Convex Identification of Stable Dynamical Systems

Jack Umenberger, BE (Hons 1)

A thesis submitted in fulfillment
of the requirements of the degree of
Doctor of Philosophy

Australian Centre for Field Robotics
School of Aerospace, Mechanical and Mechatronic Engineering
The University of Sydney

August 2017

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the University or other institute of higher learning, except where due acknowledgement has been made in the text.

Jack Umenberger, BE (Hons 1)

22 August 2017

Abstract

Jack Umenberger, BE (Hons 1)
The University of Sydney

Doctor of Philosophy
August 2017

Convex Identification of Stable Dynamical Systems

This thesis concerns the scalable application of convex optimization to data-driven modeling of dynamical systems, termed *system identification* in the control community. Two problems commonly arising in system identification are model instability (e.g. unreliability of long-term, open-loop predictions), and nonconvexity of quality-of-fit criteria, such as simulation error (a.k.a. output error). To address these problems, this thesis presents convex parametrizations of stable dynamical systems, convex quality-of-fit criteria, and efficient algorithms to optimize the latter over the former.

In particular, this thesis makes extensive use of *Lagrangian relaxation*, a technique for generating convex approximations to nonconvex optimization problems. Recently, Lagrangian relaxation has been used to approximate simulation error and guarantee nonlinear model stability via semidefinite programming (SDP), however, the resulting SDPs have large dimension, limiting their practical utility. The first contribution of this thesis is a custom interior point algorithm that exploits structure in the problem to significantly reduce computational complexity. The new algorithm enables empirical comparisons to established methods including Nonlinear ARX, in which superior generalization to new data is demonstrated.

Equipped with this algorithmic machinery, the second contribution of this thesis is the incorporation of model stability constraints into the maximum likelihood framework. Specifically, Lagrangian relaxation is combined with the expectation maximization (EM) algorithm to derive tight bounds on the likelihood function, that can be optimized over a convex parametrization of all stable linear dynamical systems. Two different formulations are presented, one of which gives higher fidelity bounds when disturbances (a.k.a. process noise) dominate measurement noise, and vice versa.

Finally, identification of positive systems is considered. Such systems enjoy substantially simpler stability and performance analysis compared to the general linear time-invariant

(LTI) case, and appear frequently in applications where physical constraints imply nonnegativity of the quantities of interest. Lagrangian relaxation is used to derive new convex parametrizations of stable positive systems and quality-of-fit criteria, and substantial improvements in accuracy of the identified models, compared to existing approaches based on weighted equation error, are demonstrated. Furthermore, the convex parametrizations of stable systems based on linear Lyapunov functions are shown to be amenable to distributed optimization, which is useful for identification of large-scale networked dynamical systems.

Acknowledgements

I would first like to thank Ian Manchester for his patience, guidance, and instruction over the past four years. The influence Ian has had on the way I approach problems and communicate ideas cannot be overstated. I am especially appreciative of Ian's attention to detail, enthusiasm for new ideas, as well as the breadth and depth of his expertise. Furthermore, the many opportunities for international collaboration that I have enjoyed were all made possible by Ian. For all these things, I am very grateful.

To the Australian taxpayer, thank you for funding the APA on which I have subsisted for the past three and a half years.

To my friends and colleagues in Uppsala and Lund, including (but not limited to) Johan, Johan, . . . , Johan, Andreas, Niklas, Christian, and Carolina, thank you for your collaboration and hospitality. To Thomas, thank you for generously hosting me, as well as reminding me, time and time again, that the mushroom I had just picked was, in fact, poisonous. To Kostya, Petr, and Tanya, thank you for making me feel so welcome in Boston.

To my friends and colleagues at the ACFR (listed in order of my affection for you): Akash, Felix, Fletcher, Graham, Humberto, John, Karen, Mounir, Nathan, Oli, Suchet, Steve, and Warwick, thank you for making the past four years so enjoyable.

Justin, thank you for making my desk look tidy in comparison (I have taken even more heat from admin since your departure), and for your company during the many weekends spent at the office.

Will, there's nobody with whom I'd rather walk 100 kilometers through the bush (on four separate occasions). It's been a great pleasure working, and walking, alongside you for the past four years.

Mum, Dad, Georgia (and now, Wes too), thank you for all your love and support, and for taking just the right amount of interest in my research; and thanks again for planning Georgia's wedding around ACC.

The laws of mathematics are very commendable,
but the only law that applies in Australia
is the law of Australia.[1]

Malcolm Turnbull, 2017

[1] To the best of the author's knowledge, all proofs presented in this thesis are in full compliance with both the laws of mathematics and the law of Australia.

Contents

Nomenclature

Common sets

$\mathbb{R}$	The set of real numbers
$\mathbb{R}_+$ ($\mathbb{R}_{++}$)	The set of nonnegative (positive) real numbers
$\mathbb{R}^{n\times m}$	The set of real $n \times m$ matrices
$\mathbb{S}^n$	The set of symmetric $n \times n$ matrices
$\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$)	The cone of $n \times n$ positive semidefinite (positive definite) $n \times n$ matrices
$\mathbb{N}$	The natural numbers, $\{1, 2, 3, \dots\}$
$\mathbb{M}$	The set of Metzler matrices
$\mathbb{D}$	The set of diagonal matrices
$\mathcal{S}$	The set of all Schur stable matrices
$\mathcal{S}_+$	The set of all Schur stable element-wise nonnegative matrices

Vectors and matrices

$A(i, j)$	Denotes the entry in the ith row and j^{th} column of $A \in \mathbb{R}^{n\times m}$
a'	Transpose of the vector a
$\text{tr}(A)$	Trace of the matrix A
$r_\lambda(A)$	The spectral radius (largest magnitude of the eigenvalues) of the matrix A
$\text{vec}(A)$	Vectorization, stacks the columns of A into a vector
$\text{mat}(a)$	Maps the vector a to a matrix, i.e., $\text{vec}(\text{mat}(a)) = a$
$\text{diag}(a)$	Diagonal matrix with elements the vector a on diagonal
$\text{diag}(A)$	Vector with elements given by diagonal of matrix A
$>$ ($\geq$)	$A > B \implies A - B \in \mathbb{R}_{++}$ ($A \geq B \implies A - B \in \mathbb{R}_+$)
$\succ$ ($\succeq$)	$A \succ B \implies A - B \in \mathbb{S}_{++}$ ($A \succeq B \implies A - B \in \mathbb{S}_+$)
$\otimes$	Kronecker product
$a_{m:n}$	Sequence, $\{a_m, a_{m+1}, \dots, a_n\}$
q (q^{-1})	Forward (backwards) shift operator, e.g. $q^{-1}a_m = a_{m-1}$
I_n	The $n \times n$ identity matrix
$\mathbf{1}$	Vector with all elements equal to one
$\star$	denotes entires in a vector or matrix that we do not care about

Norms

$\lvert \cdot \rvert$	Absolute value (applies element-wise)
$\lVert \cdot \rVert_\sigma$	σ-norm, i.e., for $a \in \mathbb{R}^n$ $\lVert a \rVert_\sigma := \left(\sum_{i=1}^n \lvert a(i) \rvert^\sigma \right)^{1/\sigma}$
$\lVert \cdot \rVert_F$	Frobenius norm, i.e., $\lVert A \rVert_F = \operatorname{tr}(AA')$ for $A \in \mathbb{R}^{n \times m}$
$\lvert a \rvert_Q^2$	Shorthand for $a'Qa$

Probability

$\sim$	Sampled from or distributed according to
$\mathcal{N}(\mu, \Sigma)$	Multivariate Guassian with mean μ and covariance Σ
E	Expected value
$\mathbb{P}$	Probability
$\bar{\mathrm{E}}\left[x_t\right]$	Shorthand for $\lim_{t \to \infty} \frac{1}{T} \sum_{t=1}^T \mathrm{E}\left[x_t\right]$
var	Variance
i.i.d.	independently and identically distributed

Miscellaneous

a.k.a.	also known as
$:=$	Definition
§	section or chapter (used for brevity in references)
sig. fig.	significant figures
s.t.	such that
w.r.t.	with respect to

Abbreviations

ADMM	alternating direction method of multipliers
ARX	AutoRegressive with eXogenous inputs
BFGS	Broyden-Fletcher-Goldfarb-Shanno
EE	equation error
EEW	weighted equation error
EM	expectation maximization
EMDL	EM with latent Disturbances and Lagrangian relaxation
EMSL	EM with latent States and Lagrangian relaxation
FIR	finite impulse response
GP	Gaussian Process
GP-SSM	Gaussian process state space model
IPM	interior point method

IQC	integral quadratic constraint
KKT	Karush-Kuhn-Tucker
KYP	Kalman-Yakubovich-Popov
LGSS	linear Gaussian state space
LP	linear program
LMI	linear matrix inequality
LR	Lagrangian relaxation
LREE	Lagrangian relaxation of equation error
LRSE	Lagrangian relaxation of simulation error
LSTM	long short-term memory
LTI	linear time-invariant
MA	moving average
MAP	maximum a posterior
MC	Monte Carlo
ML	maximum likelihood
MM	minorization maximization
NARX	nonlinear ARX
NMF	nonnegative matrix factorization
NP	non-polynomial
PDF	probability density function
PEM	prediction error method
PR	positive-real
QP	quadratic program
RIE	Robust Identification Error
RNN	recurrent neural network
RTS	Rauch-Tung-Striebel
SDP	semidefinite program
SISO	single input, single output
SM	set membership
SMC	sequential Monte Carlo
SM	set membership
SNR	signal-to-noise ratio
SOS	sum-of-squares
SVD	singular value decomposition

Chapter 1

Introduction

This thesis concerns the building of approximate models of dynamical systems from measured data, a.k.a. *system identification*. The motif that unites all of the technical developments and contributions here within is the scalable application of convex optimization to this task. In this introductory chapter, we motivate the system identification problem, hint at some of its main difficulties, and outline the contributions of this thesis to address these difficulties.

1.1 Data driven modeling

Building mathematical models from measured data is a task of fundamental importance in many areas of science and engineering. Models are useful for making predictions, optimizing design, developing automatic control systems, and furthering our understanding of the universe that we inhabit; consider the early models of the motion of the planets, and the revelation that the earth, in fact, orbits the sun. Given this prevalence, it is not surprising that the ideas, theory, and algorithms for mathematical modeling have come from a variety scientific communities, including: statistics and statistical learning, econometrics and time series analysis, and recently, machine learning and artificial neural networks.

This thesis is concerned with the modeling of *dynamical* systems. Roughly speaking, a system is said to be dynamic if what has happened in the past affects future behavior, i.e., the mapping from inputs to outputs has memory. In the control community, estimation of dynamical systems from measured data is called *system identification*, although the process of building a model may also be referred to as *training*, *fitting* or *learning*. Figure 1.1 offers a graphical depiction of the key elements comprising the system identification problem. In what follows, we briefly describe these key elements, and touch on the challenges associated with each; or, more precisely, the challenges addressed in this thesis. For a more thorough discussion, refer to Chapter 2.

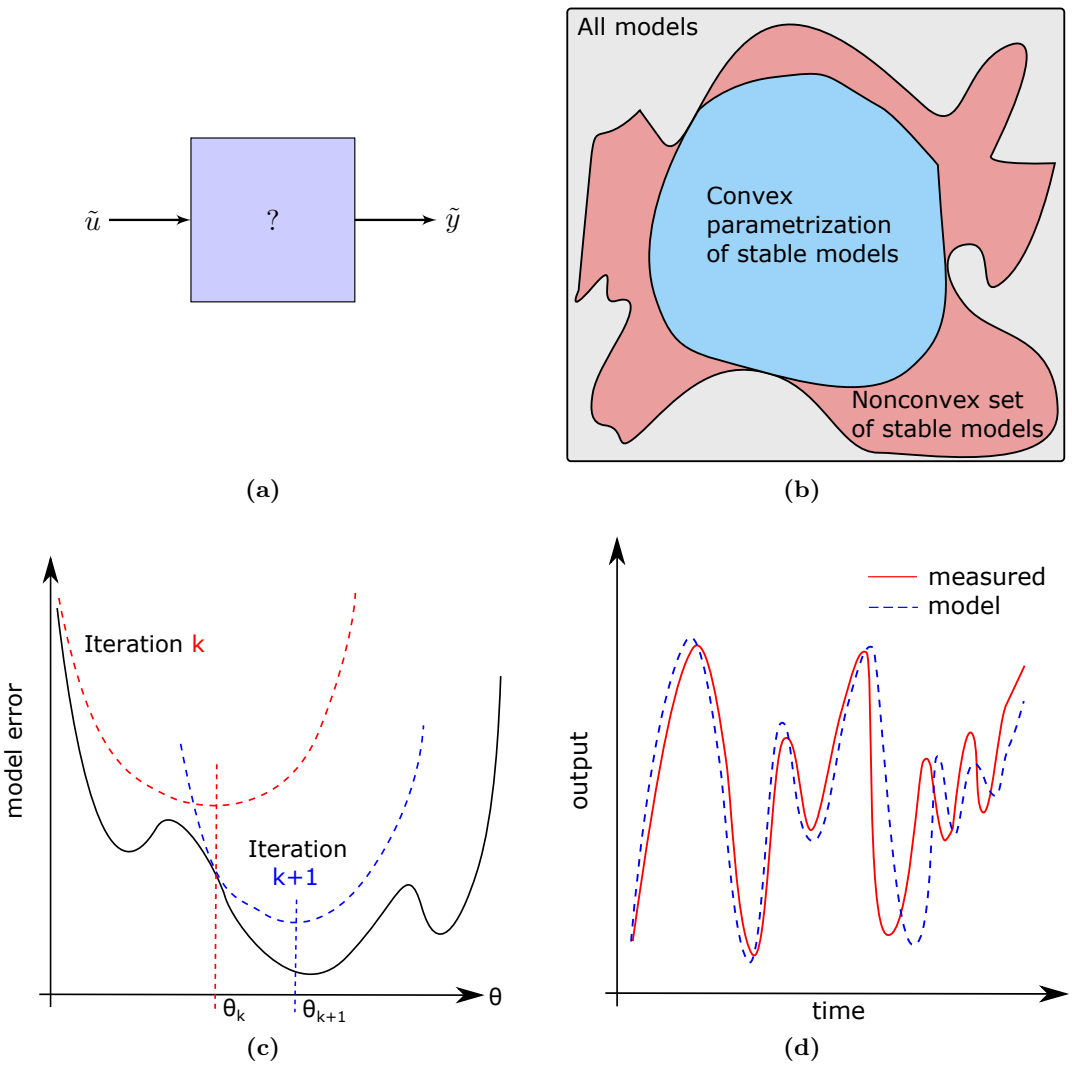


Figure 1.1 – Cartoon depiction of the system identification problem. (a) The high-level objective of a system identification problem. Given measured inputs, $\tilde{u}$, and outputs $\tilde{y}$, find a dynamical system that best fits this data. (b) A set of models to search over. Useful properties, such as stability, are defined by nonconvex constraints. A major theme of this thesis is convex parametrizations of stable models, that are easier to search over. (c) Many quality-of-fit criteria, e.g. measures of model error, are nonconvex in the model parameters, θ. Optimization of such criteria proceeds iteratively, by a sequence of approximations. A major theme of this thesis is convex approximations to nonconvex quality-of-fit criteria, that are (hopefully) robust to local minima. (d) A model should have good predictive performance for a variety of inputs, not just good fit to the training data. The process of testing this is called *validation*, and involves compared the output of the model to measured data not used for training.

Figure 1.1(a) summarizes the high-level objective of a system identification problem: given observations, or measurements, of the inputs to and outputs from some dynamical system, find a mathematical model that 'best fits' this data. There can be many challenges associated with merely attaining the data for identification, such as: choosing an appropriate input to excite the system, selecting the relevant outputs to observe, and using the right sensors or recording methods to obtain accurate measurements. This process of *experiment design* is an important topic in its own right, c.f., [76, 77, 197]; however, in this thesis we will assume that the training data has already been collected.

Once the data has been collected, the second key element is a set (or parametrization) of models to search over; this is depicted in Figure 1.1(b). As a concrete example, consider the familiar linear discrete-time-invariant system

$$x_{t+1} = Ax_t + Bu_t \tag{1.1}$$

$$y_t = Cx_t + Du_t. \tag{1.2}$$

Here, the unknown model parameters to be estimated are the matrices A, B, C, D. The linear time-invariant (LTI) system in (1.1) is deterministic; however, in reality system behavior is almost always *uncertain*, due to random [1] disturbances that affect the dynamics, as well as measurement (e.g. sensor) noise. In general then, the models that we fit are probability distributions. The emphasis of Figure 1.1(b), however, is on the *convexity* of the model set. In any optimization problem, convexity is one of the key properties that determines tractability. Broadly speaking, convex problems are easier to solve than nonconvex ones, because every locally optimal solution is also globally optimal; c.f. Section 2.2. In practice, many of the properties we desire for the identified model, e.g. stability during long-term open-loop simulations, require constraints that lead to a nonconvex model set. Developing (convex) parametrizations of models with useful properties has been recognized as an important open problem [131, §4.1], and plays a prominent role in this thesis.

To quantify the notion of the 'best model' we need some kind of 'quality-of-fit' criterion. Typically, some measure of the mismatch between the predictions from the model and the observed outputs (e.g. prediction error) is used. We then seek the model that achieves the lowest error, which is depicted in Figure 1.1(c) as numerical minimization problem. In some special cases, this minimization can be carried out analytically. More often though, an iterative procedure is required. This is complicated by the fact that many quality-of-fit criteria (especially those that capture the long-term fit of model) are nonconvex functions of the model parameters. The need for formulations of the identification problem that are robust to local minima has also been recognized as an important topic for further research [131, §4.2]. Even when the optimization is convex, scalability of the numerical algorithms (i.e., computational complexity that does not explode with problem size) is not guaranteed. In fact, the development of scalable algorithms for convex optimization is a motif that

[1] What is considered 'random' depends on how much one chooses to model; e.g., the influence of unmodeled dynamics may be approximated by random disturbances.

underpins most of the contributions of this thesis.

Finally, it is important that the identified model has good predictive performance in response to a variety of inputs, not just good fit to the data used for training. This property is referred to as the *generalizability* of the model, and the process of testing this is called *validation*. If the model performs well on training data, but poorly during validation (i.e., on data not used for training) then the model may be described as 'overfit'. Roughly speaking, this occurs when the model is 'too flexible', and the parameters are tuned to imitate the random disturbances in the measurements, rather than the underlying signal. In such a situation, it may be necessary to collect more training data, use *regularization* to penalize and avoid excessive model flexibility, or choose another model structure altogether.

1.2 Principle contributions

The central theme of this thesis is the efficient and scalable application of convex optimization to problems arising in system identification, specifically, model instability and nonconvexity of quality-of-fit criteria. With this in mind, the principle contributions of this thesis can be summarized within the context of the following three overlapping lines of research.

Recently, the technique of Lagrangian relaxation has been used to generate convex approximations to simulation error (a.k.a output error) that can be optimized via semidefinite programming; c.f. Section 2.4. The contributions of this thesis, presented in Chapter 3, are as follows:

- Specialized path-following interior point algorithms for Lagrangian relaxation of simulation error (in the linear case) and linearized simulation error (in the nonlinear case) are developed.
- These custom solvers have computational complexity that grows as a *linear* function of the number of data points used for model fitting, compared to the *cubic* growth exhibited by general-purpose semidefinite programming solvers.
- These efficient solvers enable Lagrangian relaxation to be applied to problems of a scale hitherto intractable. The method of Lagrangian relaxation is empirically evaluated and state-of-the-art performance against established methods, such as nonlinear ARX, is demonstrated.

The Maximum Likelihood criterion is used extensively in a wide range of statistical learning problems, including system identification. However, traditional formulations give no regard to the stability of the identified models. The contributions of this thesis, presented in Chapter 4, are as follows:

- Model stability constraints are incorporated into maximum likelihood identification of dynamical systems.
- Specifically, Lagrangian relaxation and expectation maximization (EM) are combined to generate new bounds on the likelihood that can be optimized over convex parametrizations of stable models; in the linear case, this parametrization includes all stable linear models.
- In this framework, a new formulation of the EM algorithm for system identification is presented. This new approach uses disturbances, rather than the usual choice of states, as the latent variables.
- The relationship between choice of latent variables (i.e. disturbances or states) and bound fidelity is studied. Theoretical and empirical studies show that bounds based on latent states offer greater fidelity when the disturbances are more significant than measurement noise; the converse is true for the latent disturbances formulation when the situation is reversed.
- It is also shown that latent disturbances provide the most broadly applicable formulation of EM for identification of models in which the disturbance covariance is rank-deficient, a.k.a, *singular state space models*.

Dynamical systems for which nonnegative inputs imply nonnegative states are said to be internally positive. Such systems appear frequently in applications, and enjoy simpler stability analysis compared to the generic LTI case. The contributions of this thesis to the identification of positive systems, presented in Chapter 5, are as follows:

- Two new convex parametrizations of all stable positive systems are derived. One is defined by a linear matrix inequality and generalizes existing approaches; the other is defined by a polytopic set.
- Two new convex quality-of-fit criteria, compatible with the above parametrizations, are derived. One is based on Lagrangian relaxation of equation error (a.k.a. the least squares criterion); the other is a convex upper bound on the ℓ_1 norm of simulation error (a.k.a. output error).
- It is shown how the new polytopic parametrization of stable positive systems permits distributed optimization of the quality-of-fit criteria. Specifically, the linear constraints defining the polytopic parametrization allow the identification problem to be more readily decomposed into simpler subproblems, compared to existing LMI conditions for stability.

1.3 Publications

Research publications relating to this thesis are listed below, in reverse chronological order.

J. Umenberger, J. Wågberg, I.R. Manchester, T.B. Schön. Maximum likelihood identification of stable linear dynamical systems. *Automatica*. 2017. *Conditionally accepted for publication*.

J. Umenberger, I.R. Manchester. Specialized interior-point algorithm for stable nonlinear system identification. *IEEE Transactions on Automatic Control*. 2017. *Under review*.

C. Grussler, **J. Umenberger**, I.R. Manchester. Identification of externally positive systems. To appear in *Proceedings of the IEEE Conference for Decision and Control (CDC)*. 2017.

J. Umenberger, I.R. Manchester. Scalable identification of stable positive systems. In *Proceedings of the IEEE Conference for Decision and Control (CDC)*. 2016.

J. Umenberger, I.R. Manchester. Specialized algorithm for identification of stable linear systems using Lagrangian relaxation. In *Proceedings of the American Control Conference (ACC)*. 2016.

