D2: Fast systems with low SNR.

– S2D2: Slow systems with low SNR.

All these data sets are accessible from
`http://www.rt.isy.liu.se/~tschen/research/regul_fir/systems_tested/`

To evaluate the various methods the estimates of the impulse response coefficients $\hat{g}_k^N$ were compared to the true ones by the measure

$$W = 100\left(1 - \left[\frac{\sum_{k=1}^{N}|g_k^0 - \hat{g}_k^N|^2}{\sum_{k=1}^{N}|g_k^0 - \bar{g}^0|^2}\right]^{1/2}\right), \quad \bar{g}^0 = \frac{1}{N}\sum_{k=1}^{N} g_k^0 \tag{5}$$

It corresponds to the "fit" in the `compare` command in the System Identification Toolbox (Ljung, 2007). Note that $W = 100$ means a perfect fit between the impulse responses. Each data set gives rise to a particular value of W, and in the tables below we give the average of W over all the sets in a certain collection.

4 A Classical Perspective

In the classical perspective $G_0(e^{i\omega})$ is an unknown quantity that is estimated from the data. The estimate is a random variable (due to the noise $v(t)$) and the quality can be assessed by the "distance" between the estimate and the true value.

A reasonable measure is the mean square error (MSE)

$$M_N = \mathsf{E}\,|\hat{G}_N(e^{i\omega}) - G_0(e^{i\omega})|^2 \tag{6}$$

Here, expectation E is with respect to both the input noise process $u(t)$ and the output noise process $v(t)$. Now, the MSE is classically split into a bias part

$$B_N = \mathsf{E}\,\hat{G}_N(e^{i\omega}) - G_0(e^{i\omega}) \tag{7}$$

and a variance part

$$V_N = \mathsf{E}\,|\hat{G}_N(e^{i\omega}) - \mathsf{E}\,\hat{G}_N(e^{i\omega})|^2 \tag{8}$$

so that

$$M_N = V_N + |B_N|^2 \tag{9}$$

4.1 Trading Variance for Bias to Minimize the MSE

In the expression for the MSE, the bias term B_N decreases and the variance term V_N increases, when the model becomes more flexible (contains more essential parameters). The MSE is then often minimized for a model flexibility that does not give zero bias. In other words, a pragmatic choice of model flexibility allows some bias to reduce variance so that the MSE is minimized.

4.2 OE-Models

We will not be concerned with noise models in this contribution, so a natural numerator/denominator model is

$$G(q,\theta) = \frac{B(q,\theta)}{F(q,\theta)} \tag{10}$$

The PEM/ML approach to the estimation of (10) would be

$$\hat{\theta}_N = \arg\min_{\theta} \sum_{t=1}^{N} |y(t) - G(q,\theta)u(t)|^2 \tag{11}$$

The estimation involves search for the solution of the non-convex problem (11), which may lead to local minima and possibly ill-conditioned calculations. An alternative is to fix the denominator $F(q,\theta)$ to 1 (or any fixed, stable, polynomial) so that a linear regression problem is obtained.

4.3 FIR-Models

The simplest approach to estimate $G(q,\theta)$ is to truncate the expansion (2) at a finite number of impulse response coefficients ("FIR" model, corresponding to fixing $F(q,\theta) = 1$ in (10))

$$G(q,\theta) = \sum_{k=1}^{n} g_k q^{-k}, \qquad \theta = \begin{bmatrix} g_1 & g_2 & \dots & g_n \end{bmatrix}^T \tag{12}$$

The vector θ is then easily estimated by the least squares method: Write the model as

$$y(t) = \varphi^T(t)\theta + v(t), \quad \varphi(t) = \begin{bmatrix} u(t-1) & \dots & u(t-n) \end{bmatrix}^T \tag{13a}$$

$$\text{or} \quad Y_N = \Phi_N^T\theta + \Lambda_N \tag{13b}$$

$$\text{where} \quad Y_N = \begin{bmatrix} y(1) & y(2) & \dots & y(N) \end{bmatrix}^T \tag{13c}$$

$$\Phi_N = \begin{bmatrix} \varphi(1) & \varphi(2) & \dots & \varphi(N) \end{bmatrix} \tag{13d}$$

$$\Lambda_N = \begin{bmatrix} v(1) & v(2) & \dots & v(N) \end{bmatrix}^T \tag{13e}$$

The least-squares solution is well known:

$$\hat{\theta}_N = \arg\min v_N(\theta) \tag{14a}$$

$$v_N(\theta) = \|Y_N - \Phi_N^T\theta\|^2 = \sum_{t=n}^{N} \left(y(t) - \varphi^T(t)\theta\right)^2 \tag{14b}$$

$$\hat{\theta}_N = (\Phi_N\Phi_N^T)^{-1}\Phi_N Y_N = R_N^{-1} F_N \tag{14c}$$

$$F_N = \Phi_N Y_N = \sum_{t=1}^{N} \varphi(t)y(t), \tag{14d}$$

$$R_N = \Phi_N \Phi_N^T = \sum_{t=1}^{N} \varphi(t)\varphi(t)^T \tag{14e}$$

The summation in (14b) starts at n to allow $\varphi(t)$ to be formed. (This is known as the 'non-windowed' case.)

How good is the resulting FIR model? Let us assume that

$$\mathsf{E}\, v(t) = 0, \quad \mathsf{E}\, v(t)v(s) = \sigma^2 \delta_{t,s}, \tag{15}$$

where $\delta_{t,s}$ is Kronecker-delta function, *i.e.,* if $t = s$, $\delta_{t,s} = 1$, otherwise $\delta_{t,s} = 0$. The input $u(t)$ (and thus $\varphi(t)$) is seen as a deterministic variable, and for the conceptual analysis here, for simplicity we will assume that there exists $\mu > 0$ such that

$$\frac{1}{N} R_N \to \mu I_n \quad \text{as } N \to \infty \tag{16}$$

where I_n is the $n \times n$ unit matrix. This will hold w.p. 1 if $u(t)$ is chosen as white noise with variance μ but may be true under many other choices of input (PRBS, certain multi-sine input *etc.*). This means that for reasonably large N,

$$\frac{1}{N} R_N \approx \mu I_n \tag{17}$$

Then it is immediate to show that

$$\mathsf{E}\, \hat{\theta}_N = \theta_0 = \begin{bmatrix} g_1^0 & g_2^0 & \dots & g_n^0 \end{bmatrix}^T \tag{18a}$$

$$\mathsf{E}(\hat{\theta}_N - \theta_0)(\hat{\theta}_N - \theta_0)^T = \sigma^2 R_N^{-1} \approx \frac{\sigma^2}{N\mu} I_n \tag{18b}$$

which gives the, variance, bias, and MSE

$$V_N = \frac{n\sigma^2}{N\mu} \tag{19a}$$

$$B_N = \sum_{k=n+1}^{\infty} g_k^0 e^{i\omega k} \tag{19b}$$

$$M_N = V_N + |B_N|^2 \tag{19c}$$

It is well known, Ljung and Wahlberg (1992), that by letting the order n increase to infinity with the number of data N, sufficiently slowly, the model (12) will converge to the true transfer function (2). To minimize the MSE with respect to the order n for a given data size N requires some idea on the size of B_N as a function of n. If the system has all poles inside a circle with radius $\bar{\lambda}$, then there exists a $\bar{c} > 0$ such that

$$|g_k^0| < \bar{c}\bar{\lambda}^k \tag{20a}$$

$$|B_N| < \frac{\bar{c}\bar{\lambda}^{n+1}}{1 - \bar{\lambda}} \tag{20b}$$

which means that an upper bound on the MSE is minimized for

$$n = \frac{\log N + \log\left(\mu\bar{c}\log(1/\bar{\lambda})/(\sigma^2/\bar{\lambda} - \sigma^2)\right)}{\log(1/\bar{\lambda})} \tag{21}$$

Therefore, in order to minimize the upper bound on the MSE, the FIR model order n should increase with the number of observations N like $\log N$ according to (21). As a result, it follows that the MSE is minimized at relatively low orders compared to the data size.

4.4 Regularization

Still, we see that the variance increases linearly with the FIR model order n so for higher order FIR models it is important to counteract the increasing variance by *regularization*. This is an example of pragmatic bias-variance trade-off, *cf.* Section 4.1. Regularization means that we replace the criterion $\nu_N(\theta)$ in (14) by

$$v_N^R(\theta, D) = \sum_{t=n}^{N}(y(t) - \varphi^T(t)\theta)^2 + \theta^T D\theta \tag{22}$$

where D is a positive semi-definite $n \times n$ matrix. That changes the estimate to be

$$\hat{\theta}_N^R = (R_N + D)^{-1}F_N = (R_N + D)^{-1}[R_N\hat{\theta}_N] \tag{23}$$

How to select D? We have (all expectations are with respect to $v(t)$)

$$\mathsf{E}\,\hat{\theta}_N^R = (R_N + D)^{-1}R_N\theta_0 \tag{24a}$$

$$\theta_{bias}^R = \mathsf{E}\,\hat{\theta}_N^R - \theta_0 = -(R_N + D)^{-1}D\theta_0 \tag{24b}$$

$$\tilde{\theta} = \hat{\theta}_N^R - \mathsf{E}\,\hat{\theta}_N^R = (R_N + D)^{-1}R_N(\hat{\theta}_N - \theta_0) \tag{24c}$$

$$\mathsf{E}\,\tilde{\theta}\tilde{\theta}^T = (R_N + D)^{-1}\sigma^2 R_N(R_N + D)^{-1} \tag{24d}$$

$$\begin{aligned} MSE(\hat{\theta}_N^R) &= \mathsf{E}(\hat{\theta}_N^R - \theta_0)(\hat{\theta}_N^R - \theta_0)^T = \mathsf{E}\,\tilde{\theta}\tilde{\theta}^T + \theta_{bias}(\theta_{bias}^R)^T \\ &= (R_N + D)^{-1}\left(\sigma^2 R_N + D\theta_0\theta_0^T D^T\right)(R_N + D)^{-1} \end{aligned} \tag{24e}$$

Suppose that D is diagonal $D = diag(d_1, d_2, \ldots, d_n)$, and we use (17) for R_N. Concentrating on the MSE of the impulse response coefficients g_k, we find from the (k, k) element of the MSE matrix $MSE(\hat{\theta}_N^R)$ that

$$MSE(\hat{g}_k^N) \approx \frac{\sigma^2\mu N + d_k^2(g_k^0)^2}{(\mu N + d_k)^2} \tag{25}$$

This is minimized with respect to d_k by $d_k = \left(\frac{\sigma}{g_k^0}\right)^2$.