J. Umenberger, J. Wågberg, I.R. Manchester, T.B. Schön. On identification via EM with latent disturbances and Lagrangian relaxation. In *Proceedings of the IFAC Symposium on System Identification (SYSID)*. 2015.

1.4 Thesis structure

This thesis proceeds as follows. **Chapter 2** provides a summary of the state-of-the-art in system identification. The purpose of this chapter is to survey the literature in such a way so as to contextualize the contributions of this thesis, as well as equip the reader with background information sufficient to understand the technical developments of the succeeding chapters.

In **Chapter 3** we present specialized interior-point algorithms for identification of stable nonlinear dynamical systems, via Lagrangian relaxation of simulation error. We explain the structural properties of Lagrangian relaxation exploited by the specialized algorithms, and demonstrate, both theoretically and empirically, a significant reduction in computational complexity compared to generic solvers; namely, linear instead of cubic growth in the

number of data points used for training. We demonstrate the performance of the algorithm via three case studies: a simulated nonlinear mechanical system, a real two-tank benchmark identification problem, and simulated linear flexible beams. Superior performance over state-of-the-art methods, such a nonlinear ARX, is achieved. We also investigate the apparent 'regularizing' effect of model stability constraints in identification of nonlinear systems.

We then extend some of these ideas to a stochastic setting in **Chapter 4**, by using Lagrangian relaxation to incorporate model stability constraints into the maximum likelihood framework. We combine Lagrangian relaxation with the expectation maximization (EM) algorithm to derive tight lower bounds on the likelihood that can be optimized over a convex parametrization of all stable linear models by semidefinite programming. We explore the effects that different choices of latent variables have on the fidelity of these bounds, and show that latent states lead to better performance when system disturbances (a.k.a. process noise) dominate measurement noise; conversely, latent disturbances perform better when the situation is reversed. These conclusions are supported by theoretical analysis and extensive simulation studies.

In **Chapter 5** we turn our attention to identification of internally positive systems. We leverage the simplified stability conditions enjoyed by such systems to derive two new convex parametrizations of stable positive systems, as well as convex quality-of-fit criteria compatible with these parametrizations. Extensive numerical simulations demonstrate superior performance of criteria derived from Lagrangian relaxation, compared to existing approaches based on weighted equation error. Distributed algorithms for the optimization of these criteria are also presented.

Finally, **Chapter 6** concludes the thesis by discussing some open problems and suggesting directions for future research.

Chapter 2

Background

In this chapter we attempt to provide something of a summary of the state-of-the-art in system identification; however, due to the maturity of the subject, a comprehensive review is beyond our scope. Rather, there are two objectives for this chapter. First, to survey the literature in such a way so as to contextualize the contributions of this thesis. Second, to equip the reader with background information sufficient to understand the technical developments of the succeeding chapters.

2.1 A framework for system identification

System identification is a mature subject, with many textbooks covering both the 'art' and 'science' of building approximate models of dynamical systems from measured data, e.g., [124, 129, 132, 159]. As discussed in Chapter 1, modeling is a task of such fundamental importance in so many fields of science and engineering, that many approaches and much terminology has developed. This can make navigating the immense literature a daunting task. In this section, we present an 'optimization-based approach' to system identification, as a way of systematically reviewing the literature. There are five elements to this optimization-based framework:

1. *Data collection*, i.e., obtaining sequences of inputs $\tilde{u}_{1:T} = \{\tilde{u}_t\}_{t=1}^T$ and outputs $\tilde{y}_{1:T} = \{\tilde{y}_t\}_{t=1}^T$ from the system being modeled.

2. A *set* or *parametrization* of models to search over. The model set specifies the structure of the model, along with the unknown parameters to be found. We shall use θ to denote the unknown parameters of a model, and $\Theta \subset \mathbb{R}^{n_\theta}$ to denote the set of possible values that θ can take.

3. A *quality-of-fit metric* or *criterion* to quantify the notion of the 'best' or 'optimal' model. Most quality-of-fit criteria are expressed in terms of model error, e.g. prediction error, and so we will often talk about the quality-of-fit metric as a *cost function* to

be minimized. The likelihood, as used in maximum likelihood methods, is an obvious exception; however the different between 'minimizing a cost function' and 'maximizing a value function' is superficial. We shall use $\mathcal{E}(\theta)$ to denote a generic quality-of-fit metric.

4. An *algorithm* to search over the model set Θ to find the model that achieves the best value of the quality-of-fit metric, i.e., an algorithm to solve the optimization problem: $\hat{\theta} := \arg\min_{\theta \in \Theta} \ \mathcal{E}(\theta)$.

5. *Validation*, i.e., checking that the identified model has good predictive performance for a variety of inputs, not merely good fit to the data used for training, [129, §16].

These five elements can be thought of as steps in a system identification task. Before proceeding, some comments on each are in order.

The data collection process incorporates tasks such as: the design of an input signal so as to appropriately excite the system, i.e., elicit the behavior that we wish to capture in the model; the selection of suitable signals to measure (to constitute the system outputs); the choice of frequency at which to sample the measured signals, so as to capture the relevant behavior; and any filtering or noise suppression. Despite the ongoing improvement of sensor technology, in some applications even measuring the signals of interest is a challenge, e.g., consider the carbon nanowires [96] required to measure electrical activity of neurons modeled in [144]. Many of these considerations fall under the subject of *experiment design*, which is an important research topic in its own right, [76, 77, 142, 197]. In this thesis, however, we do not consider experiment design; all of the methods we develop assume that the training data $\{\tilde{u}_{1:T}, \tilde{y}_{1:T}\}$ is already available.

Rather, we shall focus most of our attention on elements 2-4: parameterizations of models, quality-of-fit criteria, and search algorithms. It is to these three areas that this thesis makes its contributions. Notice that we assume a finite parametrization of the models, i.e., $\theta \in \Theta \subset \mathbb{R}^{n_\theta}$. Indeed, this thesis considers the problem of generating point estimates of parametric models from data. Approaches that do not fit quite so neatly into this paradigm, such as Bayesian methods, set membership, frequentist analysis of confidence intervals, and nonparametric methods are discussed in Section 2.5.1. Furthermore, we shall be concerned primarily with black-box modeling. Black-box modeling is characterized by flexible model parametrizations that can (hopefully) approximate a wide variety of dynamical systems. The goal is to accurately model input-output behavior, without regard for the physical 'explainability' of the structure of the identified model. This may be contrast to 'gray-box' modeling, which incorporates a priori information to develop model structures in which the unknown parameters (typically) have some physical meaning, or 'white-box' modeling, in which models are constructed from first principles; c.f., [131, §5] for a more thorough discussion.

Validation is listed as the final step of the modeling task; however, if the performance of the model on validation data is not adequate, it may be necessary to revise one (or all) of the

preceding steps (e.g., collect more data, tweak the model structure, choose a new quality-of-fit criterion and/or search algorithm), and repeat the process. In essence, validation tests the ability of a model to generalize (from data used for training to data not used during training). A useful tool for improving 'generalizability' is regularization, which penalizes or constrains model complexity (in some sense) to help avoid overfitting, c.f. Section 2.1.2 for further discussion. Although this thesis does not propose new methods for validation, we do explore the apparent regularizing effect of model stability constraints in Chapter 3.

2.1.1 Common model structures

Once measurements of the input and output have been collected, the first step in an optimization based approach to system identification is to select an appropriate set of models to search over.

Deterministic models

In essence, a system is dynamic if future behavior depends on past behavior. One approach then, is to model the output of a dynamical system as a finite truncation of previous inputs

$$y_t = f_{\mathrm{p}}(\theta, u_t, u_{t-1}, u_{t-2}, \ldots, u_{t-d}). \tag{2.1}$$

When f_{p} is a nonlinear function of past inputs, (2.1) is called a nonlinear finite impulse response (FIR) model. Common nonlinear functions include Wiener and Volterra series [52, 202]. When f_{p} is a linear function of the inputs, we have the familiar (linear) FIR model as a special case; e.g, $y_t = \sum_{i=0}^{n} b_i u_{t-i}$ where $\{b_t\}_{t=0}^{n}$ denotes the impulse response of the nth order single input, single output (SISO) system.

FIR models have a number of attractive properties for identification. For one, model stability is guaranteed; it is clear that setting $u_t = 0$ for $t > \tau$ will result in y_t decaying to a constant value $f_{\mathrm{p}}(\theta, 0, \ldots, 0)$ in finite time. Furthermore, fitting is usually straightforward; e.g., when f_{p} is linear in the model parameters, minimization of simulation error (c.f. Section 2.1.2) is convex. Unfortunately, FIR models are known to be very inefficient when modeling resonant systems, and are incapable of capturing some nonlinear behaviors such as limit cycles (stable oscillations which live on the edge of instability).

In contrast, models with feedback offer a more parsimonious representation of dynamic behavior. Perhaps the most simple example of feedback in a dynamical system is given by the linear finite difference equation

$$y_t + a_1 y_{t-1} + \cdots + a_{n_a} y_{t-n_a} = b_1 u_{t-1} + \cdots + b_{n_b} u_{t-n_b}. \tag{2.2}$$

By introducing the backwards shift operator q^{-1}, such that $q^{-1}u_t = u_{t-1}$, we can define the polynomials $A(\theta, q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}$ and $B(\theta, q) = b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b}$, which

gives a transfer function representation of (2.2)

$$y_t = \frac{B(\theta, q)}{A(\theta, q)} u_t = G(\theta, q) u_t, \tag{2.3}$$

where $\theta = \{a_1, \ldots, a_{n_a}, b_1, \ldots, b_{n_b}\}$.

The finite difference model (2.2) may be augmented to incorporate system nonlinearity. One approach is to introduce static nonlinearities at the input (Hammerstein), and/or output (Weiner) of the linear dynamical system, leading to so-called Hammerstein-Weiner models, [23]. Alternatively, one may introduce nonlinearity directly into the feedback path, resulting in a nonlinear finite difference equation of the form

$$y_t = f_{\text{fd}}(\theta, y_{t_1}, \ldots, y_{t-n_a}, u_{t-1}, \ldots, u_{t-n_b}). \tag{2.4}$$

Popular functional forms for f_{fd} include polynomials, neural networks [39], wavelets [208], as well as non-parametric kernels [178].

In a state space model, feedback is introduced via an internal state variable, $x_t \in \mathbb{R}^{n_x}$. The state summarizes or condenses all of the past behavior (i.e., input and outputs) into a single quantity that is sufficient to compute future behavior. A general deterministic state space model takes the form

$$x_{t+1} = a_\theta(x_t, u_t), \tag{2.5}$$

$$y_t = g_\theta(x_t, u_t). \tag{2.6}$$

Here $a_\theta : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \mapsto \mathbb{R}^{n_x}$ defines the system dynamics, i.e. the state transition, whereas $g_\theta : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \mapsto \mathbb{R}^{n_y}$ is a static mapping from states and inputs to the observable output. A special case of (2.5) is the familiar LTI state space model,

$$x_{t+1} = Ax_t + Bu_t, \tag{2.7}$$

$$y_t = Cx_t + Du_t, \tag{2.8}$$

which gained popularity after Kalman's celebrated work on prediction and linear quadratic optimal control. State space models offer a very general description of system dynamics; in fact, the state space model in (2.5) is able to incorporate all of the previously discussed model structures, e.g., for finite difference models, one can take a truncated history of outputs as the state variable, and set g such that $y_t = x_t$.

Time-invariant state space models can also incorporate some[1] of the dynamical systems that appear in machine learning. For instance, a recurrent neural network (RNN) is a deep feedforward network, in which all of the layers share the same weights [116]. A generic RNN can be represented as a nonlinear state space model, c.f., e.g., [173, §1]. RNNs have proven

[1] Time-varying state space models are required to represent neural networks in which the weights in each layer are independent of one another.

effective in applications that deal with inputs sequentially, such as speech and language processing. The internal state vector maintained by the 'hidden layers' of the network stores information about past observations (e.g. words in a sentence); however, theoretical and empirical results suggest that standard RNNs experience difficulty storing information over long time horizons [12]. To address this limitation, recent approaches augment the network with a 'memory module', leading to long-short term memory (LSTM) networks [91], Neural Turning Machines [78], and memory networks [254].

Probabilistic models

Often it is desirable, or necessary, to model the stochastic properties of a system, e.g. random disturbances that affect the dynamics, and/or measurement noise that affects the output y that we observe. We can augment the deterministic models introduced above with stochastic processes, to form a complete (but approximate) probabilistic description of system behavior. Once more, the simplest approach involves the finite difference equation model of (2.2). Adding a white-noise disturbance w_t directly into the difference equation yields an *equation error* model of the form

$$A(\theta, q)y_t = B(\theta, q)u_t + w_t. \tag{2.9}$$

Such a model is termed AutoRegressive with eXogenous inputs (ARX), due to the dependence of the output on both past outputs, $A(\theta, q)y_t$ (AR), and external (or exogenous) inputs, $B(\theta, q)u_t$. Early work on identification of ARX systems can be found in [10, 92].

In some applications, it may be appropriate to model the disturbance as a (weighted) moving average (MA) of white noise, leading to a so-called ARMAX model of the form

$$A(\theta, q)y_t = B(\theta, q)u_t + C(\theta, q)w_t, \tag{2.10}$$

with $C(\theta, q) = 1 + c_1 q^{-1} + \cdots + c_{n_c} q^{-n_c}$. In much the same way, stochastic processes may be introduced to the deterministic nonlinear difference equation in (2.4), leading to probabilistic nonlinear ARX (NARX) or NARMAX models [24].

In some settings, it may be more appropriate to approximate the system by a deterministic model with the output, rather than the state transition, corrupted by measurement noise, e.g., the transfer function in (2.3) with additive white measurement noise v_t,

$$y_t = \frac{B(\theta, q)}{A(\theta, q)} u_t + v_t, \quad v_t \sim \mathcal{N}(0, \Sigma_v). \tag{2.11}$$

Such a model is said to have an *output error* structure, as the noise corrupts the observed outputs, rather than the state transition. Identification of output error models typically lead to simulation error minimization problems: c.f. Section 2.1.2 and Chapter 3. In the linear setting, a very general parametrization of probabilistic models is given by the so-called

Box-Jenkins structure [27],

$$y_t = \frac{B(\theta, q)}{A(\theta, q)} u_t + \frac{C(\theta, q)}{D(\theta, q)} v_t, \tag{2.12}$$

with $D(\theta, q) = 1 + d_1 q^{-1} + \cdots + d_{n_d} q^{-n_d}$.

As in the deterministic case, the most general description of finite dimensional (and finitely parametrized) dynamical systems is offered by nonlinear state space models. A general probabilistic state space model can be expressed as

$$x_{t+1} \sim p_\theta^a(x_{t+1}|x_t, u_t), \tag{2.13}$$

$$y_t \sim p_\theta^g(y_t|x_t, u_t). \tag{2.14}$$

As a concrete example, in engineering applications it is common to augment the deterministic state space model (2.5) with additive disturbances, w_t, and measurement noise, v_t, leading to

$$x_{t+1} = a_\theta(x_t, u_t) + w_t, \quad y_t = g_\theta(x_t, u_t) + v_t. \tag{2.15}$$

Furthermore, if the additive noise is normally distributed, i.e. $w_t \sim \mathcal{N}(0, \Sigma_w)$, then we have

$$p_\theta^a(x_{t+1}|x_t, u_t) = \mathcal{N}\left(x_{t+1} - a_\theta(x_t, u_t), \Sigma_w\right), \quad p_\theta^g(y_t|x_t, u_t) = \mathcal{N}\left(y_t - g_\theta(x_t, u_t), \Sigma_v\right).$$

A special case of (2.15) occurs when the dynamics are linear, leading to a so-called linear Gaussian state space (LGSS) model,

$$x_{t+1} = Ax_t + Bu_t + w_t, \tag{2.16}$$

$$y_t = Cx_t + Du_t + v_t. \tag{2.17}$$

2.1.2 Quality-of-fit criteria

The next step of the identification problem is to find the model within the model set Θ that best fits the measured data. We quantify the 'best' model as that which optimizes (i.e. minimizes or maximizes) some quality-of-fit metric or criterion, $\mathcal{E}$. In this section, we survey some common quality-of-fit criteria. It should be noted that the choice of model structure typically has a large influence on the quality-of-fit metric that is optimized.

Maximum likelihood

Since its introduction by Fisher [62] at the beginning of the 20th century, the maximum likelihood (ML) criterion has remained an extremely popular approach to parameter identification in a range of statistical inference problems; c.f [56, §4]. A probabilistic model of a dynamical system, parameterized by θ, defines a probability density function (PDF) for the observed input and outputs, $p(\theta, u_{1:T}, y_{1:T})$. For a specified value of θ, this PDF gives

the likelihood with which observations of the system will take on certain values, i.e.,

$$\mathbb{P}(\{u_{1:T}, y_{1:T}\} \in A) = \int_{\{\bar{u}_{1:T}, \bar{y}_{1:T}\} \in A} p(\theta, \bar{u}_{1:T}, \bar{y}_{1:T})\ d\bar{u}_{1:T} d\bar{y}_{1:T}. \tag{2.18}$$

Given observed data $\{\tilde{u}_{1:T}, \tilde{y}_{1:T}\}$, i.e. realizations of the random variables $\{u_{1:T}, y_{1:T}\}$, $p(\theta, \tilde{u}_{1:T}, \tilde{y}_{1:T})$ is called the *likelihood function*. The ML parameter estimate is then given by

$$\hat{\theta}^{\mathrm{ML}} = \arg\max_{\theta}\ p(\theta, \tilde{u}_{1:T}, \tilde{y}_{1:T}). \tag{2.19}$$

In the system identification literature, the likelihood is often written as $p_\theta(y_{1:T})$, where explicit dependence on $u_{1:T}$ has been dropped for brevity. It is worth emphasizing that the likelihood $p_\theta(y_{1:T})$ is a function of θ when optimized in (2.19).

Under the assumption that the observed data was generated by a model in the model set Θ, the ML estimate $\hat{\theta}^{\mathrm{ML}}$ has a number of desirable properties, such as strong consistency and asymptotic normality, c.f. [129, §8 and §9]. In fact, the asymptotic covariance of the ML parameter estimate converges, as $T \to \infty$, to the Cramér-Rao lower bound. In this sense, $\hat{\theta}^{\mathrm{ML}}$ has the best possible asymptotic properties of all unbiased estimators, c.f. [129, §9.3] Despite this, there are two important caveats for ML identification. First, the assumption that the model set is correctly specified is unrealistic in application; real systems are almost always more complicated than the models we construct. This, in part, motivated the intense study of prediction error methods, discussed next. Second, finding $\hat{\theta}^{\mathrm{ML}}$, i.e. achieving the global maximum of (2.19), is not always straightforward. For nonlinear, non-Gaussian models, the likelihood can be difficult to even compute, let alone differentiate, and local maxima are not uncommon. ML identification is addressed in Chapter 4.

Prediction error

The basic principle of comparing predicted output to measured data as a criterion for model fidelity has a long history in the system identification literature. Early work by Akaike [3, 4] recognized the underlying connections between such criteria and maximum likelihood methods, as well as the utility of prediction error criteria in the absence of a true model. The term *prediction error method* (PEM) appears to have been introduced in a series of papers [126, 127] by Ljung, who has since championed the approach. The method centers around the *prediction error*, which is defined as the difference between the observed output $\tilde{y}_t$ and the (one-step-ahead) prediction $\hat{y}_t(\theta)$ from the model, i.e.,

$$\epsilon_t^{\mathrm{p}}(\theta) := \tilde{y}_t - \hat{y}_t(\theta). \tag{2.20}$$

It should be emphasized that $\hat{y}_t(\theta)$ need not be the *optimal* (e.g. unbiased, minimum variance) one-step-ahead predictor for the model; as discussed in [129, §7.1], there is considerable freedom in choosing the predictor. The prediction error estimator is then given

by

$$\hat{\theta}^{\mathrm{PE}} = \arg\min_{\theta} \left\{ \mathcal{E}^{\mathrm{pe}}(\theta) := \frac{1}{T} \sum_{t=1}^{T} \ell\left(L(q)\epsilon_t^{\mathrm{p}}(\theta)\right) \right\}, \tag{2.21}$$

where $\ell(\cdot)$ is some scalar valued function, such as a norm, and $L(q)$ is a stable linear filter, selected by the user to attenuate behavior not critical to the modeling task, e.g., to suppress high-frequency noise or eliminate low-frequency drift. On occasion, especially for the purpose of asymptotic analysis, we will write $\hat{\theta}_T^{\mathrm{PE}}$ to emphasize dependence of the resulting parameter estimate on the number of data points, T.

Although ML methods certainly predate PEM, the latter can be viewed as a general framework that encompasses many popular approaches, depending on the statistical properties of the model set and choice of function $\ell(\cdot)$. For instance, under the assumption of a true model θ_0 in the model set Θ, choosing $\ell(\cdot) = -\log p_e(\cdot)$ recovers the ML estimate, i.e. $\hat{\theta}^{\mathrm{PE}} = \hat{\theta}^{\mathrm{ML}}$, where $p_e(\cdot)$ is the PDF for the residuals $e_t = \epsilon_t^{\mathrm{p}}(\theta_0)$.

The popularity of PEM is largely due to its desirable asymptotic properties in the absence of a true system description within the model parametrization. Under mild assumptions about the model set and given sufficiently informative data, PEM has been shown to provide the best possible model estimate, from with the model set Θ, in the sense that

$$\hat{\theta}_T^{\mathrm{PE}} \to \arg\min_{\theta\in\Theta} \ \bar{\mathrm{E}}\left[\ell(L(q)\epsilon_t^{\mathrm{p}}(\theta))\right] \qquad \text{w.p. 1 as } T \to \infty, \tag{2.22}$$

where $\bar{\mathrm{E}}\left[f_t\right] := \lim_{t\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathrm{E}\left[f_t\right]$, c.f. [128, Lemma 3.1], and [129, §8.3]. This property is, of course, contingent on actually achieving the global minimum in (2.21), which is not always straightforward, e.g., due to capture in local minima during optimization.

Equation error

PEM can be viewed as a framework that encapsulates many popular approaches, including the least squares criterion, a.k.a. equation error minimization. Consider a linearly parametrized equation error model, as defined in §2.1.1. For a concrete example, let us use the ARX model in (2.9). The dynamics in (2.9) can be written in *regressor* form as

$$y_t = \phi_t'\theta + w_t$$

where $\phi_t' = [y_{t-1}, \ldots, y_{t-n_a}, u_{t-1}, \ldots, u_{t-n_b}]$ denotes a vector of *regressors*, and

$$\theta = [a_1, \ldots, a_{n_a}, b_1, \ldots, b_{n_b}]'$$

is the vector of model parameters. Suppose we choose a one-step-ahead predictor of the form

$$\hat{y}_t = \tilde{\phi}_t'\theta + \mu_t,$$

where $\tilde{\phi}_t' = [\tilde{y}_{t-1}, \ldots, \tilde{y}_{t-n_a}, \tilde{u}_{t-1}, \ldots, \tilde{u}_{t-n_b}]$ and μ_t is a known function of the observed data; c.f. [129, §7.3]. Taking $\ell(\cdot) = |\cdot|^2$ and $L(q) = 1$ in (2.21) leads to minimization of

$$\frac{1}{T}\sum_{t=1}^{T} |\epsilon_t^{\mathrm{p}}|^2 = \frac{1}{T}\sum_{t=1}^{T} |\tilde{y}_t - \hat{y}_t|^2 = \frac{1}{T}\sum_{t=1}^{T} |\tilde{y}_t - \tilde{\phi}_t'\theta - \mu_t|^2. \tag{2.23}$$

This cost function is often called the *least squares criterion*, or simply *equation error*. As (2.23) is convex (quadratic, in fact) in the model parameters, it is analytically minimized by the *least squares estimate*. It is worth emphasizing that the above developments are valid for any linearly parametrized equation error model, e.g. NARMAX. Linearity in the model parameters is what's important; the regressors themselves may be nonlinear functions of the problem data $\tilde{u}_{1:T}$ and $\tilde{y}_{1:T}$.

When the problem data $\tilde{y}_{1:T}$ is generated by an equation error model,

$$\tilde{y}_t = \tilde{\phi}_t'\theta_0 + w_t,$$

where θ_0 denotes the true model parameters, the least squares estimate is consistent, as long as the regressors are not correlated with the disturbances, i.e., $\mathrm{E}\left[\tilde{\phi}_t w_t\right] = 0$. However, in most applications these conditions are not satisfied, e.g., the disturbances may not be white noise, the outputs may be corrupted by measurement noise (as in 'errors in variables' problems, [212]), or the input sequence $\tilde{u}_{1:T}$ may come from a closed-loop feedback controller. In such cases, the least squares estimate is not consistent.

Minimization of equation error also appears in approaches not directly derived from the PEM framework. For example, in subspace methods one minimizes

$$\sum_{t=1}^{T} \|\tilde{y}_t - C\tilde{x}_t - D\tilde{u}_t\|^2 + \sum_{t=1}^{T} \|\tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t\|_2^2 \tag{2.24}$$

w.r.t. $\theta = \{A, B, C, D\}$, where $\tilde{x}_{1:T}$ are state estimates derived from a subspace algorithm, c.f. [239] and Section 2.1.3.

For models obtained by minimization of one-step-ahead prediction error, it is reasonable to expect good short term predictive performance. However, small equation error does not guarantee reliable long term open-loop predictive performance (e.g. due to accumulation of error at each time step during simulation).

Simulation error

When accurate open-loop predictions of long-term system behavior are required, it may be desirable to minimize *simulation error* (a.k.a. output error), defined as the difference between the measured output, $\tilde{y}_{1:T}$, and the open-loop simulated output of the model; e.g.

for the state space model (2.5), simulation error is defined as

$$\mathcal{E}^{\text{se}} := \sum_{t=1}^{T} \|\tilde{y}_t - y_t\|_2^2 \tag{2.25}$$

where $y_t = g(a(a(\ldots a(\tilde{x}_1, \tilde{u}_1) \ldots, \tilde{u}_{t-2}), \tilde{u}_{t-1}), \tilde{u}_t)$ is the simulated output of (2.5) beginning from the initial conditions $\tilde{x}_1$, which may be known *a priori* or estimated.

The definition of simulation error in (2.25) assumes a completely deterministic model, e.g., (2.5). For probabilistic models, such as (2.13), $\mathcal{E}^{\text{se}}$ is not well-defined. Having said this, in the PEM framework, there are close connections between simulation error and prediction error for output error model structures. For example, consider augmenting the deterministic model (2.5) with i.i.d. measurement noise of constant covariance. The result is an output error model of the form,

$$x_{t+1} = a(x_t, u_t), \quad y_t = g(x_t, u_t) + v_t, \quad v_t \sim \mathcal{N}\left(0, \operatorname{diag}(\sigma)\right). \tag{2.26}$$

A common one-step ahead predictor for this model is the simulated output, i.e., $\hat{y}_t(\theta) = g(a(\ldots a(\tilde{x}_1, \tilde{u}_1) \ldots, \tilde{u}_{t-1}), \tilde{u}_t)$. Minimization of prediction error in (2.21) (with $L(q) = 1$ and $\ell(\cdot) = \|\cdot\|_2^2$) is then equivalent to minimization of simulation error, and corresponds to maximum likelihood identification; c.f. [251], [22, §5].

Unlike equation error, e.g. (2.23), the dependence on the simulated output of the model renders $\mathcal{E}^{\text{se}}$ a nonconvex function of the model parameters, θ, making the search for the global minimum challenging. Even in the case of linear second order models, with finite T, the existence of poor local minima has been demonstrated [211].

Regularization

Due to the random disturbances affecting our measurements, any parameter estimate $\hat{\theta}$ is a random variable, governed by a probability density function with mean and variance. The difference between the mean value of the estimate and true model parameters (if such a true description exists) is referred to as the *bias*. Therefore, there are two contributions to the total error in a model: the bias and the variance. Roughly speaking, a model structure that is not sufficiently flexible will be unable to capture the true behavior of the system, leading to large bias. On the other hand, more flexibility in the model structure can increase the risk of over-fitting to the random disturbances in the data, thereby increasing the variance.

Regularization refers to the process of constraining or reducing model complexity (in some sense) to manage this bias-variance trade-off (i.e. prevent over-fitting) in statistical modeling [86]. Classical methods such as Tikhonov regularization, also known and ridge regression (shrinkage) in statistics [56, §7] and weight decay in machine learning [112], as well as subset selection (regressor pruning) have long been applied in nonlinear system identification [24, 99, 208].

In system identification, regularization typically involves the addition of some term that penalizes the size or complexity of the model parameters to one of the quality-of-fit criteria discussed above. For example, Tikhonov regularization of prediction error minimization takes the form

$$\min_{\theta} \ \mathcal{E}^{\mathrm{pe}}(\theta) + \delta\|\theta - \bar{\theta}\|_2^2, \tag{2.27}$$

which is just the usual PEM criterion from (2.21) plus a quadratic cost on deviation from some fixed parameter value, $\bar{\theta}$. At a conceptual level, there are three common justifications for including the regularization term; c.f., e.g., [129, §7.4] for further details:

i. The regularization term $\delta\|\theta - \bar{\theta}\|_2^2$ has the effect of 'pulling' the estimated parameters towards the fixed point $\bar{\theta}$, often chosen to be the origin, $\bar{\theta} = 0$. Roughly speaking, the parameters that have little influence on $\mathcal{E}^{\mathrm{pe}}$ will be pulled close to zero, and vice versa. This may be interpreted as a reduction in the 'effective' or 'efficient' number of parameters, thereby reducing the flexibility of the model and variance of the error.

ii. This first interpretation is also well supported by the Bayesian framework for identification, c.f. Section 2.5.1, in which the regularization term $\delta\|\theta - \bar{\theta}\|_2^2$ corresponds to a Gaussian prior, with mean $\bar{\theta}$ and covariance $\frac{1}{2T\delta}I$, on the model parameters.

iii. The Hessian of the regularized criterion in (2.27) is given by $\nabla^2\mathcal{E}^{\mathrm{pe}} + \delta I$. For models with high-dimensional θ, the addition of δI can improve the numerical conditioning of the Hessian, and ensuing optimization problem.

Recently, novel regularization strategies for the identification of linear systems have been developed, including nuclear norm regularization for subspace identification (e.g. [125]) and kernel methods for impulse-response modeling, surveyed in [179].

2.1.3 Subspace methods

Since their introduction in the 1990s, subspace methods have become an indispensable tool for the identification of linear dynamical systems. A number of early approaches, such as those presented in [97, 114, 239, 247], received a unified treatment in [240]; see also [241]. Since then, subspace methods have undergone continual development, including the study of statistical properties [11, 81, 98] as well as extensions to identification of closed loop systems [43, 45, 186, 246]; c.f. [185] for a recent survey on subspace methods.

In our framework, subspace methods straddle the boundary between 'quality-of-fit' criteria and 'search algorithms'. Subspace methods are typically used to generate state estimates $\tilde{x}_{1:T}$ from input/output data that can then be used in quality-of-fit criteria such as equation error, c.f., (2.24). In addition, many of the convex relaxations studied in this thesis rely on such an approximate state sequence for their construction; e.g. approximate states are useful for building multipliers in the Lagrangian relaxations of Section 2.4. For this reason,

we devote some time to the key ideas and geometrical principles underlying the subspace identification methods. A more thorough treatment can be found in [241, §2].

Subspace preliminaries

Idea is to estimate internal states from input-output data, by exploiting the property that linear dynamical systems are invariant under similarity transformations.

Let $\{u_t\}_{t=0}^T$ and $\{y_t\}_{t=0}^T$ denote the noiseless inputs and simulated outputs of the LTI system,

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t, \\ y_t &= Cx_t + Du_t. \end{aligned} \tag{2.28}$$

The key data structures that subspace algorithms manipulate are block Hankel matrices of these inputs and outputs, i.e., for the inputs we have

$$U_{0|2i-1} = \begin{bmatrix} u_0 & u_1 & \dots & u_{j-1} \\ u_1 & u_2 & \dots & u_j \\ \vdots & \vdots & \ddots & \vdots \\ u_{2i-1} & u_{2i} & \dots & u_{2i+j-2} \end{bmatrix} \in \mathbb{R}^{2in_u \times j}.$$

The block Hankel matrix for the outputs, $Y_{0|2i-1}$, is defined similarly. The number of block rows, i, is a user defined parameter, that ought to be set at least as large as the (unknown) order of the true system, n_x. The number of columns, j, is usually selected so as to make use of all T data samples; i.e. there are $2i+j-1$ block elements in the matrix, so $j = T-2i+1$. In what follows, it is convenient to partition the block Hankel matrices as

$$U_{0|2i+1} = \begin{bmatrix} U_{0|i-1} \\ U_{i|2i-1} \end{bmatrix} = \begin{bmatrix} U_p \\ U_f \end{bmatrix}, \quad Y_{0|2i+1} = \begin{bmatrix} Y_{0|i-1} \\ Y_{i|2i-1} \end{bmatrix} = \begin{bmatrix} Y_p \\ Y_f \end{bmatrix},$$

where the subscripts p and f denote 'past' and 'future' measurements, respectively, relative to the $t = i-1^{\text{th}}$ point in the time series data. This notion of 'past' and 'future' quantities also applies to the internal states. In particular, a sequence of j states is denoted by

$$X_i = [x_i, \dots, x_{i+j-1}] \in \mathbb{R}^{n_x \times j}$$

from which we can define the past and future state sequences, $X_p = X_0$, $X_f = X_i$. The relationship between sequences of inputs, states and outputs can be expressed as

$$Y_p = \Gamma_i X_p + H_i U_p \tag{2.29}$$

$$Y_f = \Gamma_i X_f + H_i U_f \tag{2.30}$$

$$X_f = A^i X_p + \Delta_i U_p. \tag{2.31}$$

where Γ_i denotes the extended observability matrix,

$$\Gamma_i = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{i-1} \end{bmatrix}$$

Δ_i denotes the extended controllability matrix,

$$\Delta_i = [A^{i-1}B, \ldots, AB, B],$$

and H_i denotes the block Toeplitz matrix,

$$H_i = \begin{bmatrix} D & 0 & 0 & \ldots & 0 \\ CB & D & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{i-2}B & CA^{i-3}B & CA^{i-4}B & \ldots & D \end{bmatrix}.$$

Finally, for matrices $M \in \mathbb{R}^{m\times j}$ and $N \in \mathbb{R}^{n\times j}$, let $\Pi_N := N'(NN')^\dagger N$ define the orthogonal projection operator onto the row space of N, such that

$$M/N := M\Pi_N = MN'(NN')^\dagger N$$

gives the orthogonal projection of the row space of M onto the row space of N. Similarly,

$$M/N^\perp := M\Pi_{N^\perp} = M(I - N'(NN')^\dagger N)$$

gives the projection of the row space of M onto the orthogonal complement of the row space of N. In addition, instead of decomposing $M = M\Pi_N + M\Pi_{N^\perp}$ into the row space of N and its orthogonal complement, we may wish to project the row space of M onto the row space of N *along* the row space of some other matrix $P \in \mathbb{R}^{p\times j}$. This defines the *oblique projection*

$$M\_PN = \left[M/P^\perp\right]\left[N/P^\perp\right]^\dagger N.$$

See [241, §1.4] for further details.

Geometric interpretation

We now show how the state sequence X_f can be recovered from the input-output data by algebraic operations. The first key insight exploited by subspace methods is that the state

sequence X_f lies in the row space of W_p, where

$$W_p := \begin{bmatrix} U_p \\ Y_p \end{bmatrix}, \quad W_f := \begin{bmatrix} U_f \\ Y_f \end{bmatrix}.$$

To see this, begin by rearranging (2.29) as follows

$$\Gamma_i X_p = Y_p - H_i U_p.$$

The idea is to solve this equation for the past states, X_p. Assuming a minimal realization of the linear system (in particular, that (A, C) is observable), the extended observability matrix Γ_i has rank n_x, and so

$$X_p = \Gamma_i^\dagger (Y_p - H_i U_p).$$

Substituting the above expression into (2.31) yields:

$$\begin{aligned} X_f &= A^i X_p + \Delta_i U_p \\ &= A^i \Gamma_i^\dagger (Y_p - H_i U_p) + \Delta_i U_p \\ &= \underbrace{[\Delta_i - A^i \Gamma_i^\dagger H_i, A^i \Gamma_i^\dagger]}_{L_p} \underbrace{\begin{bmatrix} U_p \\ Y_p \end{bmatrix}}_{W_p}, \end{aligned}$$

i.e., the future states, X_f, are a linear combination of the rows of W_p. Substituting $X_f = L_p W_p$ into (2.30) gives

$$Y_f = \Gamma_i L_p W_p + H_i U_f. \tag{2.32}$$

Now we are in a position to appreciate the geometrical interpretation of the subspace identification algorithm, depicted in Figure 2.1. For convenience, define $\mathcal{O}_i = \Gamma_i X_f$. Figure 2.1 clearly shows the simple fact that $Y_f = \mathcal{O}_i + H_i U_f$, which is nothing more than (2.30). However, notice that $\mathcal{O}_i$ is depicted in the row space of W_p, which incorporates the result from (2.32). Furthermore, observe that $\mathcal{O}_i$ (an unknown quantity) can be recovered by an oblique projection of Y_f along the row space of U_f and onto the row space of W_p,

$$\mathcal{O}_i = Y_f /_{U_f} W_p \tag{2.33}$$

which requires the measured quantities Y_f, U_f and W_p only.

Now, as Γ_i and X_f are both rank n_x, their product $\mathcal{O}_i = \Gamma_i X_f$ is also rank n_x. Therefore, it is possible to factor $\mathcal{O}_i$ by, e.g., using a singular value decomposition:

$$\mathcal{O}_i = [U_1, U_2] \begin{bmatrix} S_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1' \\ V_2' \end{bmatrix}. \tag{2.34}$$

Then, the state sequence X_f can be recovered (up to an unknown similarity transformation)

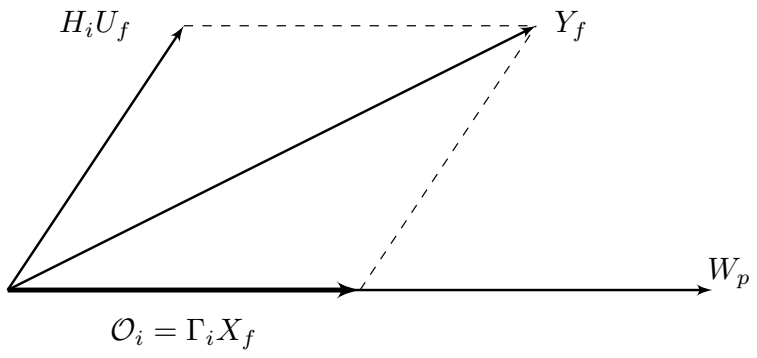


Figure 2.1 – Geometric interpretation of subspace identification algorithms. The depiction $Y_f = \mathcal{O}_i + H_iU_f$ is nothing more than (2.30). However, notice that $\mathcal{O}_i$ is depicted in the row space of W_p, which incorporates the result from (2.32). This figure is based on Figure 2.5 in [241].

by

$$X_f = S_1^{1/2} V_1'. \tag{2.35}$$

The system matrices A, B, C, D may then be recovered, e.g., by minimization of equation error, as in (2.24).

By replacing (2.34) with the singular value decomposition (SVD) of $W_1\mathcal{O}_iW_2$, where W_1 and W_2 are user-specified weighting matrices, one obtains a general framework for subspace identification, in which specific choices of weights correspond to various subspace methods, e.g. N4SID, MOESP, CVA; c.f. [241, §2.3]. For instance, in the more common (and arguably more easily understood) 'projection algorithms', one projects (2.30) onto the orthogonal complement of U_f, i.e.,

$$Y_f/U_f^\perp = (\Gamma_i X_f + H_iU_f)/U_f^\perp \implies Y_f/U_f^\perp = \Gamma_i X_f/U_f^\perp.$$

This has the effect of 'removing' the effect of the inputs U_f, which allows the extended observability matrix to be recovered by SVD of $Y_f/U_f^\perp$. By choosing weights $W_1 = I$ and $W_2 = \Pi_{U_f^\perp}$, the approach outlined above is completely equivalent to the projection algorithm.

Effect of noise

When the true system (2.28) that generates the problem data is purely deterministic, i.e. unaffected by random disturbances, the procedure outlined above returns the true simulated states, up to some coordinate transformation. When the system is affected by Gaussian disturbances, as in the LGSS model of (2.16), the same procedure, i.e. (2.33), (2.34) and (2.35), yields the (non steady-state) Kalman filter state estimates (up to some coordinate transformation) [241, Theorem 12]. If these state sequences are used directly in minimization of equation error, the resulting estimated system parameters will be asymptotically

biased [241, §4.4.2]. An unbiased, albeit more complicated, procedure is given in [241, §4.4.1].

2.1.4 Search algorithms

Once the model parametrization and quality-of-fit metric have been selected, the search method is often implicitly specified, especially when the ensuing optimization is convex. In this section we provide an overview of the algorithms typically used to solve the optimization problem $\min_{\theta\in\Theta} \mathcal{E}(\theta)$.

Linear least squares

One of the most common search algorithms employed in system identification is linear least squares, used to compute the global minimizer of the equation error criterion. For example, recall the equation error, a.k.a *least squares*, criterion from (2.23), i.e.,

$$\mathcal{E}^{\mathrm{ee}}(\theta) = \frac{1}{T}\sum_{t=1}^{T} |\tilde{y}_t - \tilde{\phi}_t'\theta|^2$$

for some autoregressive model (possibly nonlinear) with parameters θ and vector of regressors ϕ_t. Here we set $\mu = 0$ for simplicity and ease of explanation. As equation error is quadratic in θ, the global minimizer is given in closed form by

$$\hat{\theta}^{\mathrm{LS}} := \arg\min_{\theta} \mathcal{E}^{\mathrm{ee}}(\theta) = \left[\sum_{t=1}^{T} \tilde{\phi}_t\tilde{\phi}_t'\right]^{-1} \sum_{t=1}^{T} \tilde{\phi}_t\tilde{y}_t. \tag{2.36}$$

The simplicity of the formula in (2.36) is appealing; however, least squares estimates are not computed this way in practice, for numerical reasons. For instance, the matrix $R_T = \sum_{t=1}^{T} \tilde{\phi}_t\tilde{\phi}_t'$ may be ill-conditioned, especially for high dimensional systems. Fortunately, by now numerically robust methods for linear least squares problems have been developed, such as those involving QR-factorizations, c.f. e.g. [75, §5]. See also [129, §10.1] for a thorough discussion on the numerical solutions of least squares problems in a system identification context.

Nonlinear programming

Popularity of the least squares criterion is largely due to the fact that the global minimizer can be obtained non-iteratively, in closed form, as in (2.36). Unfortunately, most of the quality-of-fit criteria outlined in Section 2.1.2, e.g. prediction error, output error and likelihood, cannot (in general) be optimized analytically.

In general, the problem $\hat{\theta} = \arg\min_\theta \; f(\theta)$, for some quality-of-fit metric $f(\theta)$, must be solved iteratively, i.e., at the kth iteration of such a procedure we update our estimate $\hat{\theta}^k$ of the model parameters according to

$$\hat{\theta}^{k+1} = \hat{\theta}^k + \alpha d^k, \tag{2.37}$$

where d^k is the search direction, and α is the step length, chosen such that $f(\hat{\theta}^{k+1}) < f(\hat{\theta}^k)$; c.f. [19, 136, 262] for a general treatments of iterative numerical optimization. At a high-level, numerical optimization methods can be characterized by the complexity of the information used in the parameter update rule:

1. *first order methods* build a local linear approximation to the function, based on the function value and gradient (first derivative), to compute (2.37), e.g., for a gradient descent method $d^k = \nabla f(\hat{\theta}^k)$,

2. *second order methods* build a local quadratic approximation to the function, based on the function value, gradient and Hessian (second derivative), to compute (2.37), e.g., in a Newton algorithm the 'Newton' direction is given by

$$d^k = -\left[\nabla^2 f(\hat{\theta}^k)\right]^{-1} \nabla f(\hat{\theta}^k). \tag{2.38}$$

Second order methods provide fast convergence when f can be well approximated by a quadratic function, e.g., close to a minimum. In practice though, the full Newton step as in (2.38) is rarely used. Foremost, exact computation of the Hessian is often computationally expensive, and even storage of the Hessian could require excessive memory. So called *quasi-Newton* methods use an approximation of $\nabla^2 f(\theta^k)$ in (2.38), usually constructed using gradient information, to compute the search direction; e.g., BFGS and limited memory variants [123] use a gradient based rank-1 update of the Hessian approximation. Second, even if the exact Hessian is used, the full step length corresponding to $\alpha = 1$ is seldom used; much more common is a so-called damped Newton step in which $\alpha \leq 1$. There are two main families of approaches for computing the step length. Line search methods [262, §3], such as Gauss-Newton for nonlinear least squares problems [41, 252], first compute d^k, and then search in this direction for a suitable α such that $f(\hat{\theta}^k + \alpha d^k) < f(\hat{\theta}^k)$. Trust region methods [262, §4], such as the Levenberg-Marquardt algorithm [156], update θ by minimizing a quadratic approximation of f that is deemed to be sufficiently accurate, or *trusted*, within a defined region about the current iterate, θ^k, thereby choosing the step length and direction simultaneously.