So this gives a clue how to choose the regularization matrix: If the system is exponentially stable as in (20a) the diagonal of the regularization matrix D should increase exponentially:

$$d_k = \frac{\sigma^2}{c\lambda^k}, \quad \text{where } \lambda = \bar{\lambda}^2, c = \bar{c}^2 \tag{26}$$

Remark 1. Note that the FIR model (12) can be seen as a special case of regularization: If we choose the diagonal regularization $D = diag(d_1, d_2, \ldots, d_m)$ with $m > n$ and

$$d_k = \begin{cases} 1 & \text{if} \quad k \leq n \\ \infty & \text{if} \quad k > n \end{cases} \tag{27}$$

this is the same as using an FIR model (12).

Remark 2. Regularization as in (22) is often used in a Tikhonov sense (Tikhonov and Arsenin, 1977), where the objective is to make an ill-conditioned problem have better numerical properties. Here, however, the main aspect of regularization is to better deal with the bias-variance trade-off (9).

4.5 Using a Base-Line Model

If the impulse response is decaying slowly, high order FIR model will be required to capture that. It may then be beneficial to incorporate a "base-line model" that can take care of a dominating part of the impulse response. For example, an additive second order model, like

$$y(t) = \left[\frac{b_1 q^{-1} + b_2 q^{-2}}{1 + f_1 q^{-1} + f_2 q^{-2}} + \sum_{k=1}^{n} g_k q^{-k} \right] u(t) \tag{28}$$

The second order model can be adjusted separately, (using *e.g.,* a PEM/ML methods), form a residual from this model and estimate an FIR model from the residual using regularization as above.

4.6 Cross-Validation

Using FIR model or (actually more general) regularized estimation (22) for optimal MSE means that we must know certain variables (say β), like best FIR model order n in (21) or the optimal regularization parameters c, λ in (26). The necessary information to compute these are typically not known, which in the classical approach typically is handled by *cross-validation*:

1. Split the data record into two parts of the same length: an estimation data part and a validation data part.
2. Estimate models $\hat{G}_N(e^{i\omega}) = G(e^{i\omega}, \hat{\theta}_N)$ using the estimation data for different values of β.
3. Form the error between the measured and the model outputs for these models for the validation data:

$$\epsilon(t, \beta) = y(t) - G(q, \hat{\theta}_N) u(t) \tag{29a}$$

$$W(\beta) = \sum_t |\epsilon(t, \beta)|^2 \tag{29b}$$

and pick the value of β that minimizes $W(\beta)$. The model can then be re-estimated for this β using the whole data record.

4.7 Regularization as Model Merging

It is well known in statistics that if you have two parameter estimates θ_1 and θ_2 with covariance matrices P_1 and P_2 they can be combined to an estimate with minimal variance by

$$\theta = (P_1^{-1} + P_2^{-1})^{-1}(P_1^{-1}\theta_1 + P_2^{-1}\theta_2) \tag{30}$$

In that perspective the regularized estimate (23) can be seen as the combination of the un-regularized estimate $\hat{\theta}_N$ and an estimate $\tilde{\theta} = \begin{bmatrix}0 & 0 & \dots & 0\end{bmatrix}^T$ with variance D^{-1}.

4.8 Numerical Illustration

Example 1: Fixed order OE models

Let us try these methods on our data bank of data sets. We first follow the answer for question **a)** in the introduction. We estimate models (10) of different orders n (same order for $B(q,\theta)$ and $F(q,\theta)$) using the command `m=oe(data,[n,n,1])` in the System Identification Toolbox (Ljung, 2007), and compute the average fit (5) for all the models.

The results are shown in the table below. It also contains the fit when the order for each data set has been chosen by cross-validation (CV) testing orders 5:5:40.

	n=5	n=10	n=15	n=20	n=25	n=30	n=40	CV
S1D1	86.3	89.2	86.4	81.5	74.2	61.5	42.5	89.4
S2D1	68.7	72.8	71.7	70.5	63.1	57.2	42.0	73.0
S1D2	71.9	65.5	56.1	46.1	34.5	19.7	-1.7	70.8
S2D2	50.8	43.0	42.4	30.7	20.5	10.5	-8.6	49.5

One may note that each figure in this table is the average of 2500 fits. It is of course interesting to study the distribution of the fits over the different individual data sets. It turns out that the distributions in the CV column have long tails of poor fits, which indicates that the OE models occasionally have problems. (See also Figure 1 in Section 8.)

Example 2: Fixed order FIR models

Let us try FIR models on the data bank of data sets. We estimate models (12) of different orders n, and compute the average fit (5) for all the models. For fair comparisons we use in all cases the maximum start value of $n = 125$ in (14b). The results are shown in the table below. It also contains the fit when the order for each data set has been chosen by cross-validation (CV) testing orders 5:10:125.

	$n = 5$	$n = 35$	$n = 65$	$n = 95$	$n = 125$	CV
S1D1	32.2	83.1	85.8	81.7	76.9	86.1
S2D1	-0.7	47.1	60.0	64.0	65.3	67.4
S1D2	30.8	61.4	46.0	25.9	-0.1	59.6
S2D2	-1.8	30.5	24.3	8.1	-18.1	30.5

Example 3: FIR-models of order 125 with regularization

The data sets were tested using the FIR model (12) with $n = 125$ and regularization (22) with D diagonal, (26) for different values of c and λ. The result is shown below. The cross-validation (CV) choice of these values (from the grid of 9 values, c=1,5,9, $\lambda = 0.5, 0.9, 0.95$) is also shown.

	$c=1$ $\lambda=0.5$	$c=1$ $\lambda=0.9$	$c=1$ $\lambda=0.95$	$c=9$ $\lambda=0.5$	$c=9$ $\lambda=0.95$	CV
S1D1	51.0	84.8	79.2	58.2	77.5	84.8
S2D1	18.4	67.8	66.8	24.5	65.6	67.2
S1D2	37.4	54.9	36.3	44.7	17.1	55.6
S2D2	6.5	29.5	8.6	12.8	-7.8	23.3

Example 4: As Example 2, but with base-line model (28)

An additive second order model is first identified using the command `m=oe(data,[2,2,1])` and then the residual is regarded as a new measured output based on which the regularization method is used to estimate an FIR model.

	$c=1$ $\lambda=0.5$	$c=1$ $\lambda=0.9$	$c=1$ $\lambda=0.95$	$c=9$ $\lambda=0.5$	$c=9$ $\lambda=0.95$	CV
S1D1	74.8	85.4	79.3	78.0	77.5	86.7
S2D1	56.5	72.2	69.6	58.7	68.4	74.1
S1D2	62.2	57.5	37.4	64.3	17.1	66.4
S2D2	42.2	32.6	9.8	42.8	-6.4	45.8

Findings: *The "standard" approach to cross-validation over different order OE models (Example 1), works reasonably well. Note that in the simulated data, the "true" order is 30, but this is normally not the best order choice for the OE models. The experiments in Example 2 also show that although the true impulse response is infinite, it is normally not the best choice to use maximum FIR model order. The high variance for such models overrides the low bias. Choosing the FIR model order by cross-validation gives a fit between 30 – 85%. Using FIR models of order 125 and regularization (22) with (26) (Example 3) does not always improve the fit for all the c, λ tests, and the good affect is largely dependent on their values, so they should be chosen with care. The cross-validation choice of c, λ over the 9 point-grid gives a fit of about the same size as cross-validation over orders. Adding a second order basel-line model, (Example 4), is beneficial, mostly so for the slow systems.*

5 A Bayesian Perspective

In the Bayesian view, the parameter to be estimated is itself a random variable, and we seek the posterior distribution of this parameter, given the observations.

The following well known and simple result about conditioning jointly Gaussian random variable is a key element in Bayesian calculations:

$$\text{Let} \begin{pmatrix} X \\ Z \end{pmatrix} \in N\left(\begin{bmatrix} m_x \\ m_z \end{bmatrix}, \begin{bmatrix} P_{xx} & P_{xz} \\ P_{zx} & P_{zz} \end{bmatrix} \right) \tag{31a}$$

$$\text{Then} \quad (X|Z = a) \in N(m, P) \tag{31b}$$

$$m = m_x + P_{xz} P_{zz}^{-1} (a - m_z) \tag{31c}$$

$$P = P_{xx} - P_{xz} P_{zz}^{-1} P_{zx} \tag{31d}$$

It is also good to recall the following simple matrix equality:

$$A(I_n + BA)^{-1} = (I_m + AB)^{-1} A \tag{32}$$

where A is an $m \times n$ matrix and B is an $n \times m$ matrix.