First order methods typically provide slower convergence, especially close to local minima. Nonetheless, such approaches are less computationally expensive, and have witnessed somewhat of a resurgence recently, especially in the machine learning community, where methods such as stochastic gradient descent [26] are used extensively. There are, at least, three main reasons for this: (i) necessity: for large scale models with many parameters

(e.g. deep neural networks) it is often too expensive to compute and store approximations of the Hessian; (ii) parallelization: first order methods are more amenable to parallel implementations [271], which is important when training large models on massive datasets; (iii) it just works: reaching the global minimum is often unnecessary or even undesirable, e.g., early stopping is a common heuristic to avoid overfitting [264]. First order methods are undoubtedly more popular in machine learning than system identification; however, the interesting recent work [85] has demonstrated the efficacy of stochastic gradient descent in output error minimization problems.

There is a third class of methods, called *zero-th order*, which use only the function value $f(\hat{\theta}^k)$ to compute the parameter update, $\hat{\theta}^{k+1}$. Zero-th order methods have the slowest convergence properties, and are typically only used when gradient information is too difficult or expensive to compute. Such approaches include finite difference approximations of derivatives, grid searches, and genetic algorithms [111].

Expectation maximization

Maximum likelihood identification of nonlinear, and/or non-Gaussian dynamical systems is one such scenario in which the computation of gradients is problematic. In this setting, the likelihood must be approximated with Monte Carlo methods using sequential importance sampling (a.k.a. particle filtering) [203]. Unlike linear Gaussian models, in which the likelihood can be computed in closed form with the Kalman filter, the derivatives of these particle approximations are not readily available. One solution is to proceed via zero-th order methods, such as simulated annealing [199] or the Nelder-Mead simplex method [261]. An alternative is to optimize a differentiable lower bound to the likelihood, which is the approach provided by the expectation maximization (EM) algorithm. Refer to [51] for general details, and [71, 203, 214, 215, 237] for examples of the application of EM to systems identification.

In this section we review the basic principles of ML estimation via the EM algorithm; this material can be regarded as a primer for the technical developments of Chapter 4. The approach is predicated on the assumption that there exists a set of latent (read: 'hidden' or 'unobserved') variables, Z, such that the 'complete' or joint log likelihood function

$$L_\theta(y_{1:T}, Z) = \log p_\theta(y_{1:T}, Z)$$

is easier to optimize than the incomplete log likelihood $L(y_{1:T}) = \log p_\theta(y_{1:T})$. These latent variables may be thought of as the data that we 'wish' we could observe, in the sense that the problem would be more straightforward if Z was available.

The maximum likelihood problem can be related to the joint likelihood by marginalizing over the latent variables

$$\hat{\theta}^{\mathrm{ML}} := \arg\max_\theta L_\theta(y_{1:T}) = \arg\max_\theta \log \int p_\theta(y_{1:T}, Z) dZ.$$

This is a formidable optimization problem, as marginalization has separated the logarithm from the likelihood. The idea behind the EM algorithm is to take some estimate θ_k of the parameters, use this to build a lower bound for $L_\theta(y_{1:T})$, then maximize the lower bound in place of the likelihood function to improve our estimate of θ.

For an arbitrary distribution $\rho(Z)$, Jensen's inequality gives

$$\int \rho(Z) \log \frac{p_\theta(y_{1:T}, Z)}{\rho(Z)} dZ \leq \log \int \rho(Z) \frac{p_\theta(y_{1:T}, Z)}{\rho(Z)} dZ,$$

where the right hand side is simply $L_\theta(y_{1:T})$. Therefore, we may define a lower bound for the likelihood function by

$$B_\rho(\theta, \theta_k) := \int \rho(Z) \log \frac{p_\theta(y_{1:T}, Z)}{\rho(Z)} dZ. \tag{2.39}$$

Notice that Jensen's inequality has 'reunited' the logarithm with the likelihood, thereby making the bound more amenable to optimization. Choosing $\rho(Z) = p_{\theta_k}(Z|y_{1:T})$ yields an 'optimal', or 'tight', bound in the sense that $B_\rho(\theta_k, \theta_k) = L_{\theta_k}(y_{1:T})$ and so intuitively, maximizing $B_\rho(\theta, \theta_k)$ w.r.t θ will result in $L_\theta(y_{1:T}) \geq L_{\theta_k}(y_{1:T})$.

It is convenient to express the optimal bound in the form

$$B_\rho(\theta, \theta_k) = Q(\theta, \theta_k) + H(\theta_k),$$

where $Q(\theta, \theta_k)$ represents

$$\int p_{\theta_k}(Z|y_{1:T}) \log p_\theta(y_{1:T}, Z) \; dZ = \mathrm{E}_{\theta_k}\big[\log p_\theta(y_{1:T}, Z)|y_{1:T}\big]$$

and $H(\theta_k)$ denotes the differential entropy of $p_{\theta_k}(Z|y_{1:T})$. As $H(\theta_k)$ is independent of θ, maximizing the bound reduces to maximizing $Q(\theta, \theta_k)$.

To summarize, each iteration of the EM algorithm consists of an expectation (E) step to compute $Q(\theta, \theta_k)$, and a maximization (M) step in which $Q(\theta, \theta_k)$ is maximized to deliver an improved θ_{k+1}, such that $L_{\theta_{k+1}}(y_{1:T}) \geq L_{\theta_k}(y_{1:T})$.

Algorithm 1 Expectation Maximization algorithm

1. Set $k = 0$ and initialize θ_k such that $L_{\theta_k}(y_{1:T})$ is finite.
2. **Expectation (E) Step:**

$$Q(\theta, \theta_k) = \mathrm{E}_{\theta_k}\left[\log p_\theta(y_{1:T}, Z)|y_{1:T}\right] \tag{2.40}$$

3. **Maximization (M) Step:**

$$\theta_{k+1} = \arg\max_\theta Q(\theta, \theta_k) \tag{2.41}$$

4. If not converged, $k \leftarrow k + 1$ and return to step 2.

2.2 Convex optimization

In Section 2.1 we described system identification as an optimization problem: set, objective, algorithm. One of the most important factors that determines the tractability of any mathematical optimization problem is *convexity*, of both the cost function and feasible set. One of the main reasons for this is the property that every locally optimal solution of a convex optimization problem is also a globally optimal solution; i.e., there are no 'sub-optimal' local minima in which an iterative minimization procedure might become 'stuck'. In [196], Rockafellar describes convexity as "the great watershed in optimization", in the context of complexity, tractability and completeness of theory. More formally, [161] established that the information-based complexity of convex optimization problems is considerably lower than general optimization problems. Further work on the computational complexity of convex optimization can be found, e.g., in [162, 245]. Even for problems that aren't convex, e.g. minimization of most quality-of-fit metrics in system identification other than equation error, convexity can play an important role, for instance, in the solution intermediate problems as part of an iterative numerical method [196].

For these reasons, convex optimization is a motif that underpins most of the technical developments and contributions of this thesis; e.g. the development of convex parametrizations of stable models, convex approximations of nonconvex quality of fit metrics, and specialized algorithms that exploit the structure of these convex functions to expedite the search for model parameters.

In this section, we provide a very brief introduction to convex optimization; specifically, we review some of the ideas and methods that are pertinent to the technical developments of this thesis. For a thorough, and highly accessible, treatment of convex programming, c.f., e.g., [28].

2.2.1 Convex programs

A general optimization problem has the form

$$\min_{\theta} \quad f_0(\theta) \tag{2.42a}$$

$$\text{s.t.} \quad f_i(\theta) \geq 0, \quad i = 1, \ldots, m \tag{2.42b}$$

where $\theta \in \mathbb{R}^{n_\theta}$ is the vector of *decision variables*, $f_0 : \mathbb{R}^{n_\theta} \mapsto \mathbb{R}$ is the *cost* or *objective* function to be minimized, and the functions $f_i : \mathbb{R}^{n_\theta} \mapsto \mathbb{R}$ define the *constraints* in (2.42b) that the decision variables must satisfy. A vector θ^* is a solution of (2.42) if it has the lowest cost (i.e. smallest value of the objective function) out of all the vectors that satisfy the constraints. Such a vector is said to be *optimal*.

An optimization problem is convex if the objective and constraint functions are convex, i.e., they satisfy

$$f_i(\alpha x + (1-\alpha)y) \leq \alpha f_i(x) + (1-\alpha) f_i(y) \tag{2.43}$$

for all $x, y \in \mathbb{R}^{n_\theta}$ and $\alpha \in [0, 1]$. As discussed, it has long been appreciated that convex optimization problems are fundamentally more tractable than general optimization problems. In practice, there are certain classes of convex optimization problems, characterized by the properties of the functions f_i, for which solution methods are readily available. In this section, we describe some such programs that appear regularly in the sequel.

Linear programming

One of the best known, and most widely used, convex optimization problems is the *linear program* (LP), characterized by objective and constraint functions that are linear, i.e.,

$$f_i(\alpha x + \beta y) = \alpha f_i(x) + \beta f_i(y) \tag{2.44}$$

for all $x, y \in \mathbb{R}^{n_\theta}$ and $\alpha, \beta \in \mathbb{R}$. Comparing the linearity condition (2.44) to the convexity condition in (2.43), it is apparent that linear programs are a special case of the general convex optimization problem in (2.42). Equivalently, convex optimization can be viewed an a generalization of linear programming; indeed, many important solution methods for general convex programs (e.g., interior-point methods for semidefinite programs) developed as extensions of algorithms for linear programming; c.f. [105, 162], and Section 2.2.2.

Linear programs are often expressed in the form,

$$\min_{\theta} \quad c'\theta \tag{2.45a}$$

$$\text{s.t.} \quad a_i'\theta + b_i \geq 0, \quad i = 1, \ldots, m \tag{2.45b}$$

where $c, a_i \in \mathbb{R}^{n_\theta}$ and $b_i \in \mathbb{R}$ are constant quantities. The linear constraints may be

expressed more compactly by

$$A\theta \geq b, \quad A = \begin{bmatrix} a_1' \\ \vdots \\ a_m' \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_m \end{bmatrix}. \tag{2.46}$$

Linear programming is used extensively in a wide variety of applications, including (but not limited to), economics, manufacturing [36], electrical power and energy [270], telecommunications, as well as a number of 'logistical settings' such as transport, routing and scheduling [47]. Historically, linear programs have played a less prominent role in system identification, and control-theoretic applications more generally, due largely to the prevalence of *quadratic* quantities in these fields; for example: quadratic quality-of-fit criteria (e.g. least squares criterion), quadratic Lyapunov functions for stability analysis, and quadratic cost-to-go functions for optimal control and filtering (e.g., linear quadratic regulators and the Kalman filter). Having said this, with the recent resurgence of interest in *positive dynamical systems*, [16, 83], linear programming has found new application in scalable analysis and control design; c.f., [191, 192], and Chapter 5.

Semidefinite programming

In semidefinite programming one minimizes a linear cost function subject to the constraint that an affine combination of matrices is positive semidefinite, i.e.,

$$\min_{\theta} \quad c'\theta \tag{2.47a}$$

$$\text{s.t.} \quad F(\theta) := F_0 + \sum_{i=1}^{n_\theta} \theta(i) F_i \succeq 0, \tag{2.47b}$$

where $c \in \mathbb{R}^{n_\theta}$ and $F_i \in \mathbb{S}^n$. Problem (2.47) is called a semidefinite program (SDP), although we also use SDP as shorthand for 'semidefinite programming'. The constraint $F(\theta) \succeq 0$ is called a *linear matrix inequality* (LMI), and is convex in θ, i.e., if $F(x) \succeq 0$ and $F(y) \succeq 0$ then for $\alpha \in [0, 1]$,

$$F(\alpha x + (1-\alpha)y) = \alpha F(x) + (1-\alpha)F(y) \succeq 0.$$

A linear matrix inequality can be viewed as an extension of the inequalities in LP, i.e. (2.45), where the nonnegative orthant $\mathbb{R}_+$ is replaced by the cone of positive semidefinite matrices $\mathbb{S}_+$. In fact, semidefinite programming includes many standard convex programs as special cases; e.g. the LP in (2.45) can be solved as an SDP by setting

$$F_0 = \operatorname{diag}(b), \quad F_i = \operatorname{diag}(A(:, i)), \quad i = 1, \ldots, m,$$

where A and b are defined in (2.46). In this sense, semidefinite programming unifies several standard convex problems (e.g. linear and quadratic programming).

The combination of increased generality over LP and efficient solution methods, has seen SDP used in a wide variety of applications, c.f., [242, §2] and [29] for some examples. In the context of system and control theory, SDP and LMIs have a long and celebrated history, dating back to the stability analysis of Lyapunov in the late 19th century. For instance, consider the Lyapunov stability condition

$$A'P + PA \prec 0 \tag{2.48}$$

for a continuous-time linear dynamical system, $\dot{x} = Ax$. The condition (2.48) may be interpreted as negative definiteness of the time-derivative of the Lyapunov function $V(x) = x'Px$, and is an LMI with decision variable $\theta = P \in \mathbb{S}_{++}$. In fact, Lyapunov showed that the LMI (2.48) can be solved analytically, by first choosing any $Q \in \mathbb{S}_{++}$ and then solving the linear Lyapunov equation $A'P + AP = -Q$.

Skipping ahead 70 years, the work of Kalman, Yakubovich and Popov linked the solution of certain LMIs to Popov's frequency-domain condition for the absolute stability problem [183]. This lead to celebrated Kalman-Yakubovich-Popov (KYP) lemma, also known as the *positive-real* (PR) lemma, along with many variants, such as the circle criterion, c.f., e.g., [189],[100, §6,7]. LMIs also played a central role in the dissipativity approach to system analysis and quadratic optimal control introduced by Willems in the early 1970s, [255, 257]. It is worth noting that during this period, LMIs were solved by graphical methods that exploited equivalence to certain frequency domain conditions.

A major breakthrough in the solution of SDP occurred in the late 1980s when Nesterov and Nemirovski showed that interior point methods developed for linear programming can be generalized to all convex optimization problems [162]. The theoretical complexity guarantees for interior point methods hinge on the existence of *barrier functions* with appropriate properties, namely *self-concordance*; c.f. [28, §9.6] and Section 2.2.2 for a brief discussion. From a practical viewpoint, an important contribution from Nesterov and Nemirovski was the discovery of such a barrier function for SDP, that (along with its derivatives) is readily computable. Similar work extending interior point methods to SDP was done independently by [5] and [64].

Sum-of-squares programming

Sum-of-squares (SOS) programming is an extension, or even application, of SDP. Checking global nonnegativity of a multivariate function is fundamental problem in many areas of applied mathematics, e.g., stability analysis of nonlinear dynamical systems. The feasibility of such problems depends largely on the class of functions involved. In applications, multivariate polynomials are a popular choice, as they offer a good compromise between limited

complexity (necessary for computationally tractability) and generality (i.e. the ability to approximate a wide variety of continuous functions, e.g., by a Taylor series).

Definition 2.1. A polynomial p in $x \in \mathbb{R}^n$ is a finite linear combination of monomials

$$p(x) = \sum_{\alpha} c_\alpha x^\alpha = \sum_{\alpha} c_\alpha x_1^{\alpha(1)} \dots x_n^{\alpha(n)} \tag{2.49}$$

where $\alpha \in \mathbb{N}$. The set of all polynomials in $x \in \mathbb{R}^n$ is denoted $\mathbb{R}^n[x]$.

The sum $\alpha(1) + \cdots + \alpha(n)$ is referred to as the *total degree* of the monomial x^α. The degree of the polynomial is given by the highest degree of its constituent monomials.

Checking non-negativity of a general multivariate polynomial is known to be NP-hard. However, a simple sufficient condition for global nonnegativity of a polynomial p is the existence of a sum-of-squares (SOS) decomposition:

$$p(x) = \sum_i p_i(x)^2, \quad p(x) \in \mathbb{R}^n[x]. \tag{2.50}$$

Clearly, such a polynomial must have even degree, i.e. $2d$ for some integer d. This sufficient condition for nonnegativity is the basic idea which underpins SOS programming: given a polynomial of degree $2d$, the goal is to find a representation

$$p(x) = z'Qz$$

where Q is a positive semidefinite constant matrix, called the *Gram matrix*, and z is the vector of all possible monomials in x of degree less than or equal to d,

$$z = [1, x_1, x_2, \dots, x_n, x_1x_2, \dots, x_n^d]' \in \mathbb{R}^{\binom{n+d}{n}}. \tag{2.51}$$

Positive semidefiniteness of the Gram matrix, i.e. $Q \succeq 0$, implies $z'Qz \geq 0$ for all z (and, therefore, for all x too), which is clearly sufficient for nonnegativity of p. To link this construction directly to the decomposition in (2.50), as $Q \succeq 0$ it admits a factorization $Q = M'M$. Writing

$$Mz = \begin{bmatrix} \vdots \\ \sum_j M_{ij} z_j \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ f_i(x) \\ \vdots \end{bmatrix}$$

gives

$$p(x) = z'Qz = (Mz)'(Mz) = \sum_i f_i(x)^2.$$

To summarize, the search for a SOS decomposition requires finding a Gram matrix that lies in the intersection of: (i) the positive semidefinite cone, i.e. $Q \geq 0$, and (ii) an affine subspace, i.e., the linear equality constraints that equate the coefficients of p to those of $z'Qz$. Such a search can be formulated as a semidefinite program.

This Gram matrix representation of SOS polynomials was studied and analyzed in [44], and numerical implementations of the search for Gram matrices were presented in [184], although no consideration was given to the convexity of such a search. In a convex optimization framework, the SOS decomposition plays a pivotal role in the global bounds on polynomial functions presented in [206]. The work of Parrilo [171, 172], see also [35, 115], emphasized the connection between convex semidefinite programming and SOS decomposition, which was followed by widespread interest and adoption in the control community.

Before concluding our discussion of SOS programming, we wish to emphasize two points. First, the SOS decomposition is clearly sufficient for nonnegativity, but it is not necessary; i.e., not all nonnegative polynomials admit a SOS decomposition. The solution to Hilbert's 17th problem, posed at the beginning of the 20th century, establishes the conditions under which nonnegativity implies the existence of a SOS decomposition:

Theorem 2.1 (Hilbert)**.** Let m denote the degree of a polynomial $p \in \mathbb{R}^n[x]$. Nonnegativity and the existence of a sum-of-squares decomposition are equivalent when p is:

i. univariate: $n = 1$,

ii. quadratic: $m = 2$,

iii. bivariate and quartic: $n = 2$, $m = 4$.

Second, it is not always necessary to use all possible monomials of degree $\leq d$ in the monomial basis z, c.f., (2.51). Careful selection of the basis monomials can simplify the resulting SOS program, e.g., reduce the number of constraints and decision variables, with no increase in conservatism. Tools such as the Newton polytope [134, 222],[171, §4.2], and facial reduction [175], can be used to generate an effective basis.

Example 2.1 (Sum-of-squares decomposition)**.** The following example of a SOS decomposition is based on [168]. Consider the bivariate polynomial

$$p(x) = x_1^2 + 2x_1^4 + 2x_1^3x_2 - x_1^2x_2^2 + 5x_2^4. \tag{2.52}$$

The degree of the polynomial is 4, i.e., $d = 2$. To search for a SOS decomposition, we first collate all of the monomials in $\mathbb{R}^2$ up to degree $d = 2$:

$$z = [1, x_1, x_2, x_1x_2, x_1^2, x_2^2]'.$$

After removing the monomials that lie outside the Newton polytope, it is apparent that we only require

$$z = [x_1, x_1x_2, x_1^2, x_2^2]'. \tag{2.53}$$

With the set of candidate monomials in (2.53) we define the generic symmetric Gram matrix

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ q_{14} & q_{24} & q_{34} & q_{44} \end{bmatrix}. \tag{2.54}$$

The search for a SOS decomposition of (2.52) can then be formulated as the following semidefinite program:

$$\begin{aligned} \text{find} \quad & Q && (2.55a) \\ \text{s.t.} \quad & Q \succeq 0 && (2.55b) \\ & q_{11} = 1,\ q_{22} = 2,\ 2q_{23} = 2,\ q_{33} + 2q_{24} = -1,\ q_{44} = 5 && (2.55c) \\ & q_{ij} = 0 \text{ otherwise .} && (2.55d) \end{aligned}$$

Here, the LMI (2.55b) enforces positive semidefinitness of the Gram matrix, and the linear equality constraints ensure that the coefficients of $z'Qz$ match those of $p(x)$, e.g., $q_{33}+2q_{24} = -1$ ensures that the monomial $(q_{33} + 2q_{24})x_1^2x_2^2$ in $z'Qz$ matches the monomial $-x_1^2x_2^2$ in $p(x)$.

2.2.2 Interior point methods

In this section we discuss a family of solution methods for convex optimization problems called *interior point methods* (IPMs). The widespread availability of high-quality general-purpose solvers, and parsers, for convex optimization problems means that an understanding of the underlying solution machinery is often unnecessary for the practitioner. Nonetheless, we provide a brief introduction to the ideas behind these methods to help facilitate the exposition of the specialized interior point solvers derived in Chapter 3. For a thorough, yet concise, introduction to IPMs, refer to [28, §11], or [242] in the context of semidefinite programs. For an in-depth treatment, the reader should consult the seminal work of Nesterov and Nemirovski [162].

Barrier methods

At a high level, numerical optimization schemes solve difficult optimization problems via a sequence of simpler problems. The simplest (non-trivial) optimization problem is minimization of a convex quadratic function, for which the solution can be written in closed form (numerical conditioning and scalability issues aside). Further up the complexity hierarchy, we have smooth and unconstrained optimization problems, to which Newton's method is

applicable. The idea behind Newton's method is to solve such problems via a sequence of quadratic approximations; c.f.Section 2.1.4 and the references within. Roughly speaking, we expect Newton's method to work well when the quadratic functions constructed from local information (i.e., the value, gradient and Hessian of the cost function) provide a good approximation of the shape of the cost function globally, or at least, a good approximation close to the minimum.

Further up the complexity hierarchy we find constrained optimization problems, possibly with non-smooth constraints (e.g. feasible sets with 'sharp corners'). Newton's method is no loner applicable, as the cost function and its derivatives do not provide any information about the affect of the constraints, i.e., the shape of the feasible set. The idea behind the barrier method is to augment the cost function with additional penalty terms that give a smooth approximation of the feasible set. The augmented cost function can then be minimized as an unconstrained optimization problem, using, e.g., Newton's method.