In the current setup, we regard the parameter of the nth order FIR model (12), *i.e.,* the impulse response coefficients θ as a random variable, say of Gaussian distribution with zero mean and covariance matrix P_n:

$$\theta \in N(\theta_{ap}, P_n), \qquad \theta_{ap} = 0 \tag{33}$$

If the input $u(t)$ (and $\varphi(t)$, see (13a)) is known and the noise $v(t)$ is independent Gaussian distributed with

$$v(t) \in N(0, \sigma^2) \tag{34}$$

then with

$$Y_N = \Phi_N^T \theta + \Lambda_N \tag{35}$$

Y_N and θ will be jointly Gaussian variables:

$$\begin{pmatrix} \theta \\ Y_N \end{pmatrix} \in N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} P_n & P_n \Phi_N \\ \Phi_N^T P_n & \Phi_N^T P_n \Phi_N + \sigma^2 I_N \end{bmatrix} \right) \tag{36}$$

The posterior distribution of θ given Y_N follows from (31)

$$(\theta|Y_N) \in N(\hat{\theta}_N^{apost}, P^{apost}) \tag{37a}$$

$$\hat{\theta}^{apost} = P_n \Phi_N (\Phi_N^T P_n \Phi_N + \sigma^2 I_N)^{-1} Y_N \tag{37b}$$

$$= (P_n \Phi_N \Phi_N^T + \sigma^2 I_n)^{-1} P_n \Phi_N Y_N \tag{37c}$$

$$= (R_N + \sigma^2 P_n^{-1})^{-1} F_N \tag{37d}$$

$$= \left((\sigma^2 R_N^{-1})^{-1} + P_n^{-1} \right)^{-1} (\sigma^2 R_N^{-1})^{-1} \hat{\theta}_N \tag{37e}$$

$$P^{apost} = P_n - P_n \Phi_N (\Phi_N^T P_n \Phi_N + \sigma^2 I_N)^{-1} \Phi_N^T P_n \tag{37f}$$

$$= P_n - (P_n \Phi_N \Phi_N^T + \sigma^2 I_n)^{-1} P_n \Phi_N \Phi_N^T P_n \tag{37g}$$

$$= \left((\sigma^2 R_N^{-1})^{-1} + P_n^{-1} \right)^{-1} \tag{37h}$$

Here $F_N, R_N, \hat{\theta}_N$ are defined in (14). Here (37b) and (37f) are the expressions from (31) while the steps to (37e) and (37h) using (32) stress the link to (30)

merging the models θ_{ap} and $\hat{\theta}_N$.

We notice that this a posteriori estimate is the same as the regularized estimate $\hat{\theta}_N^R$ if the regularization matrix D is chosen as

$$D = \sigma^2 P_n^{-1} \tag{38}$$

This is just a restatement of the well-known fact that regularization is closely related to prior estimates.

So this gives an insight in how to choose the regularization matrix: Let it reflect the size and correlations of the impulse response coefficients. For the size, it is entirely in line with the choice of diagonal elements (26). If the impulse response is smooth (for example a fast sampled continuous system) it is also natural to let P_n reflect that, by letting the diagonals close to the main diagonal show high correlation. A simple choice is to let the correlation coefficient between θ_k and θ_j be $\rho^{|k-j|}$. With diagonal elements of P_n being $c\lambda^k$ as in (26) we then get a covariance matrix P_n whose (k, j) element is

$$c\rho^{|k-j|}\lambda^{(k+j)/2} \tag{39}$$

The estimates that we come up with are thus the same as in the classical, regularized estimate (23), but the Bayesian perspective has given additional insights into the choice of D.

5.1 Estimating Hyper-Parameters

The Bayesian perspective gives one more insight: Suppose that prior knowledge does not give a definite choice of P_n, but it is natural to let it depend on unknown hyper-parameters parameters β, $P_n(\beta)$ (like $\beta = [c\ \lambda]$ in (26)). From (36) we see that

$$Y_N \in N(0, \sigma^2 I_N + \Phi_N^T P_n(\beta)\Phi_N) \tag{40a}$$

so with a classical twist in this Bayesian framework we can form the likelihood function for β given the observation Y, and estimate β by the maximum likelihood method:

$$\hat{\beta} = \arg\min_{\beta} Y_N^T \Sigma(\beta)^{-1} Y_N + \log\det\Sigma(\beta) \tag{40b}$$

where $\Sigma(\beta) = \sigma^2 I_N + \Phi_N P_n(\beta)\Phi_N^T$. This method of estimating hyper-parameters in the prior distribution is known as the *empirical Bayes* methods.

The noise variance σ^2 used in (40b) and (38) can of course be included among the hyper-parameters, but in the simulations in this paper we have chosen to estimate it from the sample variance of the FIR model (12) with $n = 125$.

5.2 Testing ML Estimation of Hyper-Parameters

Example 5

Let us return to the data bank and test regularization matrices with parameters estimated by the method (40) and $n = 125$. We try the prior covariances: the diagonal (26) and the correlation (39)

$$P_{DI}(k,j) = \begin{cases} c\lambda^k & \text{if } k = j \\ 0 & \text{else} \end{cases}, \text{('Diagonal')} \tag{41a}$$

$$P_{DC}(k,j) = c\rho^{|k-j|}\lambda^{(k+j)/2} \quad \text{('Diagonal/correlated')} \tag{41b}$$

We also test a related prior, where we link ρ and λ: $\rho = \sqrt{\lambda}$:

$$P_{TC}(k,j) = c\min(\lambda^j, \lambda^k), \quad \text{('Tuned/correlated')} \tag{41c}$$

$$\text{In all these cases } 0 < \lambda < 1,\ |\rho| < 1\ c > 0 \tag{41d}$$

The resulting fits are shown in the table below. With DIe, DCe and TCe we mean the same priors but applied to the problem with a second order base-line model (28) identified using the command `m=oe(data,[2,2,1])` first.

	DI	DC	TC	DIe	DCe	TCe
S1D1	86.7	90.7	90.3	88.9	91.0	91.0
S2D1	67.3	73.7	74.5	74.4	77.8	78.6
S1D2	61.8	72.4	72.3	69.1	72.9	74.2
S2D2	33.3	50.4	52.9	50.7	54.5	56.5

Findings: *We see that estimating the hyper-parameters for DI and DIe gives about the same fit as the CV in examples 3 and 4. The ML estimates are slightly better though, perhaps since the search is over a continuum of c, λ and not just the 9-point grid, used for CV. It is also clear that allowing and estimating correlation between the impulse response coefficients with DC, and TC gives a clear improvement. It should be noted that the criterion (40b) is not convex, so it requires some care to initialize the search and search for the minimum. This can be illustrated by the fact that TC actually behaves better than DC in some cases, although it is a special case of DC, but with fewer parameters. In all the tests, we set $c = exp(5)$, $\lambda = \rho = 0.5$.*

6 Gaussian Process Method to Estimate the Transfer Function

Gaussian Process Regression (GPR) has become a widely spread and very popular method for inference in Machine Learning, see, *e.g.*, Rasmussen and Williams (2005). In short, it is about inferring an unknown function $f(x)$ from measurements $y_i, i = 1, 2, \ldots, N$ that bear some information about $f(x)$. The argument

x can either be a continuous or a discrete variable. The prior information about the function is that it is a Gaussian Process, with certain mean and covariance function. This means that the vector $[f(x_1), f(x_2), ..., f(x_n)]$, for any collection of points x_k is a jointly Gaussian random vector, with mean $m(x) = \mathsf{E}\, f(x)$ and covariances

$$Cov\Big(f(x_k), f(x_j)\Big) = P\Big(x_k, x_j\Big) \tag{42}$$

where $P(x_k, x_j)$ is often called a *kernel*. Often $m(x) \equiv 0$. Typically, the observation y_i is a linear functional of $f(x_i)$, measured in additive Gaussian noise. This causes $f(x), y_1, \ldots, y_N$ to be a jointly Gaussian vector, which means that the posterior distributions,

$$p\Big(f(x_1), ..., f(x_n)\Big|y_1, \ldots, y_N\Big) \tag{43}$$

can be calculated by the rules for conditioning jointly Gaussian random variables, (31).

In Pillonetto and De Nicolao (2010a) the GPR is applied to estimating the impulse response of a stable linear system. For a sampled model, the impulse response function is given by $g_k^0, k = 1, \ldots, \infty$ in (2). The observation y_i is the measured output in (1) at time $t = i$. Modeling the impulse response function as a Gaussian process means that, for any n,

$$[g_1, \ldots, g_n] \in N(0, P_n) \tag{44}$$

where P_n is the $n \times n$ upper left block matrix of a semi-infinite matrix P_n with elements $P_{k,j} = P(x_k, x_j)$ (corresponding to the assumption (42)).

This is the same situation as in the Bayesian Perspective (33)–(37). The Gaussian Process estimate of any collections of impulse response coefficients is thus given by (37).

The only thing that remains to be discussed is the choice of prior covariances (44) (or (42)). Of course, the considerations for choosing P_n in (44) and in (33) must be the same, and the relation to the thoughts about the regularization matrix D in (38) still holds. But in GPR several standard choices for (42) exist.

In Pillonetto and De Nicolao (2010a) the following kernels/covariance functions are discussed

$$P_{CS}(k, j) = \begin{cases} c\frac{k^2}{2}(j - \frac{k}{3}), k \le j \\ c\frac{j^2}{2}(k - \frac{j}{3}), k > j \end{cases} \quad \text{('Cubic Spline')} \tag{45a}$$

$$P_{SE}(k, j) = ce^{-\frac{(k-j)^2}{2\lambda^2}} \quad \text{('Squared Exponential')} \tag{45b}$$

$$P_{SS}(k, j) = \begin{cases} c\frac{\lambda^{2k}}{2}(\lambda^j - \frac{\lambda^k}{3}), k \le j \\ c\frac{\lambda^{2j}}{2}(\lambda^k - \frac{\lambda^j}{3}), k > j \end{cases} \quad \text{('Stable Spline')} \tag{45c}$$

Here c and λ are hyper-parameters. There is also a MATLAB toolbox, Pillonetto and De Nicolao (2010b), that implements the GPR, including estimating the hyper-parameters using (40).

Example 6: D-matrices suggested in the GP approach

Let us compute the estimates corresponding to the kernels (regularization matrices) (45), with and without a second order base-line model (28) identified using the command `m=oe(data,[2,2,1])` first. If a base-line model is used, we append 'e' to the kernel name in the table below.

	CS	SE	SS	CSe	SEe	SSe
S1D1	78.0	81.0	90.3	81.6	84.2	89.8
S2D1	-51620	74.8	71.7	-73313	78.9	76.4
S1D2	16.6	44.2	68.0	60.8	65.6	70.3
S2D2	-14289	48.2	48.2	-17373	58.5	51.6

Findings: *The CS kernel, has difficulties with the slow systems, while the kernel SS shows a performance compatible with DC, DI and TC in Example 5.*

Remark 3. For the SS estimate, we used the `SSpline` command in the identification toolbox Pillonetto and De Nicolao (2010b) (with `p=125`, `Lab='ny'`, `mv=0`, `mb=1`, `cn=0`, `red=375`, `LP=0` and `LP2=0`). For the remaining estimates, we used our own implementation, which only differs in the estimation of σ^2 to be in line with the simulations in Example 5. With our implementation, the four figures for the SS estimate become 90.3, 74.2, 67.9 and 49.3 in order.