To make these ideas more precise, recall the general constrained convex optimization problem of the form (2.42), repeated here for convenience:

$$\min_{\theta} \; f_0(\theta) \quad \text{s.t.} \; f_i(\theta) \leq 0, \; i = 1, \ldots, m.$$

Direct application of Newton's method is not possible. To see this clearly, consider the equivalent formulation

$$f_0^* = \min_{\theta} \; f_0(\theta) + \sum_{i=1}^{m} \mathcal{I}_-(f_i(\theta)) \tag{2.56}$$

where $\mathcal{I} : \mathbb{R} \mapsto \mathbb{R}$ is the *indicator function* for nonpositive real numbers:

$$\mathcal{I}_-(c) = \begin{cases} 0 & x \leq 0 \\ \infty & x > 0 \end{cases}. \tag{2.57}$$

The indicator function is non-smooth; see Figure 2.2 for an illustration. Therefore, the cost function in (2.56) is not (in general) differentiable, which precludes application of Newton's method. The idea behind an interior point method is to replace each non-smooth indicator function in the objective of (2.56) with a smooth approximation, $\phi(\dot{})$, called a *barrier function*. We discuss barrier functions in the sequel, but for now, we emphasize that $\phi(x)$ should: (i) be smooth, (ii) be finite inside the feasible set, e.g. for $x \leq 0$ when approximating $\mathcal{I}_-$, (iii) tend towards infinity at the boundary of the feasible set, e.g. $x = 0$. The barrier function approximation of (2.56) is given by

$$\theta^*(\tau) = \arg\min_{\theta} \; \left\{ f_\tau(\theta) := f_0(\theta) + \frac{1}{\tau} \sum_{i=1}^{m} \phi(f_i(\theta)) \right\} \tag{2.58}$$

where $\tau \geq 0$ is a parameter that control the accuracy with which (2.58) approximates (2.56). Specifically, when τ is large the affect of the barrier function is only 'felt' at the boundary

of the feasible set, and (2.58) approximates (2.56) closely; the converse is true for small τ; c.f. Figure 2.2 for an illustration. In fact, it can be shown that $\theta^*(\tau)$ is no more than m/τ suboptimal, meaning

$$f_0(\theta^*(\tau)) - f_0^* \leq \frac{m}{\tau},$$

c.f., [28, §11.2.2]. Therefore, we obtain a more accurate solution for large τ, as expected. Notice that (2.58) is a smooth and unconstrained optimization problem, which can be solved, e.g., by Newton's method. However, we cannot (in general) simply solve (2.58) for large τ, as the Hessian of f_τ will be too large close to the boundary of the feasible set, rendering Newton's method unreliable; again, refer to Figure 2.2 for an illustration.

To circumvent this tradeoff (between an accurate solution and a well-conditioned Hessian), we can solve a *sequence* of optimization problems of the form (2.58) for increasing τ. This leads to the so-called *barrier* or *path-following* interior point method:

Algorithm 2 Basic barrier method.

Given: strictly feasible θ, $\tau > 0$, $\mu > 1$ and tolerance δ

1. Solve (2.58), e.g. by Newton's method, initialized at θ.
2. Update $\theta \leftarrow \theta^*(\tau)$.
3. Terminate if $m/\tau < \delta$.
4. Increase $\tau \leftarrow \mu\tau$ and return to step 1.

Before proceeding, we ought to make a few points regarding the basic barrier method:

i. The solution $\theta^*(\tau)$ of (2.58) is often called the *center point*, and Step 1 the *centering step*. The sequence of points $\theta^*(\tau)$ for increasing τ is called the central path, which leads to the alternate name 'path-following method'.

ii. Concerning the centering step, it is not in fact necessary to solve (2.58) exactly. In [162] it is proven that a single Newton step per τ update will still lead to a sequence of points $\theta^*(\tau)$ that converge to the global optimum.

iii. Path-following methods are not particularly sensitive to the choice of barrier parameter μ; values between 10 and 100 seem to work well in practice; c.f. [28, §11.3.1]. Furthermore, the total number of Newton iterations required for convergence (and thus the total run time of the algorithm) remains approximately constant for a large range of τ. Smaller values of τ lead to smaller changes in the shape of f_τ between τ updates; this means that fewer Newton iterations are required in the centering step, but more τ updates are required for $\theta^*(\tau)$ to converge. The converse is true for larger values of τ.

Example 2.2 (A simple barrier function illustration)**.** Consider the simple linearly con-

strained convex quadratic optimization problem

$$\theta^* = \arg\min_{\theta} \; f_0 := \theta^2 \quad \text{s.t.} \; f_1(\theta) := 1 - \theta \leq 0. \tag{2.59}$$

The solution is clearly given by $\theta^* = 1$. However, the indicator function $\mathcal{I}_-(1-\theta)$, which encodes the constraint $1 - \theta \leq 0$, is non-smooth thereby precluding direct application of Newton's method. We employ the barrier function $\phi(\theta) = -\log(1-\theta)$. Refer to Figure 2.2 for an illustration of the weighted barrier function $\phi(\theta)/\tau$, and observe how more accurate approximations of the indicator function are attained for larger values of the barrier parameter, τ.

The ideas behind the barrier method were first proposed by Fiacco and McCormick in the 1960s [60] in an approach referred to as the *sequential unconstrained minimization technique*. Interest in interior-point algorithms was rekindled in the 1980s, after the work of [72] pointed out the connections to Karmarkar's popular polynomial-time projection algorithm for linear programming [105]. As discussed earlier, a major breakthrough in the development of interior-point methods followed soon after with the work of Nesterov and Nemirovski [162], which extended the complexity theory from linear programming to general nonlinear convex programs.

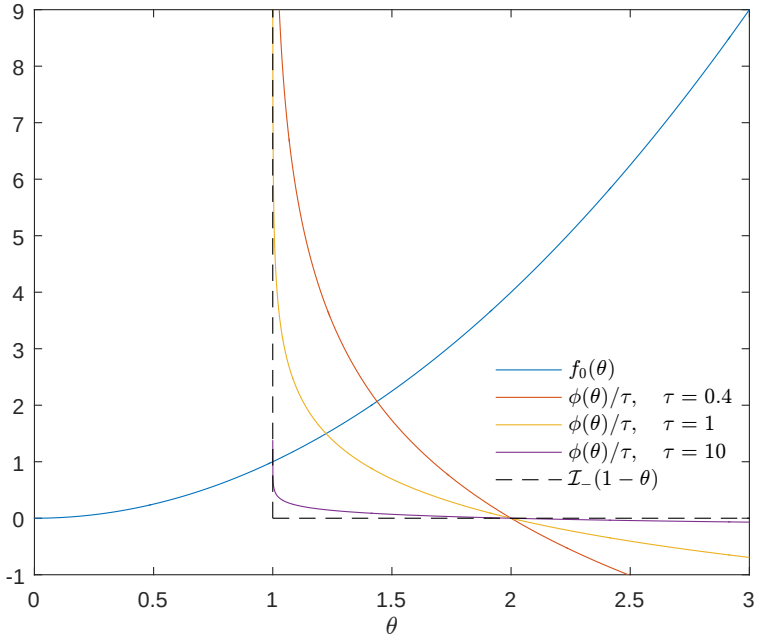
Barrier functions

More specifically, Nesterov and Nemirovski provided precise polynomial-time convergence results for general convex optimization problems. By 'precise', we mean worst-case bounds on the number of iterations for convergence that do not depend on unknown constants. Their analysis hinged on a property of the barrier functions called *self-concordance*. Roughly speaking, a barrier function is self concordant if the rate of change of its Hessian (i.e. the third derivative of the function) is not too big. To gain insight into the importance of self-concordance, recall that we expect Newton's method to work well when the Hessian of the function being optimized is not changing too quickly, as this means that a quadratic approximation based on local information is likely to effectively capture the shape of the cost function in the surrounding region of the domain. For a trivial illustrative example, consider a convex quadratic cost function. In this case, the Hessian is constant and Newton's method finds the global minimum in a single step.

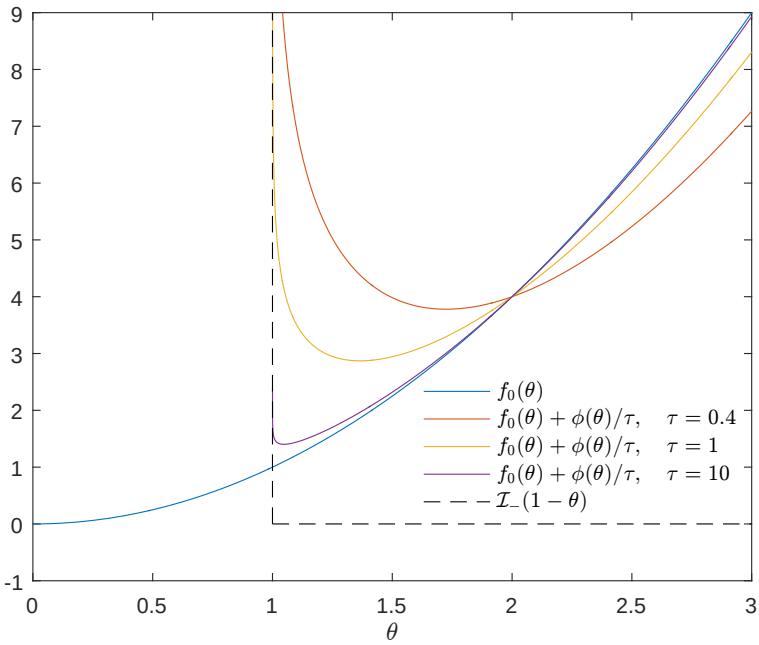
More precisely, self-concordance is defined as follows:

Definition 2.2 (Self-concordance)**.** A convex function $\phi : \mathbb{R} \mapsto \mathbb{R}$ is *self-concordant* if

$$|\nabla^3 \phi(x)| \leq k \left(\nabla^2 \phi(x)\right)^{3/2} \tag{2.60}$$



(a) Weighted barrier functions $\phi(\theta)/\tau = -\log(1-\theta)/\tau$ for the problem (2.59), for various values of τ.



(b) $f_\tau(\theta) = \theta^2 - \log(1-\theta)/\tau$ for problem (2.59), for various values of τ.

Figure 2.2 – (a) Observe how the weighted barrier function approximates the true constraint, given by the indicator function $\mathcal{I}_-(1-\theta)$, more accurately for larger values of τ. (b) Similarly, observe how $\theta^*(\tau) = \arg\min_\theta f_\tau(\theta)$ approximates the true global optimum $\theta^* = 1$ more accurately for larger values of τ.

for some positive constant $k > 0$. The constant k can be chosen arbitrarily, by linearly scaling ϕ. Usually $k = 2$ to simplify related formulae.

A multivariate function $\phi : \mathbb{R}^n \mapsto \mathbb{R}$ is said to be self-concordant if $\tilde{\phi}(\tau) := \phi(x + \tau v)$ is self-concordant for all $x \in \mathbf{dom}\ \phi$ and all v.

Nesterov and Nemirovski proved that a self-concordant barrier function always exists for general convex constraints, c.f., [162, §2.5]. In practical applications, however, the ability to efficiently compute the barrier function (along with its derivatives) is what really matters. The purpose of this section is not to discuss complexity analysis of interior-point methods; c.f., e.g., [162] or [28, §9.6] for details. Rather, we wish to emphasize that for many convex programs encountered in applications, and certainly all the convex programs considered in this thesis, efficiently computable self-concordant barrier functions do exist.

For instance, for linear inequality constraints of the form $a'\theta \leq b$, a self-concordant barrier function is given by

$$\phi(\theta) = -\log(b - a'\theta).$$

For the linear matrix inequalities appearing in semidefinite programming, a self-concordant barrier function for constraints of the form $Z \succeq 0$ with $Z \in \mathbb{S}^n$ is given by

$$\phi(Z) = -\log\det(Z). \tag{2.61}$$

Primal-dual methods

Many state-of-the-art interior-point solvers employ a variant of the barrier method called a *primal-dual method*. Primal-dual methods tend to achieve faster convergence compared to barrier methods, which is particularly advantageous when high accuracy is required. In Chapter 3 we benchmark our custom interior-point algorithms (based on barrier methods) against state-of-the-art primal-dual solvers. To provide context for this comparison, in what follows we briefly enumerate some of the key differences between the barrier and primal-dual methods. For a more comprehensive treatment, c.f., e.g., [28, §11.7] and [162, §4.5].

One of the key differences between primal-dual and barrier methods is the search direction used. In the preceding discussion of the barrier method, it was suggested that (2.58) be solved by Newton's method, for increasing values of τ. As discussed, Newton's method minimizes a cost function by solving sequence of quadratic approximations. More precisely, the 'Newton step' in (2.38) is given by the global minimum of a quadratic approximation to the cost function,

$$f_0(\theta^k + \Delta\theta) \approx f_0(\theta^k) + \nabla f_0(\theta^k)\Delta\theta + \Delta\theta \left[\nabla^2 f_0(\theta^k)\right] \Delta\theta.$$

Alternatively, the same Newton step can be derived as the approximate solution to equations that specify the conditions of optimality for solutions to the constrained optimization

problem (2.42). Specifically, when strong duality holds[2], an optimal solution θ^* to (2.42) must satisfy

$$f_i(\theta^*) \leq 0, \quad i = 1, \ldots, m \tag{2.62a}$$

$$\nu_i^* \geq 0, \quad i = 1, \ldots, m \tag{2.62b}$$

$$\nu^* f_i(\theta^*) = 0, \quad i = 1, \ldots, m \tag{2.62c}$$

$$\nabla f_0(\theta^*) + \sum_{i=1}^{m} \nu_i^* f_i(\theta^*) = 0. \tag{2.62d}$$

These equations are called the Karush-Kuhn-Tucker (KKT) conditions; c.f. [28, §5.5]. Here, ν_i^* are (the optimal values of) Lagrange multipliers used in the construction of a lower bound to the optimal value of (2.42). Condition (2.62c) is referred to as *complementary slackness*, and, roughly speaking, condition (2.62d) can be thought of as a generalization of the first order optimality conditions for unconstrained problems (i.e. gradient equal to zero). When (2.42) is a convex program, the KKT conditions are necessary and sufficient for optimality.

In general, the equations (2.62) are nonlinear in θ^*, and cannot be solved analytically. However, substituting (2.62c) into (2.62d) (thereby eliminating ν^*), and solving a linearization of the resulting condition exactly recovers the Newton step given in (2.38). Moreover, for minimization of $f_\tau(\theta)$ as in (2.58), the same procedure applied to (slightly) modified KKT conditions leads to the search direction (i.e., Newton step) used in the barrier method discussed earlier. In the primal-dual method, one does not eliminate the Lagrangian multiplier; rather, both (2.62c) and (2.62d) are linearized. One then solves for a *primal-dual search direction* comprised of both $\Delta\theta$ (the primal search direction) and $\Delta\nu$ (the dual search direction). The primal and dual search directions are coupled, leading to different search directions than in the primal-only barrier method. It should be noted that for semidefinite programming, linearization of the KKT conditions can be carried out in a number of different ways, leading to different search directions and algorithms, e.g., [88, 110, 163, 234].

Another key difference between primal-dual and barrier methods is the update of the barrier weight parameter, τ. In a primal-dual method, there are no 'outer-iterations' (in which $\tau \leftarrow \mu\tau$) and 'inner-iterations' (i.e. Newton steps). Rather, in each iteration the barrier parameter is updated and a single search direction is computed (which is then used to update both the primal (θ) and dual (ν) variables, via a line search). Furthermore, unlike the barrier method, the values of the primal and dual variables are not guaranteed to be feasible at each iteration.

[2]Strong duality occurs when the maximum value of the Lagrangian dual problem is equivalent to the optimal value of the primal problem, (2.42), i.e., the best Lagrangian lower bound to the original problem is tight. For convex programs, existence of a strictly feasible point (i.e. Slater's condition) is sufficient for strong duality, c.f., [28, §5].

2.2.3 Extensions and alternatives to interior point methods

Second order interior point methods have many desirable properties, such as reliable convergence (polynomial worst-case complexity) to very accurate solutions, with no need for fine tuning of algorithm parameters. However, such methods are not practical for high-dimensional problems, especially semidefinite programs (2.47) with large n, due to the computation time required to solve for the search direction, and memory requirements (namely, storing the Hessian). Solution methods for high-dimensional problems[3] is an active area of research in convex optimization, particularly with the recent surge in popularity of machine learning. We conclude this section on convex optimization with a brief discussion of extensions to the interior point methods (IPMs) presented above, as well as some alternatives to IPMs that may be more suitable for large scale problems. Our treatment of this material is far from exhaustive. The main objective is to contextualize the custom IPM presented in Chapter 3, and provide an introduction to the alternating direction method of multipliers (ADMM) used in Chapter 5.

Exploiting structure in semidefinite programs

In many large scale applications of SDP, e.g. network optimization, the matrices F_i in the LMI of (2.47) are *sparse*. This sparsity can be exploited to reduce the computational complexity of the IPMs described in Section 2.2.2.

The most expensive operation in each iteration of an IPM is usually computation of the search direction, e.g., solving a linearized version of the KKT conditions (2.62), in the case of a primal-dual IPM. One approach is to exploit sparsity in the program structure to solve for the search direction more efficiently, either directly, via a sparse Cholesky decomposition [137], or indirectly, via an iterative solution method, e.g., conjugate gradients [63], as used in [108], or the LSQR algorithm [170] for sparse least squares; c.f. also [242, §7.6]. Further reductions in complexity are possible when the sparsity patterns (i.e. positions of nonzero entries) of the matrices F_i are described by a chordal graph. A graph is said to be chordal (a.k.a. triangulated or decomposable) if every cycle of length greater than three has a chord. For the purpose of complexity reduction in IPMs, matrices with 'chordal sparsity' have the useful property of a 'zero fill-in' Cholesky factorization: i.e., if $X \in \mathbb{S}^n_{++}$ then there exists a permutation matrix P and lower triangular matrix L such that

$$P'XP = LL', \tag{2.63}$$

where $L + L'$ has the same sparsity pattern as X. This factorization simplifies the computation of the usual log determinant barrier functions; e.g., (2.61) can be computed as

$$\phi X := -\log\det(X) = -2\sum_{i=1}^{n} \log L(i,i).$$

[3] Not necessarily semidefinite programs.

Similar simplifications exist for the gradient and Hessian of ϕX, and have been utilized by the solvers presented in [219] and [6].

Another approach to complexity reduction is to decompose large SDPs into many smaller SDPs. For example, when $F(\theta)$ in (2.47) is block-diagonal, i.e., $F(\theta) = \text{blkdiag}(\bar{F}_1, \ldots, \bar{F}_d)$, the LMI is said to be 'fully separable', and is equivalent to d lower dimensional LMIs, $\bar{F}_i \succeq 0$, $i = 1, \ldots, d$. This idea can be extended to the case of 'partially separable' LMIs, particularly those in which the sparsity is characterized by a chordal graph. For the purpose of decomposition, the key property of matrices with chordal sparsity is as follows: partially complete symmetric matrices (i.e. some entries specified, other unspecified) have a positive definite completion if and only if: (i) the graph describing the positions of the specified entries is chordal (and includes the diagonals), and (ii) the sub-matrices corresponding to the cliques of the chordal graph are themselves positive definite [79]. This property, first exploited in [67, 158], allows large positive definiteness constraints $X \succeq 0$ to be decomposed into many smaller positive definiteness constraints $X(c_r, c_r) \succeq 0$, $r = 1, \ldots, l$, where c_r are the cliques of the chordal graph describing the sparsity pattern of X. These decomposed SDPs can then be solved more efficiently (e.g. by introducing splitting variables and consistency constraints), especially when combined with first-order splitting methods, e.g. [46, 139, 223]. For further research on exploiting chordal sparsity in SDP, refer to [32, 107, 253] as well as [244] for a comprehensive overview.

In addition to sparsity, another important class of SDPs for which structural properties can be exploited are those arising from the application of the Kalman-Yakubovich-Popov lemma (so-called KYP-SDPs). Specialized KYP-SDP solvers such as [249] improve efficiency by solving the dual problem over a reduced set of decision variables, whereas [243] eliminates dual variables in the Newton equations of primal-dual method. For KYP-SDPs related to integral quadratic constraint (IQC) analysis, [102] presents a cutting plane method and an interior point algorithm with a custom barrier function based on a frequency domain integral. The work of [84] also presents specialized solvers for such problems: one which eliminates variables in Newtwon-Todd search direction equations, and another based on conjugate gradients.

Finally, the recent work [174] applies facial reduction to SDPs for which no strictly feasible solution exists, leading to simplified equivalent problems.

Alternating direction method of multipliers

Developed in the 1970s [69, 73], and studied extensively in the 80s and 90s [54, 55, 68], the alternating direction method of multipliers (ADMM) has become particularly popular in recent years for large scale convex optimization problems, such as those found in machine learning [30, 74, 228, 268], power flow optimization [46, 139], and network utility

maximization [148]. ADMM applies to problems of the form

$$\min_{\theta,z} \; f(\theta) + g(z), \quad \text{s.t. } A\theta + Bz = c, \tag{2.64}$$

for convex functions f and g. The method begins with formation of the *augmented Lagrangian*,

$$L_\rho = f(\theta) + g(z) + \mu'(A\theta + Bz - c) + \frac{\rho}{2}\|A\theta + Bz - c\|_2^2. \tag{2.65}$$

It is 'augmented' in the sense that L_ρ represents the usual Lagrangian for (2.64) augmented by the additional term $\frac{\rho}{2}\|A\theta + Bz - c\|_2^2$. Here, $\rho > 0$ is a user-specified *penalty parameter*, and μ is the Lagrange multiplier for the constraint $A\theta + Bz = c$. Problem (2.64) is solved by alternately minimizing (2.65) w.r.t. θ and z, and updating the multiplier μ, i.e., at the kth iteration, θ^k, z^k and μ^k are updated according to:

$$\begin{aligned}
\theta^{k+1} &= \arg\min_{\theta} \; L_\rho(\theta, z^k, \mu^k), \\
z^{k+1} &= \arg\min_{z} \; L_\rho(\theta^{k+1}, z, \mu^k), \\
\mu^{k+1} &= \mu^k + \rho(A\theta^{k+1} + Bz^{k+1} - c).
\end{aligned}$$

2.3 Model stability

In [131, §4.1], Ljung outlines some open problems in system identification that are considered to be worthy of further study. Featuring prominently in this list is a need for useful parametrizations of (nonlinear) models, in particular, classes of models for which "simulation stability could be tested with reasonable effort." Simulation stability of a model is desirable for (at least) two reasons. Foremost, an unstable model that diverges during open-loop simulation cannot be relied upon for accurate long term predictions. More subtly, low sensitivity of long-term behavior to initial conditions can improve the ability of a model to generalize to inputs not observed during the fitting/training procedure. In this sense, model stability can be interpreted as somewhat of a 'regularizer', that penalizes model complexity by constraining the behavior of the model. This view is touched upon in [232], and further explored in Chapter 3.