Remark 4. It is fair to add that the theory around GPR and its relation to Bayesian estimation is much richer than shown here. The estimation of continuous time impulse responses can be handled in the same framework and there are interesting connections to *Reproducing Kernel Hilbert Spaces* (RKHS) and spline approximation. Our point here is that the actual resulting impulse response estimate is a regularized FIR model (23) for a certain choices of regularization matrix D. We refer to Pillonetto and De Nicolao (2010a) for a more complete account of the theory.

7 Estimating a Model of Given Order

Let us now turn to question **b)** in the introduction, to find a model (10) of a given order, that has the best fit to the true impulse response.

The PEM/ML approach (11) has two good features, *e.g.,* Ljung (1999):

1. If the given model structure contains the true, unknown system, PEM/ML has the smallest possible variance (asymptotically) [among all unbiased estimates].
2. If not, PEM/ML will converge, as $N \to \infty$ to the best possible approximation within the given structure.

Is there a catch? Yes, if the true system and model is of high order, the estimate will have rather high variance. It will be the smallest one possible for unbiased estimates, but just as shown in Section 4.1 it is conceivable that the MSE could be

smaller if we allow some bias. There are several ways to achieve this. One would be to regularize the estimation criterion (11), just as in (22). Another would be to use the best available impulse response estimate and fit it to the required model structure. That can be done by model reduction either by minimizing the L_2-fit, *e.g.,* Tjärnström and Ljung (2002) or by balanced realization reduction (see `balred` in the System Identification Toolbox, Ljung (2007)), or any other model reduction technique.

Example 7: Estimating models of a given structure

We use the data bank of data sets to estimate models of the kind (10) of different orders n. We try the following methods:

- OE*: `m=oe(data,[n,n,1])`
- OE†: `m=oe(data(126:end),[n,n,1])`
- TC + BR: `mf=TC(data), m=balred(mf,n)`

where `TC` is the command that generates the FIR model of order 125 as described in Example 5 and `m` denotes the estimated impulse response.

It may be questionable how to deal with initial conditions: On the one hand the FIR-based methods do not use the first 125 outputs, due to (14b). One the other hand, the `oe` command in the System Identification Toolbox, Ljung (2007), is capable of estimating the required initial conditions. Discarding the first 125 data points means that `oe` does not have access to the first 125 inputs, which the FIR based methods have access to. Therefore we compute the `oe` estimate for both cases: OE* and OE†.

The resulting fits between the estimated impulse response and true one are given below.

	n=5	n=10	n=15	n=20	n=25	n=30
S1D1						
OE*	86.3	89.2	86.4	81.5	74.2	61.5
OE†	85.0	87.1	82.9	74.7	67.0	57.1
TC+BR	82.3	90.2	90.4	90.4	90.3	90.3
S2D1						
OE*	68.7	72.8	71.7	70.5	63.1	57.2
OE†	62.7	68.6	66.4	62.1	56.2	43.1
TC+BR	52.1	70.8	73.1	73.9	74.1	74.3
S1D2						
OE*	71.9	65.5	56.1	46.1	34.5	19.7
OE†	65.6	54.6	37.7	26.6	6.4	-2.0
TC+BR	67.7	72.2	72.3	72.3	72.3	72.3
S2D2						
OE*	50.8	43.0	42.3	30.7	20.5	10.5
OE†	38.2	31.7	18.3	7.0	-10.3	-23.2
TC+BR	39.0	50.9	52.2	52.5	52.7	52.8

We also tried the regularizations DI, DC and SS instead of TC, and they gave very similar or slightly inferior results.

Findings: *We see that for low order models (5th order model), the PEM/ML method OE* gives the best fit. Then the variance of OE* is small enough so that reducing it using the regularized FIR at the price of some bias gives a worse MSE fit. For the higher order models it is beneficial to first estimate a 125th order regularized FIR model and then reduce its order using model reduction. We also see that the simple 5th order model for the OE* gives not much worse performance than the best that can be achieved (within 5%). The relatively short data records for all these tests have information contents that are simply not enough to well support higher order models.*

8 Conclusions

We have studied how the impulse response from an unknown system can be estimated as well as possible. The focal point of the paper is the classical regularized method to estimate (high order) FIR model (23). We have discussed how this estimate is obtained in different frameworks and how different interpretations and approaches can be invoked for the choice of regularization matrix D.

The message is also that the impulse response obtained by this simple and robust regularized FIR method has good quality and may have smaller MSE than other, more sophisticated methods. For complex systems, even with rather poor data quality we get somewhat between 50 – 90% fit, in the sense defined in Section 3.

It is conceivable that refined techniques to select and tune suitable D-matrices,

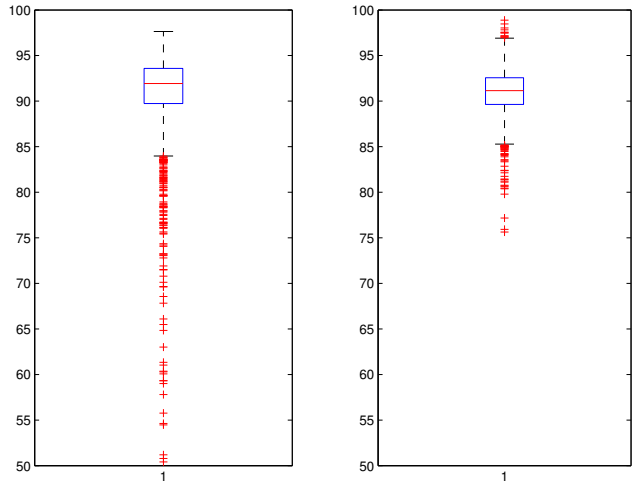

Figure 1: *Box-plots for the 2500 fits for data set S1D1 for OE (left figure) and TCe (right figure). The left figure has an additional 1.4% fits below 50.*

guided by the data can improve the fits further. This is a good topic for further research.

To return to the two questions posed in the introduction, we may note that for the tested data sets, the "standard approach" for question **a)**, works rather well, but the fit can be slightly improved by careful regularization: See Example 1, column CV and Example 5, column TCe. An important remark is that the TCe estimate is considerably more robust than the `oe` estimate with orders determined by CV. Box-plots for the 2500 fits corresponding to CV/S1D1 in Example 1 and the TCe/S1D1 in Example 7 are shown in Figure 1.

For question **b)** we have seen that for higher order models it is better to use model reduction on regularized FIR models than the standard approach (but this may be for models that have higher order than CV would suggest).

Acknowledgments

The work was supported by the Foundation for Strategic Research, SSF, under the center MOVIII and by the Swedish Research Council, VR, within the Linnaeus center CADICS. The authors express their sincere thanks to Gianluigi Pillenetto for helpful discussions and for making his MATLAB code available to us.

Bibliography

T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes – Revisited. In *Proceedings of the 18th IFAC World Congress*, Milano, Italy, 2011. Submitted.

G. Goodwin, M. Gevers, and B. Ninness. Quantifying the error in estimated transfer functions with application to model order selection. *IEEE Trans. Automatic Control*, 37(7):913–929, 1992.

G. Goodwin, J. Braslavsky, and M. Seron. Non-stationary stochastic embedding for transfer function estimation. *Automatica*, 38:47–62, 2002.

L. Ljung. On the estimation of transfer functions. *Automatica*, 21(6):677–696, 1985.

L. Ljung. *System Identification - Theory for the User*. Prentice-Hall, Upper Saddle River, N.J., 2nd edition, 1999.

L. Ljung. *System Identification Toolbox for use with MATLAB. Version 7*. The MathWorks, Inc, Natick, MA, 7th edition, 2007.

L. Ljung and B. Wahlberg. Asymptotic properties of the least-squares method for estimating transfer functions and disturbance spectra. *Advances in Applied Probability*, 24:412–440, 1992.

G. Pillonetto and G. De Nicolao. A new kernel-based approach for linear system identification. *Automatica*, 46(1):81–93, January 2010a.

G. Pillonetto and G. De Nicolao. The stable spline toolbox for system identification. Technical report, University of Padova, Padova, Italy, 2010b.

G. Pillonetto, A. Chiuso, and G. De Nicolao. Prediction error identification of linear systems: A nonparametric Gaussian regression approach. *Automatica*, 2010. To appear.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, December 2005.

A.-I N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York, 1977.

F. Tjärnström and L. Ljung. L-2 model reduction and variance reduction. *Automatica*, 38:1517–1530, 2002.

Paper G

Enabling Bio-Feedback Using Real-Time fMRI

Authors: Henrik Ohlsson, Joakim Rydell, Anders Brun, Jacob Roll, Mats Andersson, Anders Ynnerman and Hans Knutsson

Edited version of the paper:

H. Ohlsson, J. Rydell, A. Brun, J. Roll, M. Andersson, A. Ynnerman, and H. Knutsson. Enabling bio-feedback using real-time fMRI. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008c.

Enabling Bio-Feedback Using Real-Time fMRI

Henrik Ohlsson*, Joakim Rydell**†, Anders Brun**†, Jacob Roll*,
Mats Andersson**†, Anders Ynnerman†‡ and Hans Knutsson**†

*Dept. of Electrical Engineering,
Linköping University,
SE–581 83 Linköping, Sweden

**Division of Medical Informatics,
Department of Biomedical Engineering,
Linköping University, Sweden

‡Division for Visual Information
Technology and Applications,
Department of Science and Technology,
Linköping University, Sweden

†Center for Medical Image Science and
Visualization,
Linköping University, Sweden

{ohlsson,roll}@isy.liu.se, {joary,andbr,matsa,
knutte}@imt.liu.se, andyn@itn.liu.se

Abstract

Despite the enormous complexity of the human mind, fMRI techniques are able to partially observe the state of a brain in action. In this paper we describe an experimental setup for real-time fMRI in a bio-feedback loop. One of the main challenges in the project is to reach a detection speed, accuracy and spatial resolution necessary to attain sufficient bandwidth of communication to close the bio-feedback loop. To this end we have banked on our previous work on real-time filtering for fMRI and system identification, which has been tailored for use in the experiment setup.

In the experiments presented the system is trained to estimate where a person in the MRI scanner is looking from signals derived from the visual cortex only. We have been able to demonstrate that the user can induce an action and perform simple tasks with her mind sensed using real-time fMRI.

The technique may have several clinical applications, for instance to allow paralyzed and "locked in" people to communicate with the outside world. In the meanwhile, the need for improved fMRI performance and brain state detection poses a challenge to the signal processing community. We also expect that the setup will serve as an invaluable tool for neuro science research in general.

1 Introduction

Revealing the functionality of the human brain continues to be one of the grand scientific challenges. Although considerable effort has been made toward this end, many issues remain unresolved.

A new tool in this endeavor is functional Magnetic Resonance Imaging (fMRI). The aim in fMRI is to map cognitive, motor and sensor functions to specific areas in the brain (Weiskopf et al., 2007). The physical foundation for the method is the fact that oxygenated and deoxygenated blood have different magnetic properties. When a neuron in the brain is active it consumes oxygen, which is supplied by the blood. To compensate for the increased rate of oxygen consumption in an active brain area the blood flow is increased and the result is that the oxygenation level of the blood to this area is, in fact, increased. This increase, commonly known as the BOLD (Blood Oxygen Level Dependent) effect, can be measured in a magnetic resonance scanner. Thus, we can locate areas of brain activity indirectly by locating areas with elevated blood oxygen levels.

To map, for example, the sensory function area of a finger, one can stimulate the finger on a volunteer with a brush, while images of the brain are continuously acquired by the MR-scanner. During the stimulation of the finger there is an increase in image intensity (*i.e.,* the active area becomes brighter) compared to a resting state. Thus, to detect activity we need to compare images where the finger is stimulated by the brush to images acquired in a resting state. The areas where the "activated" images are brighter than images acquired in the "rest" state indicate brain areas involved when the brush stimulates the finger.

In the project presented in this paper, we aim at using the estimates of brain activity for the purpose of bio-feedback, *i.e.,* to use the information obtained in the fMRI scan to alter the stimuli generating the fMRI response and thus generating a feedback loop involving the brain. This requires that all parts of the loop, in particular the brain activity estimation, run in real-time. To capture real-time dynamics of the brain, we must acquire each image-slice rapidly. Unfortunately, this makes the images heavily contaminated with random noise. Hence, it is not enough to acquire just one image in activity and one in rest, as it is likely that we can not detect any significant change in intensity due to the high noise level. How the experiment and acquisition of the image volumes are performed is termed the paradigm and is, as a rule, a determining factor for success or failure.

Bio-feedback has since long been explored using electromyography (EMG), temperature and electroencephalography (EEG), see among others Kaushik et al. (2005); Harden et al. (2005); Horowitz (2006); Weiskopf et al. (2004); Birbaumer (2006); Kübler et al. (2001); Kotchoubey et al. (2001); Pfurtscheller et al. (2000); Neuper et al. (2003); Fuchs et al. (2003), but is relatively new in the field of fMRI. Some of the most known examples are the one by DeCharms et al., who showed how patients suffering from chronic pain could learn how to control their pain by bio-feedback based on fMRI (DeCharms et al., 2005), and the one by Yoo et al., who made it possible to navigate throw a 2D maze through fMRI bio-feedback (Yoo et al., 2004).

The long term vision behind the present project is to apply techniques used in system identification for the analysis and 'control' of brain activity. Potentially the 'state of mind' could be steered towards a goal state (activation pattern) by producing a sequence of stimuli that is dependent on the estimated activation pattern sequence. A dual view is that a person can be told to try to make the stimuli produced move towards a target stimulus by will. In the future it may in this way be possible to analyze certain brain functions in terms of brain state transition probability matrices.

However, being in the startup phase of the project, the goal of this first experiment has been to explore the response times that can be expected using fMRI for bio-feedback. We have chosen to work with measurements from the visual cortex, and based on those, track the sight of a person in the MRI scanner.

The paper is structured as follows: We start by formulating our problem in Section 2 and follow up by describing the experiments setup in Section 3. The way we have chosen to solve the problem is presented in Section 4, followed by a description of obtained results in Section 5. We finish with a discussion in Section 6.

Figure 1: *The MRI scanner used in the experiments.*

2 Problem Description

As an example of generating stimuli based on feedback from an fMRI signal, and thereby closing the loop, we here consider a visually-based experiment.

The stimuli are selected to consist of a flashing checkerboard, placed either on the left or the right of the screen. The aim of the experiment is to make the non-flashing part of the visual stimuli to follow the eye movements of the subject, *i.e.*, to flash to the left if the subject is looking to the right, and to flash on the right

side if the subject is looking to the left. Hence, the problem is to detect where the subject is looking at the moment, using the measured fMRI data. Once this is done, the stimulus is simply set to the opposite side.

To judge if the subject is looking to the left or to the right, we need to build a prediction model, with the measurements from the fMRI as the input, and the direction of the subject's gaze as the output. This is a regression problem of high-dimensional nature. The input, *i.e.,* the fMRI measurements, will typically be a signal of approximately 40000 elements or dimensions. Without any kind of regressor selection or regularization, we would therefore get a severe overfit to estimation data.

For the particular experiment setup described, we could use a two-class classifier to determine whether the subject is looking to the left or right. However, aiming at an extension where the stimulus can be moved more than to the left or the right side of the field of vision, regression was considered and not classification.

Previous attempts to handle fMRI data have used a range of various methods, from sliding-window General Linear Modeling (GLM) to Support Vector Machines (SVM), see *e.g.,* Laconte et al. (2007); Nakaia et al. (2006); Cox and Savoy (2003); Gembris et al. (2000); Esposito et al. (2003); LaConte et al. (2005). A good overview is given in Bagarinao et al. (2006).

3 Experiment Setup

As mentioned, the goal of this first real-time feedback experiment has been to create a simple eye-tracker, which will detect if the subject in the scanner is looking to the left or right and show a flashing checkerboard on the right or left 30% of the screen, respectively (see Figure 2, left figure).

The data was acquired using a 1.5 T Philips Achieva MR scanner, see Figure 1. The acquisition resolution was 80 by 80 pixels in each slice, and 7 slices were acquired. Field of view and slice thickness were chosen to obtain a voxel size of approximately $3 \times 3 \times 3$ mm. The use of cubic voxels make three-dimensional signal processing (*e.g.,* smoothing) viable. The acquired data cover the primary visual cortex, and a surface coil was used to provide an optimal signal-to-noise ratio within this region. To obtain high BOLD contrast, the echo time (TE) was set to 40 ms and the repetition time (TR) was set to 1000 ms. Hence we acquire one volume per second, which we consider to be sufficient to deliver close to realtime feedback to the subject.

The subject in the scanner was exposed to a visual stimulus through a pair of head mounted displays. The data processing was done in MATLAB on a standard laptop.

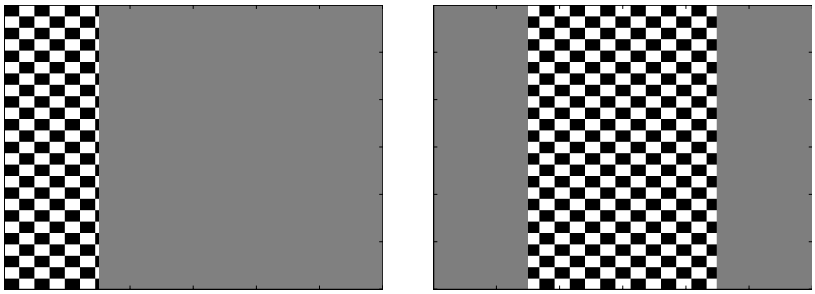

Figure 2: *Visual stimuli used. Left figure: left* 30% *of the screen as a flashing checkerboard. Right figure: a centered vertical stripe, covering* 100% *vertically and* 40% *horizontally of the screen.*

4 Training and Real-Time fMRI

Before starting the real-time feedback phase, a training phase was performed to build a prediction model.

4.1 Training Phase

During the training phase, two training data sets were gathered. First, the subject in the scanner was exposed to a flashing checkerboard, a centered vertical stripe covering 100% vertically and 40% horizontally of the screen. Figure 2 shows the visual stimulus used. Data was gathered for approximately 40 seconds.

The second training data set was gathered by instructing the subject in the scanner to look away from a periodically shifting flashing checkerboard (15 seconds flashing checkerboard on the left, 15 seconds flashing checkerboard on the right, see Figure 2). Data was gathered for approximately 90 seconds.

Using this last data set, 8 voxels were picked out correlating the best with the paradigm. The reason for not just using the two best correlating voxels was to be able to use the redundancy in data to reduce the impact of noise. The 8 voxels were picked out by first computing the correlation to a sine wave with a period of 30 seconds. This was done voxel-by-voxel. In order not to have to go through all possible phase shifts for the sine wave, to find the phase shift associated with the best correlation, Canonical Correlation Analysis (CCA, Hotelling (1936)) was used. In this context, CCA has the property to automatically find the time delay in the sine wave giving the best correlation. Note that this usage of CCA would not be possible using a square wave. The voxel with the best correlation was chosen as the first of the 8 voxels. The three voxels with a phase within 90 degrees of the first one and with the highest correlations were also picked out. Finally, the 4 voxels correlating best with a sine wave at least 90 degrees out of phase compared to the best correlating voxel were chosen.

At this time, the voxel locations were verified to be within the visual cortex. This was done manually by inspection of a plot like the one shown in Figure 3. To fur-

ther reduce noise and to gain some robustness against movements of the subject, the two training data sets were spatially smoothed. Note that this will turn the 8 chosen voxels into 8 neighborhoods, centered at the previously chosen voxels.