In this section, we detail some recently developed parametrizations of nonlinear dynamical systems for which model stability can be guaranteed *a priori*, and review some more well-established approaches used in the linear setting.

2.3.1 Notions of stability

Stability of linear systems

Consider the state space representation of discrete time LTI dynamics,

$$x_{t+1} = Ax_t + Bu_t. \tag{2.66}$$

Such a system is said to be globally *asymptotically stable* if the unforced (i.e. $u = 0$) solution $x_t = A^t x_0$ from initial state x_0 satisfies $|x_t|^2 \to 0$ for all x_0. For such an LTI system, stability is completely characterized by the eigenvalues of A, also known as the system poles. In fact, we have the following well known result:

Theorem 2.2. Consider a discrete time LTI system $x_{t+1} = Ax_t + Bu_t$. The following statements are equivalent:

i. The system is globally asymptotically stable.

ii. The spectral radius of A is less than unity, i.e. A is a Schur matrix.

iii. There exists $P \in \mathbb{S}^{n_x}_{++}$ such that $A'PA - P \prec 0$.

The condition $A'PA - P \prec 0$ is called a Lyapunov equation. In Lyapunov stability theory, $V(x) = x'Px$ with $P \in \mathbb{S}^{n_x}_{++}$ defines a Lyapunov function, and

$$A'PA - P \prec 0 \iff x'_{t+1}Px_{t+1} - x'_tPx_t < 0,\ \forall t \iff V(x_{t+1}) < V(x_t),\ \forall t,$$

i.e., the Lyapunov function decreases uniformly with the evolution of the system. From this we can conclude that $V(x_t) \to 0$ as $t \to \infty$. As $P \in \mathbb{S}^{n_x}_{++}$, $V(x_t) = x'_tPx_t \to 0$ implies $x_t \to 0$.

For an LTI system, all notions of stability are equivalent, i.e. Theorem 2.2 implies Lyapunov, (global) asymptotic, (global) exponential, and finite-gain $\mathcal{L}_2$ stability of the system.

Stability of nonlinear systems

For a nonlinear dynamical system we must be more precise, as there are many notions of stability, owing to the complexity of behavior exhibited by such systems. Some common notions of stability for nonlinear systems include:

- Stability in the sense of Lyapunov, c.f., e.g., [106, §4].
- Input-output stability, originally developed in [266] and [267], c.f., also [198, 256] and [106, §5].

- Input-to-state stability, introduced in [217], see also [216].
- Limit cycles, dating back to the work of Poincaré [180], c.f., also [140, 143, 260].
- Incremental Lyapunov [9] and contraction theory [135].

It is natural to ask: what kind of stability properties would we like our identified nonlinear models to possess? One such notion, proposed and advocated for in [232], is as follows:

Definition 2.3 (Global incremental ℓ^2 stability)**.** The model (2.5) is said to be globally incremental ℓ^2 stable if the sequences $\{\bar{y}_t - \hat{y}_t\}_{t=1}^{\infty}$ and $\{\bar{x}_t - \hat{x}_t\}_{t=1}^{\infty}$ are square summable for every two solutions $(\bar{u}, \bar{x}, \bar{y})$ and $(\hat{u}, \hat{x}, \hat{y})$ of (2.5), subject to the same input $\bar{u} = \hat{u}$.

Incremental ℓ^2 stability implies convergence of trajectories under same inputs, regardless of initial conditions, i.e. the model 'forgets about' initial conditions. This reduced sensitivity to initial conditions can help improve the long-term predictive ability of the model, and also ensure a reliable response to a wide-variety of inputs. Furthermore, if $(u, x, y) = (0, 0, 0)$ is a valid solution, then global asymptotic stability of the origin is also implied.

Parameterizations of stable models

Deriving parametrizations of stable nonlinear dynamical systems is straightforward, *if* one constrains oneself to (nonlinear) finite impulse response models, c.f. (2.1). It is evident that the output y_t of such a model will decay to zero in finite time if the input u_t is set to zero, regardless of past behavior. Difficulties arrive when one incorporates *feedback* into the model structure. Feedback is necessary for a parsimonious description of resonant systems, and essential to some uniquely nonlinear behaviors such as limit cycles; however, it also introduces the possibility for instability. The main challenge in deriving parametrizations of stable dynamical systems is the nonconvexity of the simultaneous search for both the model parameters, and a certificate of model stability, e.g. a Lyapunov function or contraction metric. Consider the LTI dynamics in (2.66). To verify stability of a given system (i.e. the case where A is known), we must find $P \in \mathbb{S}^{n_x}_{++}$ that satisfies $A'PA - P \prec 0$. The Lyapunov equation is linear in P, and the search can be formulated as a convex program (SDP). The simultaneous search for *both* A and P is, however, nonconvex.

2.3.2 Convex parametrizations of stable linear models

To circumvent the nonconvexity of the simultaneous search for A and P in identification of stable LTI models, one strategy is to introduce a new variable $\mathcal{A}$ which the product of P and A, i.e. $\mathcal{A} = PA$. This is the approach adopted in [113]. Then the Lyapunov stability condition can be expressed as

$$A'PA - P \preceq -\delta I \iff \mathcal{A}'P^{-1}\mathcal{A} - P \preceq -\delta I, \tag{2.67}$$

where $\delta > 0$ is some arbitrarily small, positive constant. By an application of the Schur complement, the right hand side of (2.67) is equivalent to

$$\begin{bmatrix} P - \delta I & \mathcal{A} \\ \mathcal{A}' & P \end{bmatrix} \succeq 0, \tag{2.68}$$

which is a LMI in P and $\mathcal{A}$.

A related approach, introduced in [230], introduces an *implicit* representation of the LTI dynamics in (2.66), i.e.,

$$Ex_{t+1} = Fx_t + Ku_t. \tag{2.69}$$

Here, E is full rank, so an explicit representation as in (2.66) can be recovered as $A = E^{-1}F$ and $B = E^{-1}K$. With this implicit representation, one can define the Lyapunov function $V_{\mathrm{E}}(x) = |Ex|^2_{P^{-1}}$, for $P \in \mathbb{S}_{++}$. Notice that, for the unforced state transition $Ex_{t+1} = Fx_t$, we have $V_{\mathrm{E}}(x_{t+1}) = |Ex_{t+1}|^2_{P^{-1}} = |Fx_t|^2_{P^{-1}}$. Then, the Lyapunov inequality $V_{\mathrm{E}}(x_{t+1}) - V_{\mathrm{E}}(x_t) < 0$ is equivalent to

$$|Fx_t|^2_{P^{-1}} - |Ex_t|^2_{P^{-1}} < 0 \implies |Fx_t|^2_{P^{-1}} - x_t' \left(2E - P\right) x_t < 0.$$

The implication makes use of the inequality

$$-|Ex_t|^2_{P^{-1}} \leq -x_t' \left(2E - P\right) x_t,$$

which is a special case of the following simple linear upper bound on a concave quadratic function,

$$-a'Qa \leq b'Q^{-1}b - 2b'a, \quad \forall a, b, Q \in \mathbb{S}_{++} \tag{2.70}$$

with $a = Ex_t$, $b = x_t$ and $Q = P^{-1}$. By an application of the Schur complement,

$$F'P^{-1}F - E - E' + P \preceq -\delta I \iff \begin{bmatrix} E + E' - P - \delta I & F' \\ F & P \end{bmatrix} \succeq 0. \tag{2.71}$$

Notice that for $E = P$, (2.71) reduces to (2.68). Despite this equivalence, the advantages of this alternative formulation are twofold:

1. The implicit formulation of the dynamics is essential to the extension to the nonlinear case, which we discuss next.

2. The additional flexibility of the implicit formulation also improves the accuracy of certain convex relaxations for simulation error minimization, as discussed in Section 2.4.2.

Before proceeding to the nonlinear case, we wish to emphasize that such convex parametrizations of stable models are not the only approach to ensuring stability of identified linear

models. For instance, for inputs generated by an autoregressive process, stability of linear ARX models is guaranteed when the length of the training data sequence is sufficiently long [195]. Stability of identified state space models generated by subspace identification was studied in [138], where it was noted that stability can be enforced by inserting blocks of zeros in the shifted state matrix. In [238] stability was ensured via regularization, and a method to constrain pole locations to polytopic convex sets was presented in [154].

2.3.3 Convex parametrizations of stable nonlinear models

In contrast to the linear case, there are very few published methods that guarantee stability of identified nonlinear models *a priori*. In practice, stability is often verified empirically (posteriori) via extensive simulation. Of the few methods concerned with *a priori* stability guarantees, the work of [21] gives conditions under which passivity and small-gain stability properties are preserved for model reduction of a high order linear system in feedback with (comparatively lower order) nonlinear system.

In this section, we review a recently developed family of methods for deriving convex parametrizations of stable nonlinear models based on dissipativity and contraction theory, c.f., [25, 141, 150, 229, 230, 232].

Implicit models

As alluded to in §2.3.2, the convex parametrization of stable models proposed by [232] utilizes an *implicit* representation of nonlinear state space systems, i.e.,

$$e(x_{t+1}) = f(x_t, u_t), \tag{2.72a}$$

$$y_t = g(x_t, u_t), \tag{2.72b}$$

where e, f, g are linearly parametrized vector functions of the form

$$e(x) = \sum_{i=1}^{n_\theta} \theta_i e_i(x),\ \ f(x,u) = \sum_{i=1}^{n_\theta} \theta_i f_i(x,u),\ \ g(x,u) = \sum_{i=1}^{n_\theta} \theta_i g_i(x,u). \tag{2.73}$$

Here $\theta \in \mathbb{R}^{n_\theta}$ denotes the model parameters. Popular choices for the basis functions $e_i : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x}$, $f_i : \mathbb{R}^{n_x \times n_u} \mapsto \mathbb{R}^{n_x}$, $g_i : \mathbb{R}^{n_x \times n_u} \mapsto \mathbb{R}^{n_y}$ include vectors of multivariate polynomials or trigonometric polynomials, as this permits the use of sums-of-squares optimization techniques, as discussed in the sequel.

The model (2.72) is said to be *well-posed* when $e(\cdot)$ is a bijection, such that an explicit nonlinear state space model of the form (2.5) can be recovered with $a(x,u) = e^{-1}(f(x,u))$. Sufficient conditions for well-posedness of the model are given below; see also [232, Theorem 1].

Incremental Lyapunov condition for stability

The idea behind the convex parametrizations presented in [232, §3] is to enforce incremental ℓ_2 stability via an incremental Lyapunov approach. For two solutions of (2.72), $\bar{x}$ and $\hat{x}$, consider the inequality

$$V(\bar{x}_{t+1}, \hat{x}_{t+1}) - V(\bar{x}_t, \hat{x}_t) + |g(\bar{x}_t, u_t) - g(\hat{x}_t, u_t)|^2 \leq -\mu|\bar{x}_t - \hat{x}_t|^2, \tag{2.74}$$

where $V : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \mapsto \mathbb{R}$ is a positive definite incremental Lyapunov function.

Summation of (2.74) yields

$$\sum_{t=0}^{T} |g(\bar{x}_t, u_t) - g(\hat{x}_t, u_t)|^2 + \mu \sum_{t=0}^{T} |\bar{x}_t - \hat{x}_t|^2 \leq V(\bar{x}_0, \hat{x}_0) - V(\bar{x}_T, \hat{x}_T) \leq V(\bar{x}_0, \hat{x}_0) \tag{2.75}$$

which is clearly sufficient for incremental ℓ_2 stability.

The practical utility of this approach hinges on the convexity of the simultaneous search for both model parameters of (2.72) and an incremental Lyapunov function V such that (2.74) holds. This is where the implicit dynamics of (2.72) proves useful, in much the same way that the implicit dynamics of (2.69) convexified the simultaneous search for LTI model parameters and a Lyapunov function in (2.71).

The authors of [232] propose an incremental Lyapunov function of the form

$$V(\bar{x}, \hat{x}) = |e(\bar{x}) - e(\hat{x})|^2_{P^{-1}}.$$

Notice that

$$V(\bar{x}_{t+1}, \hat{x}_{t+1}) = |e(\bar{x}_{t+1}) - e(\hat{x}_{t+1})|^2_{P^{-1}} = |f(\bar{x}_t, u_t) - f(\hat{x}_t, u_t)|^2_{P^{-1}} \tag{2.76}$$

for solutions of (2.72). With this choice of V the Lyapunov condition (2.74) becomes

$$|f(\bar{x}, u) - f(\hat{x}, u)|^2_{P^{-1}} - |e(\bar{x}) - e(\hat{x})|^2_{P^{-1}} + |g(\bar{x}, u) - g(\hat{x}, u)|^2 \leq -\mu|\bar{x} - \hat{x}|^2. \tag{2.77}$$

For given $(\bar{x}, \hat{x}, u)$, (2.77) is nonconvex in the model parameters, as $-|e(\bar{x}) - e(\hat{x})|^2_{P^{-1}}$ is concave quadratic (recall that e, f, g are linearly parametrized). To address this, the authors of [232] apply the inequality

$$-|e(\bar{x}) - e(\hat{x})|^2_{P^{-1}} \leq |\bar{x} - \hat{x}|^2_P - 2(\bar{x} - \hat{x})'(e(\bar{x}) - e(\hat{x})), \tag{2.78}$$

which is a special case of (2.70). Substituting (2.78) into (2.77) gives

$$|f(\bar{x}, u) - f(\hat{x}, u)|^2_{P^{-1}} - 2(\bar{x} - \hat{x})'(e(\bar{x}) - e(\hat{x})) + |\bar{x} - \hat{x}|^2_{\mu I + P} + |g(\bar{x}, u) - g(\hat{x}, u)|^2 \leq 0 \tag{2.79}$$

which is sufficient for (2.77), but has the advantage of being convex in e, f, g for given

$(\bar{x}, \hat{x}, u)$. In summary, we have the following result:

Theorem 2.3. [232, Theorem 2] For any model of the form (2.72), existence of $P \in \mathbb{S}^{n_x}_{++}$ such that (2.79) holds for all $(\bar{x}, \hat{x}, u)$ is sufficient for incremental ℓ_2 stability.

Contraction condition for stability

In this section, we present contraction conditions for incremental ℓ_2 stability of (2.72). Contraction analysis studies the dynamics of *virtual displacements*, i.e. differences between infinitesimally close trajectories of dynamical systems [135]. Parametrizations derived from contraction theory are typically simpler than those based on dissipation inequalities. Furthermore, there are strong connections between contraction conditions and quality-of-fit metrics based on linearized model behavior, c.f. Section 2.4.3 and Chapter 3.

Consider two trajectories, x_t and $\bar{x}_t$, of (2.72). The first order Taylor series approximation of (2.72) about x_t is given by

$$e(x_{t+1}) + E(x_{t+1})(\bar{x}_{t+1} - x_{t+1}) \approx f(x_t, u_t) + F(x_t, u_t)(\bar{x}_t - x_t) \tag{2.80}$$

$$\bar{y}_t \approx g(x_t, u_t) + G(x_t, u_t)(\bar{x}_t - x_t), \tag{2.81}$$

which, after some elementary substitutions from (2.72), reduces to

$$E(x_{t+1})(\bar{x}_{t+1} - x_{t+1}) \approx F(x_t, u_t)(\bar{x}_t - x_t) \tag{2.82}$$

$$\bar{y}_t - y_t \approx G(x_t, u_t)(\bar{x}_t - x_t). \tag{2.83}$$

When the two trajectories are close, then this linear approximation will accurately describe the dynamics of the virtual displacement $\bar{x}_t - x_t$. This motivates the introduction of the so-called *differential dynamics*,

$$E(x_{t+1})\Delta_{t+1} = F(x_t, u_t)\Delta_t, \tag{2.84}$$

$$\Delta^y_t = G(x_t, u_t)\Delta_t, \tag{2.85}$$

where $E(x) = \frac{\partial}{\partial x} e(x)$, $\Delta_t = \bar{x}_t - x_t$, $F(x, u) = \frac{\partial}{\partial x} f(x, u)$, $\Delta^y_t = \bar{y}_t - y_t$, and $G(x, u) = \frac{\partial}{\partial x} g(x, u)$. These differential dynamics motivate the following differential dissipation inequality

$$V(x_{t+1}, \Delta_{t+1}) - V(x_t, \Delta_t) \leq -|G(x_t, u_t)\Delta_t|^2 - \mu|\Delta_t|^2 \tag{2.86}$$

with differential storage function $V : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \mapsto \mathbb{R}$. If (2.86) holds for solutions x_t and Δ_t of (2.72) and (2.84), respectively, then by summation of (2.86)

$$\sum_{t=0}^{T} |G(x_t, u_t)\Delta_t|^2 + \mu \sum_{t=0}^{T} |\Delta_t|^2 \leq V(x_0, \Delta_0) - V(x_T, \Delta_T) \leq V(x_0, \Delta_0) \tag{2.87}$$

for all T. Before detailing the specific choice of V proposed in [232], let us first discuss

the reasoning that links (2.86) to incremental ℓ_2 stability, from a contraction analysis perspective. Consider two infinitesimally close solutions of (2.72), denoted (x_t, y_t) and $(\bar{x}_t, \bar{y}_t)$, driven by the same input u_t. The difference between these two trajectories will satisfy the differential dynamics (2.84). Then (2.87) says that $\sum_{t=0}^{T} |y_t - \bar{y}_t|^2$ is finite for all T, which implies that $\bar{y}_t \to y_t$ as $t \to \infty$, i.e., the outputs of the two infinitesimally close trajectories converge asymptotically. If the dissipation inequality (2.86) holds for all $x_{t+1}, x_t, \Delta_{t+1}, \Delta_t$, then this analysis holds for every pair of infinitesimally close trajectories; i.e., the outputs of all pairs of nearby trajectories will converge, which implies that the outputs of all trajectories (subject to the same inputs) will converge, regardless of initial conditions.

Once more, convexity of the simultaneous search for model parameters and a contraction metric (a.k.a differential storage function) is key to the practical utility of this approach. The authors of [232] propose the Riemannian metric $V(x, \Delta) = |E(x)\Delta|^2_{P^{-1}}$ with $P \in \mathbb{S}^{n_x}_{++}$. Notice that $V(x_{t+1}, \Delta_{t+1}) = |E(x_{t+1})\Delta_{t+1}|^2_{P^{-1}} = |F(x_t, u_t)\Delta_t|^2_{P^{-1}}$ for solutions of (2.72) and (2.84). With this choice of V the dissipation inequality (2.86) becomes

$$|F(x,u)\Delta|^2_{P^{-1}} - |E(x)\Delta|^2_{P^{-1}} + |G(x,u)\Delta|^2 \leq -\mu|\Delta|^2. \tag{2.88}$$

For given (x, u), (2.88) is nonconvex in the model parameters, as $-|E(x)\Delta|^2_{P^{-1}}$ is concave quadratic (recall that e, f, g and thus E, F, G are linearly parametrized). The authors of [232] address this with an application the inequality

$$-|E(x)\Delta|^2_{P^{-1}} \leq \Delta' P \Delta - \Delta'(E(x) + E(x)')\Delta, \tag{2.89}$$

which is a special case of (2.70) to obtain

$$F(x,u)'P^{-1}F(x,u) + P - E(x) - E(x)' + G(x, u'G(x,u)) \preceq -\mu I. \tag{2.90}$$

From (2.89), (2.90) implies (2.88) but has the advantage of being convex in e, f, g and P. This can be clearly seen by taken the Schur complement of (2.90), leading to

$$\begin{bmatrix} E(x) + E(x)' - P - \mu I & F(x,u)' & G(x,u) \\ F(x,u) & P & 0 \\ G(x,u) & 0 & I \end{bmatrix} \succeq 0.$$

We now arrive at the following result for incremental ℓ_2 stability:

Theorem 2.4. [232, Theorem 5] For any model of the form (2.72), existence of $P \in \mathbb{S}^{n_x}_{++}$ such that (2.90) holds for all x, u, is sufficient for incremental ℓ_2 stability. Furthermore, such a model is well-posed.

2.3.4 Related properties

The convexification procedures described in this chapter can be extended to enforce other behavioral properties of dynamical systems, in addition to stability, by appropriate choice of storage function and supply rate; c.f., [232, Remark 1]. For example, passivity [106, §6] of the identified model can be ensured via the dissipation inequality,

$$V(x_{t+1}) - V(x_t) \leq -u_t' g(x_t, u_t), \tag{2.91}$$

with storage function $V(x) = |e(x)|_{P^{-1}}^2$ and supply rate $-u_t' g(x_t, u_t)$. Summing (2.91) for solutions of (2.5) gives $\sum_{t=0}^{T} u_t' y_t \leq V(x_0)$ for all T, which implies passivity of the model.

In the work of [144], these ideas were extended to convex parametrizations of models with stable limit cycles. In a stable limit cycle, solutions of the system converge to the same *path* (through state space) but not necessarily the same *trajectory*, as differences in phase may persist. Incremental ℓ_2 stability, i.e. the property that all solutions will eventually converge regardless of initial conditions, is therefore 'too strong' a notion of stability to impose if one wishes to identify systems with stable limit cycles. The approach adopted in [144] enforces contraction only in directions (of the state space) transverse to the limit cycle, thereby allowing differences in phase to persist; c.f., also [143].

2.4 Convex bounds on simulation error

For linearly parametrized models, equation error quality-of-fit metrics are convex in the model parameters; however, small equation error does not guarantee good long-term predictive performance. To achieve the latter, it may be desirable to minimize simulation error (a.k.a output error). Such an optimization is nonconvex due to dependence on multi-step ahead simulated output of the model. In this section we review a family of methods [25, 141, 150, 229–232] that seek a middle ground between these two approaches: convex upper bounds on simulation error.