The 8 chosen neighborhood signals were then picked out from the two training data sets, detrended voxel-by-voxel, and merged together (90 seconds of data associated with the left-right stimuli followed by 40 seconds of data associated with the centered vertical flashing stripe). Finally, a linear predictor, using the 8 signals as regressors, was fit to a square wave, switching between −1 and +1 (in phase with first sine wave used above), and followed by zeros for the last 40 seconds. Hence, the predictor was expected to give −1 if the subject was looking to the left of the checkerboard, +1 if the subject is looking to the right, and zero otherwise.

The training phase is summarized in Algorithm 1.

Algorithm 1 Training Phase

Given data from a voxel i associated with stimulus on the left-right, $X_i^{lr}(t)$, $t = 1 \ldots 90$, and from stimulus as a centered vertical stripe, $X_i^f(t)$, $t = 1 \ldots 40$.

1. Use CCA to find how well $X_i^{lr}(t)$ correlates to a sine wave with a period of 30 seconds.
2. Find the index for the voxel with the highest correlation.
3. Find the three voxels with the highest correlation but with a phase difference less then 90 degrees compared to the best correlating voxel.
4. Find the 4 voxels with the highest correlation having a phase difference of more than 90 degrees compared to the best correlated voxel.
5. Make sure that the chosen voxels are in the visual cortex.
6. For the chosen voxels, make a spatial smoothing using a Gaussian spatial filter to obtain $\tilde{X}_i^{lr}(t)$ and $\tilde{X}_i^f(t)$.
7. Detrend, voxel-by-voxel, the signals $\tilde{X}_i^{lr}(t)$ and $\tilde{X}_i^f(t)$ from the 8 chosen neighborhoods.
8. Concatenate the detrended $\tilde{X}_i^{lr}(t)$ and $\tilde{X}_i^f(t)$ to form $X_i(t)$.
9. Find the θ_i such that $\sum_{t=1}^{130} |y(t) - \sum_{i=1}^{8} \theta_i X_i(t)|^2$ is minimized; $y(t)$, $t = 1 \ldots 90$ being a $-1/+1$ square wave in phase with the best correlated voxel, and $y(t) = 0$, $t = 91 \ldots 130$.

4.2 Real-Time Phase

During the real-time data phase, the data was first spatially smoothed, just as the training data set. The signals from the 8 chosen neighborhoods were then detrended using a Windowed Least Squares (WLS) approach, with a window size of 50 seconds. With $\bar{X}_i(t)$ being the data at time t from neighborhood i, let

$$\vec{X}_i(t) = \begin{bmatrix} \bar{X}_i(t) & \bar{X}_i(t-1) & \ldots & \bar{X}_i(t-50) \end{bmatrix}. \quad (1)$$

We can remove a linear trend in $\vec{X}_i(t)$ by subtracting the best fitted line

$$\tilde{X}_i(t) = \vec{X}_i(t) - \begin{bmatrix} \alpha_i & \beta_i \end{bmatrix} \begin{bmatrix} 1 & 1 & \dots & 1 \\ t & t-1 & \dots & t-50 \end{bmatrix} \tag{2}$$

where α_i, β_i minimizes

$$\left\| \vec{X}_i(t) - \begin{bmatrix} \alpha_i & \beta_i \end{bmatrix} \begin{bmatrix} 1 & 1 & \dots & 1 \\ t & t-1 & \dots & t-50 \end{bmatrix} \right\|^2 . \tag{3}$$

The first element in $\tilde{X}_i(t)$ after the trend has been removed is used as input for the linear predictor. The resulting signal from this procedure will take values close to one when the subject is looking to the right and minus one when the subject is looking to the left. The flashing checkerboard was therefore moved to the left side when the predictor signal exceeded a certain threshold, and correspondingly for the right side.

For validation, the subject in the scanner was during the real-time phase instructed to keep its eyes on a moving point on the screen. In this way, we could keep track of where the subject was looking, which was used to validate the results.

The real-time phase is summarized in Algorithm 2.

Algorithm 2 Real-Time Phase

Given new data $X_i(t)$. Let T be a threshold and assume that the θ_i and the 8 chosen neighborhoods are given from the training phase. Do the following:

1. For the chosen voxels, perform a spatial smoothing using a Gaussian spatial filter to obtain $\bar{X}_i(t)$.
2. Detrend, voxel-by-voxel, the signals $\bar{X}_i(t)$ from the 8 chosen neighborhoods to get $\tilde{X}_i(t)$.
3. Compute $\hat{y}(t) = \sum_{i=1}^{8} \theta_i \tilde{X}_i(t)$.
4. If $\hat{y}(t) < -T$: move the stimulus to the right side; if $\hat{y}(t) > T$: move the stimulus to the left side; and if $-T < \hat{y}(t) < T$: use the same stimulus as for $t-1$.

5 Results

Figure 3 shows the 8 voxels picked out in the training phase. The 4 voxels correlating best with the flashing checkerboard on the left are shown in the top row of Figure 3. The best correlation was computed for the voxel shown in the first column from the left, second best for the second column from the left and so on. A correlation of 0.6 was the highest correlation computed, and the signal from this voxel during the training phase is shown in the top figure of Figure 4.

The bottom row of Figure 3 shows voxels with the highest correlation to stimuli on the right, arranged in the same way as the top row. As can be seen, the neighborhoods shown in the second row, columns 2–4, are not within the visual cortex.

The signals from these neighborhood were therefore not considered. The signal from the voxel correlating best (correlation 0.55) with stimuli on the right side is shown in the bottom figure of Figure 4.

The signal from the 5 remaining neighborhoods were weighted together to give an as good fit to the stimuli as possible (see Figure 5).

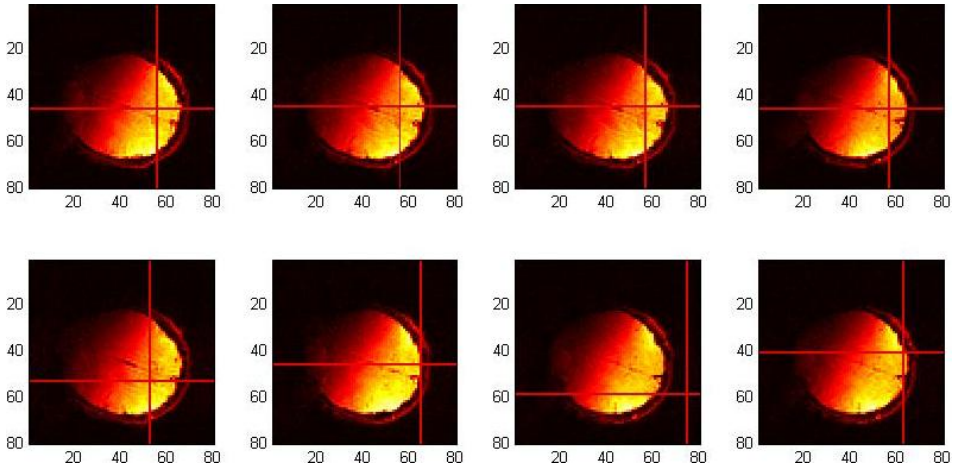

Figure 3: *Slices associated with the chosen 8 voxels. A red cross, centered at the chosen voxel, is used to show the location of the chosen voxel. The top row shows the voxels correlating best with stimuli to the left and the bottom row with stimuli to the right. The best correlation was found for voxels shown in the first column, then second best in the next column and so on.*

Figure 6 shows logged results from the real-time phase using the computed weighting and choice of neighborhood. The horizontal coordinate for the reference point where the subject in the scanner was aiming to look at is shown in the top subplot. The computed signal from the fMRI data is given in the middle subplot. The bottom subplot shows if the flashing checkerboard is to the left or the right (-1 if the checkerboard is to the left and $+1$ if it is to the right). It can be seen that, as the subject shifts focus from one side to the other, it takes between 2.5 and 7 seconds until the visual stimulus has changed.

6 Discussion

It should be emphasized that the purpose of this work has not been to introduce a method for an eye-gaze interface; the authors are well aware that there exist more simple, inexpensive and exact solutions for that specific purpose. The main contribution is instead the closing of the bio-feedback loop where the user experiences a real-time response from the state of his or her mind and is able to perform a simple task.

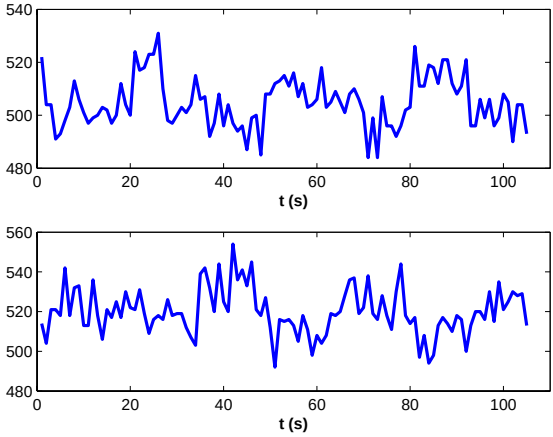

Figure 4: *The signals coming from the voxels correlating best with stimuli. Top figure: best correlated signal with stimuli to the left, bottom figure: best correlated signal with stimuli to the right.*

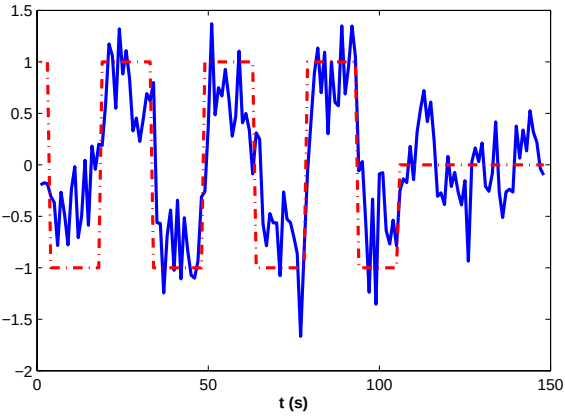

Figure 5: *The weighted signal computed from the* 5 *chosen neighborhoods (solid line). Dash-dotted line represents the stimuli. First* 105 *seconds: stimuli switching periodically between left and right. Last* 43 *seconds: the flashing vertical stripe at the center of the field of view. Three of the* 8 *chosen neighborhoods have been removed because of their location outside the visual cortex.*

The choice of a visual stimulus is not of central importance for this work. A reason for choosing the specific experimental setup was that MR-compatible goggles provide a simple perception of a stimulus inside the MR-scanner, and the flashing checkerboard pattern enables a distinctive activation in the visual cortex due to both temporal variation and spatial high contrast edges.