2.4.1 Incremental gain from equation error to simulation error

This line of research was introduced in a paper [150] by Megretski, which built upon earlier work on relaxation based model reduction techniques [149, 218]. The paper proposed an incremental ℓ_2 gain condition under which small equation error does imply small simulation error. For the purpose of exposition, consider an implicit nonlinear ARX model of the form,

$$f(y_t, y_{t-1}, \ldots, y_{t-n_a}, u_{t-1}, \ldots, u_{t-n_b}) = 0. \tag{2.92}$$

Note that this structure can incorporate the NARX model (2.4) by choosing

$$f(y_t, y_{t-1}, \ldots, y_{t-n_a}, u_{t-1}, \ldots, u_{t-n_b}) = y_t - f_{\mathrm{AR}}(y_{t_1}, \ldots, y_{t-n_a}, u_{t-1}, \ldots, u_{t-n_b}).$$

Equation error for (2.92) is then given by

$$\epsilon_t = f(\tilde{y}_t, \tilde{y}_{t-1}, \ldots, \tilde{y}_{t-n_a}, \tilde{u}_{t-1}, \ldots, \tilde{u}_{t-n_b}). \tag{2.93}$$

To study the relationship between equation error and simulation error, one can introduce the error model

$$f(y_t, y_{t-1}, \ldots, y_{t-n_a}, u_{t-1}, \ldots, u_{t-n_b}) = \epsilon_t, \tag{2.94}$$

where ϵ_t should be thought of as an additional input to the model.

Definition 2.4 (Incremental ℓ_2 gain of error model)**.** Consider two solutions, $\bar{y}_t$ and $\hat{y}_t$, of (2.94) driven by two different inputs $(\bar{\epsilon}, \bar{u})$ and $\hat{\epsilon}, \hat{u}$, respectively, but starting from the same initial conditions. The smallest $\gamma > 0$ such that

$$\sum_{t=0}^{T} |\bar{y}_t - \hat{y}_t|^2 \leq \gamma \sum_{t=0}^{T} |\hat{\epsilon}_t - \hat{\epsilon}_t|^2 \tag{2.95}$$

is called the incremental ℓ_2 gain of the error model.

If the error model has finite incremental ℓ_2 gain, then by

i. setting the initial conditions to those of the measured data, i.e., $\bar{y}_t = \hat{y}_t = \tilde{y}_t$ for $t = -1, \ldots, -n_a$,

ii. setting $\bar{u}_t = \hat{u}_t = \tilde{u}_t$ for all t,

iii. setting $\bar{\epsilon}_t = \epsilon_t = f(\tilde{y}_t, \tilde{y}_{t-1}, \ldots, \tilde{y}_{t-n_a}, \tilde{u}_{t-1}, \ldots, \tilde{u}_{t-n_b})$ such that $\bar{y}_t = \tilde{y}_t$,

iv. setting $\hat{y}_t = 0$ such that $\hat{y}_t = y_t$, where y_t is the simulated output of (2.92),

the inequality (2.95) implies

$$\sum_{t=0}^{T} |\tilde{y}_t - y_t|^2 \leq \gamma \sum_{t=0}^{T} |\epsilon_t|^2, \tag{2.96}$$

i.e., equation error upper bounds simulation error. Finite incremental ℓ_2 gain is implied by the dissipation inequality

$$\begin{aligned}\gamma|f(\bar{y}_t, \ldots, \bar{y}_{t-n_a}, \bar{u}_{t-1}, \ldots, \bar{u}_{t-n_b}) - f(\hat{y}_t, \ldots, \hat{y}_{t-n_a}, \hat{u}_{t-1}, \ldots, \hat{u}_{t-n_b})|^2 - |\hat{y}_t - \bar{y}_t|^2 \geq \\ V(\bar{y}_t, \ldots, \bar{y}_{t-n_a}, \hat{y}_t, \ldots, \hat{y}_{t-n_a}) - V(\bar{y}_{t-1}, \ldots, \bar{y}_{t-n_a-1}, \hat{y}_{t-1}, \ldots, \hat{y}_{t-n_a-1}).\end{aligned} \tag{2.97}$$

Summing (2.97) for solutions $(\bar{y}, \bar{u}, \bar{\epsilon})$ and $(\hat{y}, \hat{u}, \hat{\epsilon})$ yields (2.95) as

$$V(\bar{y}_T, \ldots, \bar{y}_{T-n_a}, \hat{y}_T, \ldots, \hat{y}_{T-n_a}) \geq 0, \quad \forall \bar{y},\ \hat{y},$$

and $V(\bar{y}_{-1}, \ldots, \bar{y}_{-n_a}, \hat{y}_{-1}, \ldots, \hat{y}_{-n_a}) = 0$ when $\bar{y}_t = \hat{y}_t = \tilde{y}_t$ for $t = -1, \ldots, -n_a$ (i.e. both solutions begin from the same initial conditions).

For linearly parametrized f and V, the dissipation inequality (2.97) is nonconvex in the model parameters. We make use of the linear upper bound on concave quadratic functions given in (2.70) to obtain

$$\begin{aligned} -\gamma|f(\bar{y}_t, \ldots, \bar{u}_{t-n_b}) - f(\hat{y}_t, \ldots, \hat{u}_{t-n_b})|^2 \leq \\ \frac{1}{\gamma}|\bar{y}_t - \hat{y}_t|^2 - 2(\bar{y}_t - \hat{y}_t)'(f(\bar{y}_t, \ldots, \bar{u}_{t-n_b}) - f(\hat{y}_t, \ldots, \hat{u}_{t-n_b})). \end{aligned} \tag{2.98}$$

Substituting (2.98) into (2.97) gives convex sufficient conditions for finite incremental ℓ_2 gain of the error model,

$$\begin{aligned} 0 \geq \frac{1}{\gamma}|\bar{y}_t - \hat{y}_t|^2 - 2(\bar{y}_t - \hat{y}_t)'(f(\bar{y}_t, \ldots, \bar{u}_{t-n_b}) - f(\hat{y}_t, \ldots, \hat{u}_{t-n_b})) + |\hat{y}_t - \bar{y}_t|^2 + \\ V(\bar{y}_t, \ldots, \bar{y}_{t-n_a}, \hat{y}_t, \ldots, \hat{y}_{t-n_a}) - V(\bar{y}_{t-1}, \ldots, \bar{y}_{t-n_a-1}, \hat{y}_{t-1}, \ldots, \hat{y}_{t-n_a-1}). \end{aligned} \tag{2.99}$$

In summary, minimization of equation error (2.93) subject to the convex constraint (2.99) upper bounds simulation error, and ensures stability of the identified model. We will return to the concept of incremental gain of error models in Chapter 5 to derive convex upper bounds on simulation error for positive dynamical systems.

2.4.2 Lagrangian relaxation of simulation error

In this same paper [150], Megretski also proposed a more accurate convex upper bound on simulation error, based on a version of the Lagrangian relaxation [119]. Lagrangian relaxation refers to a family of methods for generating convex approximations to optimization problems rendered nonconvex by 'difficult' constraints, such as combinatorial [160] and integer programming problems [61]. In the control community, the technique is known as the S-procedure [263] and has long been used in applications such as stability analysis [29] and robust control [201], even before the theoretical properties of the method, i.e. the S-lemma, were fully understood [181].

For our purposes, the variant of the Lagrangian relaxation we shall use concerns constrained optimization problems of the form

$$\min_{\theta, x} J(\theta, x) \text{ s.t. } F(\theta, x) = 0, \tag{2.100}$$

where J and F are assumed to be convex and affine, respectively, in θ. For a concrete example, consider minimization of simulation error for a linearly parametrized state space model of the form (2.5). Here, θ denotes the model parameters, and $x = x_{1:T}$ denotes the internal states of the model. The cost function $J(\theta, x) = \sum_{t=1}^{T} |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2$ is the

sum-of-squares error between the model output and the measured data, and

$$F(\theta, x) = \begin{bmatrix} x_1 - \tilde{x}_1 \\ x_2 - a(x_1, \tilde{u}_1) \\ \vdots \\ x_T - a(x_{T-1}, \tilde{u}_{T-1}) \end{bmatrix} = 0$$

encodes the dynamics of the model.

The Lagrangian relaxation of (2.100) is then given by the convex program

$$\min_{\theta} \{\hat{J}_\lambda(\theta) \triangleq \sup_{x} \; J(\theta, x) - \lambda(x)' F(\theta, x)\} \tag{2.101}$$

where

1) It is convex in θ. Recall that J and F are convex and affine in θ, respectively. As such, $\hat{J}_\lambda(\theta)$ is the supremum of an infinite family of convex functions, and is therefore convex in θ; see §3.2.3 of [28].

2) It is an upper bound for the original problem (2.100). Given θ, let x^* be any x such that $F(\theta, x^*) = 0$. Then

$$J(\theta, x^*) + \lambda F(\theta, x^*) = J(\theta, x^*),$$

which implies that the supremum over all x can be no smaller; i.e. $\hat{J}_\lambda(\theta)$ is an upper bound for the simulation error.

For minimization of simulation error, the Lagrangian relaxation in (2.101) takes the form

$$\hat{J}_\lambda(\theta) = \sup_{x} \left\{ \sum_{t=1}^{T} |\tilde{y}_t - g(\theta, x_t)|^2 - \sum_{t=1}^{T-1} \lambda_{t+1}(x)' \left(x_{t+1} - a(x_t, \tilde{u}_t)\right) \right\}. \tag{2.102}$$

It is known that the Lagrangian relaxation of equivalent constraints gives non-equivalent bounds. Specifically, redundant parametrizations of the constraint function $F(\theta, x)$ in (2.100) provides additional flexibility that leads to tighter bounds, i.e. lower values of $\min_\theta \; \hat{J}_\lambda(\theta)$. For Lagrangian relaxation of output error, as in (2.102), redundancy can be introduced to $F(\theta, x)$ by using an *implicit* representation of the dynamics, as in (2.72). When $a(\cdot, \cdot) = e^{-1}(f(\cdot, \cdot))$, the constraints

$$\begin{bmatrix} x_1 - \tilde{x}_1 \\ x_2 - a(x_1, \tilde{u}_1) \\ \vdots \\ x_T - a(x_{T-1}, \tilde{u}_{T-1}) \end{bmatrix} = 0 \quad \text{and} \quad \begin{bmatrix} e(x_1) - e(\tilde{x}_1) \\ e(x_2) - f(x_1, \tilde{u}_1) \\ \vdots \\ e(x_T) - f(x_{T-1}, \tilde{u}_{T-1}) \end{bmatrix} = 0$$

have the same feasible set, $x_{1:T}$, and so the upper bound given by

$$\hat{J}_\lambda(\theta) = \sup_x \left\{ \sum_{t=1}^{T} |\tilde{y}_t - g(\theta, x_t)|^2 - \sum_{t=1}^{T-1} \lambda_{t+1}(x)' \left(e(x_{t+1}) - f(x_t, \tilde{u}_t) \right) \right\} \tag{2.103}$$

will, in general, be tighter than that of (2.102).

Computational complexity

The function $\hat{J}_\lambda(\theta)$, as defined in (2.103), gives a convex upper bound on output error; however, optimization of $\hat{J}_\lambda(\theta)$ is far from straightforward. Foremost, for nonlinear models evaluation of $\hat{J}_\lambda(\theta)$ requires computing the supremum of the Lagrangian,

$$\sum_{t=1}^{T} |\tilde{y}_t - g(\theta, x_t)|^2 - \sum_{t=1}^{T-1} \lambda_{t+1}(x)' \left(e(x_{t+1}) - f(x_t, \tilde{u}_t) \right),$$

which is nonlinear (and, in general, nonconcave) in x. When the functions e, f, g are given by polynomials (or trigonometric polynomials), as proposed in Section 2.3.3, the supremum in (2.103) may be approximated by sum-of-squares (SOS) methods; however, due to the very large number of indeterminate variables (i.e. $x_{1:T}$), such an approach scales poorly. Whether recent developments that take advantage of chordal sparsity in SOS programing, e.g. [248], can improve tractability is a subject for future research.

In the following sections we outline two approaches to improve the computational tractability of the Lagrangian relaxation: minimization of linearized simulation error, c.f. Section 2.4.3, and approximation of the Lagrangian relaxation with dissipation inequalities, c.f. Section 2.4.4.

Surrogate state sequences

The developments of the following sections depend on a *surrogate* state sequence $\{\tilde{x}_t\}_{t=1}^T$. While it is not assumed that these are *true* internal states, the more accurate they are the more effective these approaches will be. For linear systems, subspace algorithms provide an effective method for generating state estimates from input-output data [241]. For nonlinear systems, state estimation is more challenging and solutions can be quite case specific. Possible strategies include: subspace methods in the case of weakly nonlinear systems, c.f. the example in Section 3.6; exploiting physical or structural knowledge, c.f. the example in Section 3.5; alternating between model-based state estimation and model refinement, e.g. Expectation Maximization, c.f. Chapter 4; and using truncated histories of inputs and outputs (as in NARX) [208].

2.4.3 Linearized simulation error

The principle difficulty associated with Lagrangian relaxation for nonlinear models, is the computation of the supremum in (2.103). For the special case of linear models, the Lagrangian in (2.103) is quadratic in x (for multipliers $\lambda(x)$ that are affine in x), which allows the supremum to be calculated analytically. This motivates the study of linearized simulation error. Roughly speaking, this can be thought of as the simulation error of the linear time varying model obtained by linearizing the nonlinear model about some trajectory, $\{\tilde{x}_t\}_{t=1}^T$, c.f. the discussion on *surrogate* state sequences in Section 2.4.2.

More precisely, we introduce a perturbed version of the nonlinear model in (2.72),

$$e(x_{t+1}^\rho) = f(x_t^\rho, u_t) + \rho\epsilon_t \tag{2.104a}$$

$$y_t^\rho = g(x_t^\rho, u_t) + \rho\eta_t, \tag{2.104b}$$

where $\rho \in [0, 1]$ and

$$\epsilon_t = e(\tilde{x}_{t+1}) - f(\tilde{x}_t, \tilde{u}_t), \quad \eta_t = \tilde{y}_t - g(\tilde{x}_t, \tilde{u}_t)$$

denote the equation errors. Observe that when $\rho = 0$, (2.104) reduces to (2.72) and so $y_t^0 = y_t$, i.e., the usual unperturbed simulated output. Conversely, when $\rho = 1$ we have $x_t^1 = \tilde{x}_t^1$ and $y_t^1 = \tilde{y}_t^1$, i.e., the simulated states and outputs of the perturbed system are equal to surrogate states and measured outputs, respectively. One could think of ρ as a 'leash' on the perturbed system (2.104): a 'tight leash' ($\rho = 1$) constrains (2.104) to exactly reproduce the measured output, whereas a 'loose leash' ($\rho = 0$) allows (2.104) to behave as freely as the unperturbed system. Linearized simulation error is then defined as

$$\mathcal{E}^0 := \lim_{\rho\to 1} \frac{1}{(1-\rho)^2} \sum_{t=1}^T |\tilde{y}_t - y_t^\rho|^2. \tag{2.105}$$

One application of L'Hospital's Rule gives

$$\mathcal{E}^0 = \lim_{\rho\to 1} \frac{-2\sum_{t=1}^T \left(\nabla_\rho y_t^\rho\right)' \left(\tilde{y}_t - y_t^\rho\right)}{-2(1-\rho)},$$

where

$$\nabla_\rho y_t^\rho = \frac{\partial}{\partial\rho}\left(g(x_t^\rho, \tilde{u}_t) + \rho\eta_t\right) = G(x_t^\rho, \tilde{u}_t)\frac{\partial x_t^\rho}{\partial\rho} + \eta_t.$$

A second application of L'Hospital's Rule yields,

$$\mathcal{E}^0 = \lim_{\rho\to 1} \frac{-2\sum_{t=1}^T \left(\nabla_\rho^2 y_t^\rho\right)' \left(\tilde{y}_t - y_t^\rho\right) + +2\sum_{t=1}^T \left(\nabla_\rho y_t^\rho\right)' \left(\nabla_\rho y_t^\rho\right)}{2} = \sum_{t=1}^T |\left(\nabla_\rho y_t^\rho|_{\rho=1}\right)|^2.$$

Linearized simulation error is then equivalent to $\mathcal{E}^0 = \sum_{t=1}^T |G(\tilde{x}_t, \tilde{u}_t)\Delta_t + \eta_t|^2$, where

$\Delta = \frac{\partial x_t^\rho}{\partial \rho}\Big|_{\rho=1}$ satisfies $\Delta_1 = 0$, as $x_1^\rho = \tilde{x}_1$ for all ρ, and

$$E(\tilde{x}_{t+1})\Delta_{t+1} = F(\tilde{x}_t, \tilde{u}_t)\Delta_t + \epsilon_t, \tag{2.106}$$

which follows from differentiation of (2.104a),

$$\frac{\partial}{\partial \rho} e(x_{t+1}^\rho) = E(x_{t+1}^\rho)\frac{\partial x_{t+1}^\rho}{\partial \rho} = \frac{\partial}{\partial \rho} f(x_t^\rho, \tilde{u}_t) = F(x_t^\rho, \tilde{u}_t)\frac{\partial x_t^\rho}{\partial \rho} + \epsilon_t.$$

The Lagrangian relaxation of minimization of linearized simulation error then takes the form

$$\hat{J}_\lambda(\theta) = \sup_\Delta \left\{ \sum_{t=1}^{T} |G(\tilde{x}_t, \tilde{u}_t)\Delta_t + \eta_t|^2 - \sum_{t=1}^{T-1} \lambda_{t+1}(\Delta)' \left(E(\tilde{x}_{t+1})\Delta_{t+1} - F(\tilde{x}_t, \tilde{u}_t)\Delta_t - \epsilon_t\right) \right\}, \tag{2.107}$$

which requires computing the supremum of a quadratic function in Δ. The ability to compute the supremum analytically represents a dramatic reduction in computational complexity compared to the Lagrangian relaxation of simulation error in (2.103). Nonetheless, complexity of the minimization of (2.107) with general purpose semidefinite programming (SDP) solvers grows cubicly with the number of data points, T. This is the motivation for the specialized algorithms presented in Chapter 3, which exploit structure in the Lagrangian relaxation to further improve computational tractability.

2.4.4 Robust identification error

One of the principle challenges to applying the Lagrangian relaxation in (2.103) is computing supremum of a function of many variables, namely the entire state sequence $x_{1:T}$. The idea behind the so-called Robust Identification Error (RIE), first introduced in [230], c.f. also [232, §4], is to approximate (2.103) with an alternative convex upper bound derived from dissipation inequalities. This leads to a more computationally tractable point-wise measure of model fit; to evaluate the upper bound at each data point, it is only necessary to compute the supremum of a function of a single state, x_t, as opposed to the suremum over the entire state sequence.

The key idea is as follows. For the general nonlinear deterministic state space model (2.5), restated here for convenience,

$$x_{t+1} = a(x_t, u_t), \tag{2.108}$$

$$y_t = g(x_t, u_t), \tag{2.109}$$

consider the following dissipation inequality,

$$V(a(x_t, \tilde{u}_t), t+1) - V(x_t, t) \leq s_t - |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2. \tag{2.110}$$

Here $V(x,t)$ denotes the storage function, $s_t - |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2$ denotes the supply rate, and s_t is a slack variable, the role of which will be made clear in the sequel. Suppose (2.110) holds for all $x_t \in \mathbb{R}^{n_x}$. Then it clearly also holds for the special case of solutions to the dynamical system, i.e., $x_{t+1} = a(x_t, \tilde{u}_t)$. Summing (2.110) for solutions to (2.5) gives

$$\sum_{t=1}^{T} V(a(x_t, \tilde{u}_t), t+1) - V(x_t, t) = V(x_{t+1}, t+1) - V(x_1, 1) \leq \sum_{t=1}^{T} s_t - |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2, \quad (2.111)$$

which implies the following upper bound on simulation error,

$$V(x_1, 1) + \sum_{t=1}^{T} s_t \geq \sum_{t=1}^{T} |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2. \quad (2.112)$$

As with the convex parametrization of stable models, c.f. Section 2.3.3, one of the principle challenges is the nonconvexity of the simultaneous search for storage function V and the dynamics $a(\cdot,\cdot)$, specifically due to the composition $V(a(x_t, \tilde{u}_t), t+1)$. Once again, the implicit dynamics (2.72) proves useful. In [230], the authors propose the specific choice of storage function

$$V(x_t, t) = |e(x_t) - e(\tilde{x}_t)|^2_{P^{-1}}, \quad (2.113)$$

which leads to

$$V(a(x_t, \tilde{u}_t), t+1) = |e(a(x_t, \tilde{u}_t)) - e(\tilde{x}_t)|^2_{P^{-1}} = |f(x_t, \tilde{u}_t)) - e(\tilde{x}_t)|^2_{P^{-1}}. \quad (2.114)$$

Notice that the troublesome composition has been replaced by $f(\cdot,\cdot) = e(a(\cdot,\cdot))$. Notice also that with $x_1 = \tilde{x}_1$, we have $V(x_1, 1) = 0$, and so from (2.112) we see that $\sum_t s_t$ upper bounds simulation error. To ensure that (2.110) holds for all x_t we can enforce

$$s_t = \sup_{x_t} |f(x_t, \tilde{u}_t)) - e(\tilde{x}_t)|^2_{P^{-1}} - |e(x_t) - e(\tilde{x}_t)|^2_{P^{-1}} + |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2, \quad (2.115)$$

where we have substituted the specific choice (2.113) of V into (2.110). It is clear that $\sum_t s_t$ upper bounds simulation error; however, we seek an upper bound that is convex in the model parameters, θ. The supremum of an infinite family of convex functions is itself convex; c.f. [28, §3.2.3]. Unfortunately, the family of functions

$$|f(x_t, \tilde{u}_t)) - e(\tilde{x}_t)|^2_{P^{-1}} - |e(x_t) - e(\tilde{x}_t)|^2_{P^{-1}} + |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2 \quad (2.116)$$

is not convex in θ, due to the concave term $-|e(x_t) - e(\tilde{x}_t)|^2_{P^{-1}}$. As in Section 2.3.3, we replace the troublesome concave term with an affine upper bound

$$-|e(x_t) - e(\tilde{x}_t)|^2_{P^{-1}} \leq |x_t - \tilde{x}_t|^2_P - 2(x_t - \tilde{x}_t)'(e(x_t) - e(\tilde{x}_t)), \quad (2.117)$$

which is a special case of the inequality in (2.70). We can now define the RIE, as presented

in [230], by

$$\mathcal{E}_t^{\text{rie}}(\theta) := \sup_{x_t} |f(x_t, \tilde{u}_t)) - e(\tilde{x}_t)|_{P^{-1}}^2 + |x_t - \tilde{x}_t|_P^2 - 2(x_t - \tilde{x}_t)'(e(x_t) - e(\tilde{x}_t)) + |\tilde{y}_t - g(x_t, \tilde{u}_t)|^2. \tag{2.118}$$

As $\mathcal{E}_t^{\text{rie}}(\theta) \geq s_t$, it is clear that $\sum_t \mathcal{E}_t^{\text{rie}}(\theta)$ gives a convex upper bound on simulation error; c.f. [232, Theorem 4]. Notice that the RIE in (2.118) only requires supremum over x_t rather than $x_{1:T}$, which dramatically improves computational tractability, especially for large T. Nevertheless, in general (2.118) still involves maximization of a nonconvex function in x_t, and so sum-of-squares (SOS) approximations are still necessary. To further reduce computational complexity, a similar dissipation inequality construction can be applied to linearized simulation error, discussed in Section 2.4.3, leading to the so-called local-RIE; c.f. [232, §5.B]. With linearized simulation error, (2.118) becomes the supremum of a quadratic function in x_t, which can be computed analytically.