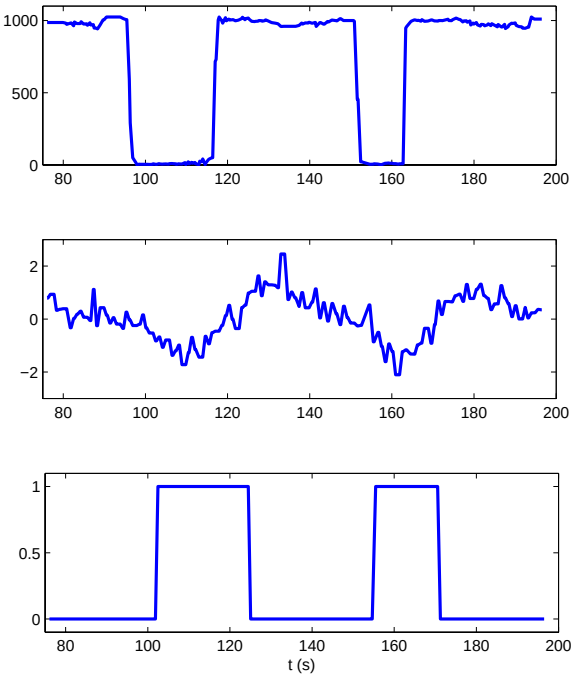

Figure 6: *Logged results. Top figure: The reference signal showing where the subject should focus. A small value corresponds to the subject in the scanner looking to the left, while a high value corresponds to looking to the right. Middle figure: Computed signal from the fMRI measurements. Bottom figure: The location of the stimuli. Small value: flashing checkerboard on the left part of the screen; high value: checkerboard on the right part of the screen.*

The use of an MR-scanner as a Brain Computer Interface (BCI) in a real-time bio-feedback loop stresses the boundaries for image acquisition and signal processing to the absolute limit. In our current setup an average user experiences a response time of 5 seconds. However, we observed times down to 2.5 seconds. Similar results have recently been shown by Laconte et al. (2007). Considering that the BOLD signal, in itself, has a response time of the same order, these response times can be seen as quite good results. However, it has been shown that it is possible to spot activity in the BOLD signal considerably earlier, see Kollias et al. (2000) and Yacoub and Hu (1999). The question of whether these early signs of activity are large enough to be able to reliably detect activity is still open. MRI is continually improving with respect to acquisition time, SNR and resolution. A limiting factor for functional-MRI is the temporal dynamics of the BOLD response. For the visual cortex, stimuli like the flashing checkerboard pattern induce a BOLD

response that is present for approximately 30 seconds (Harel et al., 2001). During the first half, the BOLD signal increases in intensity apart from a very small initial dip. After that time, the blood oxygen control system of the brain compensates the blood oxygen distribution for this new state, and the BOLD response disappears.

An objective method to evaluate the performance of such a real-time fMRI system is to estimate the bandwidth in the bio-feedback loop. For the present setup the bandwidth is approximately 0.2 bits/s. A shorter acquisition time (currently about 1 s) will not by itself be a key factor to increase of the bandwidth above 1 bit/s limit, considering the temporal dynamics of the BOLD response. An improved SNR of the MRI would on the other hand provide the means to discern the BOLD response within the noise at a much earlier stage in the activation process, which has the potential to increase the bandwidth several orders of magnitude. This is a real future challenge both for the manufacturers of MRI equipment as well as for the signal processing community.

Although it is convenient to use visual stimuli inside the MR-scanner some issues must be considered. During the training phase, both unconscious and reflex-based eye movements degrade the training data. Using more advanced VR-goggles with an eye tracker device that fixate the stimuli at a local area in the visual cortex, independently of the eye motions of the user, would provide a significant improvement of the training data set. An additional problem using a gaze based BCI is that the user may unintentionally move the head a little synchronously to the movement of the gaze. These motion artifacts are the main reason why neighborhoods outside the visual cortex sometimes may provide high correlation to the paradigm. To detect and compensate for occasional head motions would improve the performance of the real-time phase. The head motion can be modeled as a rigid body motion and the new locations of the selected neighborhoods are straight forward to compute once the global head motion is estimated. To compensate for a user that continuously moves his or head is much more cumbersome due to the complex motion artifacts which are associated to MRI. Detection and compensation for small occasional head movements should be possible to perform within this setup.

A next step in our research is to extend the simple left/right response to a more complicated task involving a graded response. A possible task would be a virtual pole balancing problem. Such a graded response could be computed in different ways, but a straight-forward method is to apply a temporal integration on the present output signal.

A possible way to further increase the bandwidth in the bio-feedback loop would be to use parallel or sequential activation of different brain areas. Broca's and Wernicke's areas are *e.g.*, activated in speech processing using language or signs. An activation in these areas could be deliberately induced by the person in the scanner by focusing the mind on a sentence, which can be done without any movement of the eyes. Activating several cortical areas at once will make the training phase more complex, and more advanced adaptive training methods will

be required to fully explore these possibilities. To optimize the BCI bandwidth for a specific task, adaptation to each user's own capabilities is necessary.

Acknowledgments

This work was supported by the Strategic Research Center MOVIII, funded by the Swedish Foundation for Strategic Research, SSF.

Bibliography

E. Bagarinao, T. Nakai, and Y. Tanaka. Real-time functional MRI: Development and emerging applications. *Magn Reson Med Sci*, 5(3):157–165, 2006.

N. Birbaumer. Breaking the silence: Brain-computer interfaces (BCI) for communication and motor control. *Psychophysiology*, 43(6):517–532, November 2006.

D. D. Cox and R. L. Savoy. Functional magnetic resonance imaging (fMRI) "brain reading": Detecting and classifying distributed patterns of fMRI activity in human visual cortex. *NeuroImage*, 19:261–270, June 2003.

R. C. DeCharms, F. Maeda, G. H. Glover, D. Ludlow, J. M. Pauly, S. Whitfield, J. D. E. Gabrieli, and S. C. Mackey. Control over brain activation and pain learned by using real-time functional MRI. *Proc Natl Acad Sci USA*, 102:18626–18631, 2005.

F. Esposito, E. Seifritz, E. Formisano, R. Morrone, T. Scarabino, G. Tedeschi, S. Cirillo, R. Goebel, and F. Di Salle. Real-time independent component analysis of fMRI time-series. *NeuroImage*, 20(4):2209–2224, December 2003.

T. Fuchs, N. Birbaumer, W. Lutzenberger, J. H. Gruzelier, and J. Kaiser. Neurofeedback treatment for attention-deficit/hyperactivity disorder in children: A comparison with methylphenidate. *Applied Psychophysiology and Biofeedback*, 28(1):1–12, March 2003.

D. Gembris, J. G. Taylor, S. Schor, W. Frings, D. Suter, and S. Posse. Functional magnetic resonance imaging in real time (FIRE): Sliding-window correlation analysis and reference-vector optimization. *Magnetic Resonance in Medicine*, 43:259–268, March 2000.

R. N. Harden, T. T. Houle, S. Green, T. A. Remble, S. R. Weinland, S. Colio, J. Lauzon, and T. Kuiken. Biofeedback in the treatment of phantom limb pain: A time-series analysis. *Applied Psychophysiology and Biofeedback*, 30(1):83–93, March 2005.

N. Harel, A. Shmuel, S-L. Lee, D-S. Kim, T. Q. Duong, E. Yacoub, X. Hu, K. Ugurbi, and S-G. Kim. Observation of positive and negative BOLD signals in visual cortex. In *Proceedings of the 9th Meeting of the International Society for Magnetic Resonance in Medicine (ISMRM)*, Glasgow, Scotland, 2001.

S. Horowitz. Biofeedback applications - a survey of clinical research. *Alternative & complementary therapies*, 12(6):275–281, December 2006.

H. Hotelling. Relation between two sets of variates. *Biometrika*, 28:322–377, 1936.

R. Kaushik, R. M. Kaushik, S. K. Mahjan, and V. Rajesh. Biofeedback assisted diaphragmatic breathing and systematic relaxation versus propranolol in long

term prophalaxis of migraine. *Complement Ther Med*, 13(3):165–174, September 2005.

S. S. Kollias, X. Golay, P. Boesiger, and A. Valavanis. Dynamic characteristics of oxygenation-sensitive MRI signal in different temporal protocols for imaging human brain activity. *Neuroradiology*, 42:591–601, August 2000.

B. Kotchoubey, U. Strehl, C. Uhlmann, S. Holzapfel, M. König, W. Fröscher, V. Blankenhorn, and N. Birbaumer. Modification of slow cortical potentials in patients with refractory epilepsy: A controlled outcome study. *Epilepsia*, 42 (3), March 2001.

A. Kübler, B. Kotchoubey, J. Kaiser, J. R. Wolpaw, and N. Birbaumer. Brain-computer communication: Unlocking the locked in. *Psychological Bulletin*, 127:358–375, May 2001.

S. LaConte, S. Strother, V. Cherkassky, J. Anderson, and X. Hu. Support vector machines for temporal classification of block design fMRI data. *NeuroImage*, 26:317–329, June 2005.

S. Laconte, S. Peltier, and X. Hu. Real-time fMRI using brain-state classification. *Human Brain Mapping*, 28:1033–1044, 2007.

T. Nakaia, E. Bagarinaob, K. Matsuoa, Y. Ohgamic, and C. Katod. Dynamic monitoring of brain activation under visual stimulation using fMRI – The advantage of real-time fMRI with sliding window GLM analysis. *Journal of Neuroscience Methods*, 157:158–167, October 2006.

C. Neuper, G. R. Müller, A. Kübler, N. Birbaumer, and G. Pfurtscheller. Clinical application of an EEG-based brain-computer interface: A case study in a patient with severe motor impairment. *Clin Neurophysiol*, 114:399–409, March 2003.

H. Ohlsson, J. Rydell, A. Brun, J. Roll, M. Andersson, A. Ynnerman, and H. Knutsson. Enabling bio-feedback using real-time fMRI. In *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, December 2008.

G. Pfurtscheller, C. Guger, G. Muller, G. Krausz, and C. Neuper. Brain oscillations control hand orthosis in a tetraplegic. *Neuroscience Letters*, 292:211–214, October 2000.

N. Weiskopf, F. Scharnowski, R. Veit, R. Goebel, N. Birbaumer, and K. Mathiak. Self-regulation of local brain activity using real-time functional magnetic resonance imaging (fMRI). *Journal of Physiology-Paris*, 98:357–373, July-November 2004.

N. Weiskopf, R. Sitaram, O. Josephs, R. Veit, F. Scharnowski, R. Goebel, N. Birbaumer, R. Deichmann, and K. Mathiak. Real-time functional magnetic resonance imaging: Methods and applications. *Magnetic Resonance Imaging*, 25: 989–1003, July 2007.

E. Yacoub and X. Hu. Detection of the early negative response in fMRI at 1.5 tesla. *Magn Reson Med*, 41:1088–1092, 1999.

S. S. Yoo, T. Fairneny, N. K. Chen, S. E. Choo, L. P. Panych, H. Park, S. Y. Lee, and F. A. Jolesz. Brain-computer interface using fMRI: Spatial navigation by thoughts. *Neuroreport*, 15(10):1591–1595, July 2004.

Index

PhD Dissertations
Division of Automatic Control
Linköping University

M. Millnert: Identification and control of systems subject to abrupt changes. Thesis No. 82, 1982. ISBN 91-7372-542-0.

A. J. M. van Overbeek: On-line structure selection for the identification of multivariable systems. Thesis No. 86, 1982. ISBN 91-7372-586-2.

B. Bengtsson: On some control problems for queues. Thesis No. 87, 1982. ISBN 91-7372-593-5.

S. Ljung: Fast algorithms for integral equations and least squares identification problems. Thesis No. 93, 1983. ISBN 91-7372-641-9.

H. Jonson: A Newton method for solving non-linear optimal control problems with general constraints. Thesis No. 104, 1983. ISBN 91-7372-718-0.

E. Trulsson: Adaptive control based on explicit criterion minimization. Thesis No. 106, 1983. ISBN 91-7372-728-8.

K. Nordström: Uncertainty, robustness and sensitivity reduction in the design of single input control systems. Thesis No. 162, 1987. ISBN 91-7870-170-8.

B. Wahlberg: On the identification and approximation of linear systems. Thesis No. 163, 1987. ISBN 91-7870-175-9.

S. Gunnarsson: Frequency domain aspects of modeling and control in adaptive systems. Thesis No. 194, 1988. ISBN 91-7870-380-8.

A. Isaksson: On system identification in one and two dimensions with signal processing applications. Thesis No. 196, 1988. ISBN 91-7870-383-2.

M. Viberg: Subspace fitting concepts in sensor array processing. Thesis No. 217, 1989. ISBN 91-7870-529-0.

K. Forsman: Constructive commutative algebra in nonlinear control theory. Thesis No. 261, 1991. ISBN 91-7870-827-3.

F. Gustafsson: Estimation of discrete parameters in linear systems. Thesis No. 271, 1992. ISBN 91-7870-876-1.

P. Nagy: Tools for knowledge-based signal processing with applications to system identification. Thesis No. 280, 1992. ISBN 91-7870-962-8.

T. Svensson: Mathematical tools and software for analysis and design of nonlinear control systems. Thesis No. 285, 1992. ISBN 91-7870-989-X.

S. Andersson: On dimension reduction in sensor array signal processing. Thesis No. 290, 1992. ISBN 91-7871-015-4.

H. Hjalmarsson: Aspects on incomplete modeling in system identification. Thesis No. 298, 1993. ISBN 91-7871-070-7.

I. Klein: Automatic synthesis of sequential control schemes. Thesis No. 305, 1993. ISBN 91-7871-090-1.

J.-E. Strömberg: A mode switching modelling philosophy. Thesis No. 353, 1994. ISBN 91-7871-430-3.

K. Wang Chen: Transformation and symbolic calculations in filtering and control. Thesis No. 361, 1994. ISBN 91-7871-467-2.

T. McKelvey: Identification of state-space models from time and frequency data. Thesis No. 380, 1995. ISBN 91-7871-531-8.

J. Sjöberg: Non-linear system identification with neural networks. Thesis No. 381, 1995. ISBN 91-7871-534-2.

R. Germundsson: Symbolic systems – theory, computation and applications. Thesis No. 389, 1995. ISBN 91-7871-578-4.

P. Pucar: Modeling and segmentation using multiple models. Thesis No. 405, 1995. ISBN 91-7871-627-6.
H. Fortell: Algebraic approaches to normal forms and zero dynamics. Thesis No. 407, 1995. ISBN 91-7871-629-2.
A. Helmersson: Methods for robust gain scheduling. Thesis No. 406, 1995. ISBN 91-7871-628-4.
P. Lindskog: Methods, algorithms and tools for system identification based on prior knowledge. Thesis No. 436, 1996. ISBN 91-7871-424-8.
J. Gunnarsson: Symbolic methods and tools for discrete event dynamic systems. Thesis No. 477, 1997. ISBN 91-7871-917-8.
M. Jirstrand: Constructive methods for inequality constraints in control. Thesis No. 527, 1998. ISBN 91-7219-187-2.
U. Forssell: Closed-loop identification: Methods, theory, and applications. Thesis No. 566, 1999. ISBN 91-7219-432-4.
A. Stenman: Model on demand: Algorithms, analysis and applications. Thesis No. 571, 1999. ISBN 91-7219-450-2.
N. Bergman: Recursive Bayesian estimation: Navigation and tracking applications. Thesis No. 579, 1999. ISBN 91-7219-473-1.
K. Edström: Switched bond graphs: Simulation and analysis. Thesis No. 586, 1999. ISBN 91-7219-493-6.
M. Larsson: Behavioral and structural model based approaches to discrete diagnosis. Thesis No. 608, 1999. ISBN 91-7219-615-5.
F. Gunnarsson: Power control in cellular radio systems: Analysis, design and estimation. Thesis No. 623, 2000. ISBN 91-7219-689-0.
V. Einarsson: Model checking methods for mode switching systems. Thesis No. 652, 2000. ISBN 91-7219-836-2.
M. Norrlöf: Iterative learning control: Analysis, design, and experiments. Thesis No. 653, 2000. ISBN 91-7219-837-0.
F. Tjärnström: Variance expressions and model reduction in system identification. Thesis No. 730, 2002. ISBN 91-7373-253-2.
J. Löfberg: Minimax approaches to robust model predictive control. Thesis No. 812, 2003. ISBN 91-7373-622-8.
J. Roll: Local and piecewise affine approaches to system identification. Thesis No. 802, 2003. ISBN 91-7373-608-2.
J. Elbornsson: Analysis, estimation and compensation of mismatch effects in A/D converters. Thesis No. 811, 2003. ISBN 91-7373-621-X.
O. Härkegård: Backstepping and control allocation with applications to flight control. Thesis No. 820, 2003. ISBN 91-7373-647-3.
R. Wallin: Optimization algorithms for system analysis and identification. Thesis No. 919, 2004. ISBN 91-85297-19-4.
D. Lindgren: Projection methods for classification and identification. Thesis No. 915, 2005. ISBN 91-85297-06-2.
R. Karlsson: Particle Filtering for Positioning and Tracking Applications. Thesis No. 924, 2005. ISBN 91-85297-34-8.
J. Jansson: Collision Avoidance Theory with Applications to Automotive Collision Mitigation. Thesis No. 950, 2005. ISBN 91-85299-45-6.
E. Geijer Lundin: Uplink Load in CDMA Cellular Radio Systems. Thesis No. 977, 2005. ISBN 91-85457-49-3.
M. Enqvist: Linear Models of Nonlinear Systems. Thesis No. 985, 2005. ISBN 91-85457-64-7.
T. B. Schön: Estimation of Nonlinear Dynamic Systems — Theory and Applications. Thesis No. 998, 2006. ISBN 91-85497-03-7.

I. Lind: Regressor and Structure Selection — Uses of ANOVA in System Identification. Thesis No. 1012, 2006. ISBN 91-85523-98-4.

J. Gillberg: Frequency Domain Identification of Continuous-Time Systems Reconstruction and Robustness. Thesis No. 1031, 2006. ISBN 91-85523-34-8.

M. Gerdin: Identification and Estimation for Models Described by Differential-Algebraic Equations. Thesis No. 1046, 2006. ISBN 91-85643-87-4.

C. Grönwall: Ground Object Recognition using Laser Radar Data – Geometric Fitting, Performance Analysis, and Applications. Thesis No. 1055, 2006. ISBN 91-85643-53-X.

A. Eidehall: Tracking and threat assessment for automotive collision avoidance. Thesis No. 1066, 2007. ISBN 91-85643-10-6.

F. Eng: Non-Uniform Sampling in Statistical Signal Processing. Thesis No. 1082, 2007. ISBN 978-91-85715-49-7.

E. Wernholt: Multivariable Frequency-Domain Identification of Industrial Robots. Thesis No. 1138, 2007. ISBN 978-91-85895-72-4.

D. Axehill: Integer Quadratic Programming for Control and Communication. Thesis No. 1158, 2008. ISBN 978-91-85523-03-0.

G. Hendeby: Performance and Implementation Aspects of Nonlinear Filtering. Thesis No. 1161, 2008. ISBN 978-91-7393-979-9.

J. Sjöberg: Optimal Control and Model Reduction of Nonlinear DAE Models. Thesis No. 1166, 2008. ISBN 978-91-7393-964-5.

D. Törnqvist: Estimation and Detection with Applications to Navigation. Thesis No. 1216, 2008. ISBN 978-91-7393-785-6.

P-J. Nordlund: Efficient Estimation and Detection Methods for Airborne Applications. Thesis No. 1231, 2008. ISBN 978-91-7393-720-7.

H. Tidefelt: Differential-algebraic equations and matrix-valued singular perturbation. Thesis No. 1292, 2009. ISBN 978-91-7393-479-4.