2.5 Concluding remarks

2.5.1 Other approaches

We conclude our survey of 'optimization-based' approaches to system identification by discussing some methods that don't fit quite so neatly into the framework we have outlined, or at the very least, offer a conceptually different treatment of the parameter estimation problem. Specifically, the optimization approach we have outlined seeks a point estimate of the model parameters, i.e. $\hat{\theta} = \arg\min_{\theta \in \Theta} \mathcal{E}(\theta)$. In this section, we briefly review methods in which the goal is not a point estimate of model parameters.

Bayesian methods

In a *Bayesian* approach to system identification, the model parameters θ are explicitly modeled as a random variable. The identification problem then becomes estimation of probability density function (PDF) that governs the distribution of the parameters, rather than estimation of specific numerical values for the parameters. Specifically, given some probabilistic model of the system, e.g. (2.13), we seek the *posterior* distribution of the model parameters after the observations $y_{1:T}$ have been made, i.e., $p(\theta|y_{1:T})$. The posterior distribution is given by Bayes' rule,

$$p(\theta|y_{1:T}) = \frac{p(y_{1:T}|\theta)p(\theta)}{p(y_{1:T})}, \tag{2.119}$$

where $p(y_{1:T}|\theta)p(\theta)$ is the likelihood, commonly denoted $p_\theta(y_{1:T})$ and discussed in Section 2.1.2, and $p(\theta)$ is the *prior* distribution on θ, which encodes our beliefs about the values of θ before (i.e., prior to) the observations $y_{1:T}$. Roughly speaking, Bayes' rule can be thought of

as the formula for quantitatively updating our beliefs about an uncertain quantity in light of new information. Refer to [176] for comprehensive treatment of the Bayesian approach to system identification.

Despite the simplicity of (2.119), application of Bayes' rule to identification of general nonlinear and non-Gaussian dynamical systems can be computationally challenging. Foremost, Bayes' rule clearly requires the likelihood, $p(y_{1:T}|\theta)$, which can only be computed exactly in special cases, e.g. for linear Gaussian models via the Kalman filter. In general, approximations such as those based on sequential Monte Carlo (SMC, a.k.a. particle filtering) , a.k.a. particle filtering, must be used, c.f., e.g., [8, 122]. Secondly, application of Bayes' rule also requires the density of the observations, $p(y_{1:T})$, also known as the marginal likelihood. The marginal likelihood can be expressed as

$$p(y_{1:T}) = \int p(y_{1:T}|\theta)p(\theta)d\theta \tag{2.120}$$

which, once more, emphasizes the importance of the likelihood. However, even when the likelihood is known, one must still evaluate this (possibly high-dimensional) integral w.r.t. θ, which may not be analytically tractable. Again, SMC methods provides a general framework for approximating such integrals; c.f. [204] for a very accessible introduction to SMC in system identification, as well as [101, 122, 164, 203].

Before leaving the Bayesian framework, we note that it is not uncommon to generate a point estimate of the model parameters from the posterior distribution $p(\theta|y_{1:T})$. One popular strategy is to compute the so-called *maximum a posteriori (MAP) estimate*, given by

$$\hat{\theta}^{\text{MAP}} := \arg\max_{\theta} p(\theta|y_{1:T}) = \arg\max_{\theta} p_\theta(y_{1:T})p(\theta). \tag{2.121}$$

We wish to make two remarks. First, computing the MAP estimate clearly fits into the optimization-based framework outlined in this chapter; in this interpretation, the quality-of-fit metric is given by the posterior distribution $p(\theta|y_{1:T})$. Second, (2.121) illustrates the close connections between the Bayesian framework and regularization. In particular, maximizing the logarithm of the posterior in (2.121), leads to

$$\max_{\theta} = \log\left(p_\theta(y_{1:T})p(\theta)\right) = \max_{\theta} \log p_\theta(y_{1:T}) + \log p(\theta). \tag{2.122}$$

With a Gaussian prior, i.e. $p(\theta) = \mathcal{N}\left(\bar{\theta}, \Sigma\right)$, the MAP estimate can be interpreted as maximum likelihood identification with Tikhonov regularization, i.e., (2.122) is equivalent to

$$\min_{\theta} \left\{ -\log p_\theta(y_{1:T}) + \frac{1}{2}|\theta - \bar{\theta}|^2_{\Sigma^{-1}} \right\}.$$

Asymptotic analysis of frequentist approaches

Though interpretation of $\hat{\theta}$ as a random variable is most explicit in the Bayesian framework, such a treatment is also useful for asymptotic analysis of frequentist approaches, such as prediction error methods, i.e., the behavior of the estimate $\hat{\theta}_T^{\mathrm{PE}}$, c.f. (2.21), as the number of data points $T \to \infty$. In Section 2.1.2, it was noted that $\hat{\theta}_T^{\mathrm{PE}} \to \theta^*$ as $T \to \infty$, where[4] $\theta^* = \arg\min_{\theta\in\Theta} \bar{\mathrm{E}}\left[\ell(\epsilon_t^{\mathrm{p}}(\theta))\right]$, c.f. (2.22). Under mild assumptions, it can be shown that the asymptotic distribution of the random variable $\sqrt{T}(\hat{\theta}_T^{\mathrm{PE}} - \theta^*)$ is zero-mean Gaussian, c.f., [129, §9] for details and explicit expressions for the covariance. Even though the outcome of parameter estimation for a given realization of the problem data is a point estimate, this result emphasizes that the identification process is in fact a mapping from the random variables $\tilde{u}_{1:T}$ and $\tilde{y}_{1:T}$ to the random variable $\hat{\theta}$, e.g., $\hat{\theta}_T^{\mathrm{PE}}$ in the case of PEM. For each realization of the problem data, we can expect different numerical values of the parameter estimates, which obey a normal distribution as $T \to \infty$.

Set membership methods

Model-based design of automatic controllers is one of the most important application areas for system identification. The objective of *robust control* is to design a controller that meets certain performance specifications despite uncertainty in the system model; i.e., the input to the design process is a *set* of models, and the output is a controller guaranteed to meet the performance specifications for all models in the set, c.f., e.g., [209, 269]. The requirement of robust control for both a nominal system model as well as an explicit worst-case (deterministic) bound on system uncertainty inspired the development of the 'identification for control' framework, which includes *set membership* methods, c.f., e.g., [33, 37, 38, 89, 153]. Set membership (SM) methods are concerned with the identification of a set of unfalsified models, i.e., models that are consistent with the observed data, as well as *a priori* assumptions on the model structure and disturbances.

There are (at least) two main considerations in the design of a SM method (and identification for control methods more broadly). First, the identification method must reduce system uncertainty to a level for which a robust controller exists; if the uncertainty is too great (i.e. the model set is too large) then it may not be possible to design a controller satisfying the performance specifications for all models in the set. Second, the model set must be in a form compatible with robust control design methods. This typically requires an explicit bound on system uncertainty, to quantify the worst-case error between the nominal model and true system description. Satisfying this latter requirement is simpler for LTI systems, which have received the most attention in the SM literature. Nonetheless, extensions to linear systems with static nonlinearities (i.e., Hammerstein and Wiener systems) have been developed, c.f., [34, 225].

[4] In the case that $\bar{\mathrm{E}}\left[\ell(\epsilon_t^{\mathrm{p}}(\theta))\right]$ has no unique minimizer, θ^* belongs to the set of minimizers, and $\hat{\theta}_T^{\mathrm{PE}} \to \theta^*$ denotes convergence to a set.

Nonparametric models

In this thesis, we will be concerned with identification of finitely parametrized models. However, there is an extensive literature on *nonparametric* models; such methods have received increased attention recently, in part due to the flurry of activity in machine learning approaches to statistical inference (a.k.a. statistical machine learning) [56]. In this nonparametric setting, which is especially common in the Bayesian framework, Gaussian processes (GPs) [193] have proved to be very useful for performing inference directly over the space of functions. Modeling p_θ^a and/or p_θ^g in (2.13) with a GP leads to a so-called Gaussian process state space model (GP-SSM), c.f., e.g., [65, 66], as well as [224] for a generalization of the GP-SSM that allows for discontinuities. GPs have also found application in impulse response modeling [42, 177], and NARX [109].

2.5.2 Summary

This chapter has presented an 'optimization-based approach' to system identification, as a way of systematically reviewing the immense literature. The three key elements to such an approach are: i. a model set Θ, ii. a quality-of-fit criterion $\mathcal{E}$, and iii. an algorithm to solve $\min_{\theta\in\Theta}\ \mathcal{E}(\theta)$. As in any optimization task, convexity plays a central role. In system identification, nonconvexity enters through the constraints that ensure useful behavioral properties of the identified model (chiefly, model stability), and through quality-of-fit criteria that capture the long-term predictive power of the model, such as simulation error.

In Section 2.3 and 2.4 we reviewed a recently developed family of methods for generating convex parametrizations of stable models and convex bounds on simulation error, respectively. The remaining chapters of this thesis extend this line of research in several important directions. In Chapter 3, specialized interior point algorithms for Lagrangian relaxation are developed; these algorithms substantially reduce computational complexity compared to generic solvers. In Chapter 4, these ideas are translated to a stochastic setting, where maximum likelihood identification subject to stability constraints is addressed. Finally, in Chapter 5 the special case of identification of positive systems is considered. In this setting, many of these convex constructions are considerably simplified, allowing identification of very large-scale systems via distributed optimization.

Chapter 3

Specialized algorithms for Lagrangian relaxation

When accurate open-loop predictions of long-term system behavior are required, it is appropriate to fit a model by minimization of simulation error (closely related to output error). Unfortunately, for models containing feedback, e.g. state space models, simulation error is (in general) a nonlinear and nonconvex function of the model parameters, which makes global optimization challenging. In Section 2.4, we reviewed a recently developed family of methods for deriving convex approximations to simulation error. Among these methods was *Lagrangian relaxation*, which can be used to generate a convex upper bound for simulation error. Though convex, minimization of these bounds by general-purpose semidefinite program (SDP) solvers suffers from poor scalability. Specifically, computation time grows as a *cubic* function of the number of data points used for training.

This chapter presents specialized algorithms for minimization of Lagrangian relaxation of simulation error, over convex parametrizations of stable models, that have lower computational complexity compared to general-purpose SDP solvers. Specifically, we derive custom path-following interior point algorithms for which computation time grows as a *linear* function of the number of data points used for training. The basic idea is to exploit structural properties of the Lagrangian relaxation so as to: (i) dramatically simplify computation of the gradient and Hessian of the quality-of-fit criterion; (ii) significantly reduce the size of the linear matrix inequalities (LMI) that must be enforced, compared to the standard-form SDP representation of the problem.

The primary contribution of this chapter is the derivation and complexity analysis of these specialized algorithms. Equipped with these efficient solvers, a secondary contribution of this chapter is to empirically evaluate the performance of the Lagrangian relaxation method, compared to established methods for system identification, such as nonlinear ARX. In particular, we explore the apparent *regularizing* effect of Lagrangian relaxation with model stability constraints, which seems to eliminate the need for careful selection, or pruning, of

regressors in nonlinear model structures.

Connection to other chapters

The specialized algorithms developed in this chapter form the basic computational machinery necessary for efficient application of Lagrangian relaxation in other contexts, which includes many of the subsequent developments in this thesis. For example, in Chapter 4 we use Lagrangian relaxation to incorporate model stability constraints into the maximum likelihood framework. In the nonlinear setting, one of our proposals involves a Monte Carlo approximation of the likelihood which leads to Lagrangian relaxation of many simultaneous simulation error minimization problems at each iteration of an Expectation Maximization algorithm. Without the efficient custom interior point methods developed in this chapter, such an approach would be computationally prohibitive.

Publications

This material presented in this chapter also appears in the following publications:

> **J. Umenberger**, I.R. Manchester. Specialized algorithm for identification of stable linear systems using Lagrangian relaxation. In *Proceedings of the American Control Conference (ACC)*. 2016.
>
> **J. Umenberger**, I.R. Manchester. Specialized interior-point algorithm for stable nonlinear system identification. *IEEE Transactions on Automatic Control*. 2017. *Under review.*

3.1 Introduction

In this chapter, we consider identification of (linear or nonlinear) state-space models of the form

$$x_{t+1} = a(x_t, u_t), \quad y_t = g(x_t, u_t). \tag{3.1}$$

where x_t is an internal state, and u_t, y_t are input and output, respectively. This model class is very flexible and includes nonlinear autoregressive models [24, 208], infinite-impulse-response linear systems [129], Hammerstein and Wiener models [23], and recurrent neural networks [116].

The downside of increased flexibility is a substantial increase in the difficulty of the search for a model. Major problems include the difficulty of ensuring that the identified model

(3.1) is stable, and existence of local minima due to non-convexity of long-term simulation error (a.k.a. output error) as a function of model parameters [131, 169, 208].

Existing approaches to identification of state-space models include subspace identification for linear systems [241], the prediction error method [129], initializing the search for nonlinear models with frequency-domain fitting of linear models [169], maximum-likelihood via the expectation-maximization algorithm [203, 237], and Bayesian identification via Markov-chain Monte Carlo [164, 224]. None of these methods guarantee globally optimal fits or stability of the identified model, though for linear subspace identification a number of methods have been proposed (e.g. [113, 138, 238]).

The work in this chapter builds upon [232], which proposed a convex parametrization of nonlinear state-space models with guaranteed stability, as well as a family of convex upper bounds on simulation error. This line of research was initiated in [150], and further developed in [25, 141, 229, 230]. A central contribution of [232] is construction of a simulation-error bound based on a version of the Lagrangian relaxation (LR) [119], closely related to the S-procedure [181]. This bound can be represented as a semidefinite program (SDP), however for practical data-sets the resulting SDP is very large, and the experimental results in [232] were all based on the simpler but less-accurate *robust identification error* (RIE), first presented in [230]. Indeed, we will show that when represented as a standard semidefinite program, the LR approach has computational complexity which is *cubic* in the length of the data set, severely limiting its practical utility.

Our main contribution is a specialized algorithm that takes advantage of the structure in the LR optimization to significantly improve computational tractability: scaling of Newton iterates with respect to data-set length is now *linear* instead of cubic. Our contribution can therefore be seen in the context of a growing body of research on specialized solvers for special classes of SDPs appearing in robustness analysis via integral quadratic constraints and the Kalman-Yakubovich-Popov lemma, e.g. [103, 104, 243, 250]. Of more direct relevance to system identification is [125], which develops a custom interior point method (exploiting structure in the Nesterov-Todd equations) for nuclear norm approximation with application to subspace identification.

A secondary contribution of this chapter, enabled by the development of our specialized algorithm, is to empirically evaluate the performance of the LR method and compare it to established methods of linear and nonlinear system identification. In particular, we explore the apparent *regularizing* effect of the stability constraint and LR.

Regularization refers to the process of constraining or reducing model complexity (in some sense) to prevent over-fitting and to manage the bias-variance trade-off in statistical modeling [86]. Classical methods such as ridge regression (shrinkage) and subset selection (regressor pruning) have long been applied in nonlinear system identification [24, 99, 208]. More recently, novel regularization strategies have been developed for identification of linear systems, including nuclear norm regularization for subspace identification (e.g. [125]) and kernel methods for impulse-response modeling, surveyed in [179]. In this chapter, we

provide evidence that stability constraints and LR have an effective regularizing effect and seem to eliminate the need for regressor pruning.

This chapter is structured as follows: Section 3.2 introduces notation and problem setup. Section 3.3 contains the main contribution: the specialized algorithm. Section 3.4 demonstrates the algorithm's improved computational complexity over existing methods. Sections 3.5, 3.6, and 3.7 present empirical comparisons to established methods on a number of example problems, and finally Section 3.8 offers concluding remarks.

3.2 Preliminaries

3.2.1 Notation

Specific notation used in this chapter is as follows. The cone of real, symmetric nonnegative (positive) definite matrices is denoted by $\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$). The $n \times n$ identity matrix is denoted I_n. Let $\text{vec} : \mathbb{R}^{m\times n} \mapsto \mathbb{R}^{mn}$ denote the function that stacks the columns of a matrix to produce a column vector, and $\text{mat} : \mathbb{R}^{mn} \mapsto \mathbb{R}^{m\times n}$ for its inverse. Let $\text{s2v} : \mathbb{S}^n \mapsto \mathbb{R}^{n(n+1)/2}$ denote the functions that stacks the columns of an $n \times n$ symmetric matrix, with duplicate entries omitted. The Kronecker product is denoted $\otimes$. The transpose of a matrix a is denoted a', and $|a|^2_Q$ is shorthand for $a'Qa$. For brevity, we use $\|a\|$ to denote the Euclidean norm $\|a\|_2$ of $a \in \mathbb{R}^n$. For a polynomial p, $p \in \text{SOS}$ denotes membership in the cone of sum-of-squares polynomials [172].

3.2.2 Model class

This chapter concerns the identification of nonlinear discrete-time state-space models of the *implicit* form

$$e(x_{t+1}) = f(x_t, u_t) \tag{3.2a}$$

$$y_t = g(x_t, u_t) \tag{3.2b}$$

where $e : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x}$, $f : \mathbb{R}^{n_x\times n_u} \mapsto \mathbb{R}^{n_x}$ and $g : \mathbb{R}^{n_x\times n_u} \mapsto \mathbb{R}^{n_y}$ are multivariate polynomials or trigonometric polynomials, *linearly parametrized* by unknown model parameters $\rho \in \mathbb{R}^{n_\rho}$. We shall enforce that $e(\cdot)$ be a bijection; i.e. for any $b \in \mathbb{R}^{n_x}$ there exists a unique solution $s \in \mathbb{R}^{n_x}$ to $e(s) = b$. This means we can recover a model of the form (3.1) by computing $x_{t+1} = e^{-1}(f(x_t, u_t)) = a(x_t, u_t)$. Implicit models of this form improve the quality of Lagrangian relaxation due to redundancy in the constraints, and also permit convex parameterizations of stable models, [232].

The following strong form of stability ensures sensible model behavior for inputs not present in the training dataset:

Definition 3.1 (Global incremental ℓ^2 stability)**.** The model (3.1) is said to be stable if the sequences $\{\bar{y}_t - \hat{y}_t\}_{t=1}^{\infty}$ and $\{\bar{x}_t - \hat{x}_t\}_{t=1}^{\infty}$ are square summable for every two solutions $(\bar{u}, \bar{x}, \bar{y})$ and $(\hat{u}, \hat{x}, \hat{y})$ of (3.1), subject to the same input $\bar{u} = \hat{u}$.

3.2.3 Simulation error

Given measurements of inputs $\{\tilde{u}_t\}_{t=1}^{T}$ to and outputs $\{\tilde{y}_t\}_{t=1}^{T}$ from some dynamical system (not necessarily in the model class), our goal is to minimize the *simulation error*:

$$\mathcal{E}^{\text{se}} := \sum_{t=1}^{T} \|\tilde{y}_t - g(x_t, \tilde{u}_t)\|^2 \tag{3.3}$$

where x_t represents the *simulated state*, given by

$$x_t = e^{-1}\left(f(\ldots e^{-1}(f(e^{-1}(f(\tilde{x}_1, \tilde{u}_1)), \tilde{u}_2))\ldots, \tilde{u}_{t-1})\right),$$

i.e. the solution of (3.2), subject to the input $\{\tilde{u}_t\}_{t=1}^{T}$ and initial condition $x_1 = \tilde{x}_1$. Dependence on the simulated state renders $\mathcal{E}^{\text{se}}$ a highly nonlinear function of the model parameters.

3.2.4 Lagrangian relaxation of linearized simulation error

Rather than minimize $\mathcal{E}^{\text{se}}$ directly, the approach proposed in [232] is to approximate $\mathcal{E}^{\text{se}}$ via *Lagrangian relaxation* of the *linearized simulation error*, which we now define.

Given an estimated state sequence $\{\tilde{x}_t\}_{t=1}^{T}$, we define the *equation errors*

$$\epsilon_t = f(\tilde{x}_t, \tilde{u}_t) - e(\tilde{x}_{t+1}),\ \eta_t = g(\tilde{x}_t, \tilde{u}_t) - \tilde{y}_t \tag{3.4}$$

and Jacobians $E_t = \nabla_x\, e\,|_{x=\tilde{x}_t}$, $F_t = \nabla_x\, f\,|_{x=\tilde{x}_t}^{u=\tilde{u}_t}$, $G_t = \nabla_x\, g\,|_{x=\tilde{x}_t}^{u=\tilde{u}_t}$. The linearized simulation error is then given by

$$\mathcal{E}^0 = \sum_{t=1}^{T} \|G_t\Delta_t + \eta_t\|^2 \tag{3.5}$$

where Δ_t satisfies $\Delta_1 = 0$ and $E_{t+1}\Delta_{t+1} = F_t\Delta_t + \epsilon_t$ for $t = 1, \ldots, T-1$. The linearized simulation error $\mathcal{E}^0$ quantifies local (i.e. close to $\{\tilde{x}_t\}_{t=1}^{T}$) sensitivity of the model equations to equation errors; c.f., [232, §V] for further details.

In this work, Lagrangian relaxation refers to the approximation of the nonconvex problem $\min_\rho \mathcal{E}^0$ by the convex problem $\min_\rho \hat{J}_\lambda(\rho)$, where for $\Delta = (\Delta_1', \ldots, \Delta_T')'$

$$\hat{J}_\lambda(\rho) = \sup_{\Delta} \left\{ \sum_{t=1}^{T} \|G_t\Delta_t + \eta_t\|^2 - \lambda_1' E_1\Delta_1 - \sum_{t=1}^{T-1} \lambda_{t+1}'(E_{t+1}\Delta_{t+1} - F_t\Delta_t - \epsilon_t) \right\}. \tag{3.6}$$

$\hat{J}_\lambda(\rho)$ represents a convex upper bound on $\mathcal{E}^0$ for arbitrary multipliers λ_t. In this chapter, we will use $\lambda_t = 2\Delta_t$, due to its desirable properties, e.g. tightness of the bound under ideal circumstances [232, Theorem 6]. Note that for linear dynamical systems, simulation error and linearized simulation are equivalent; we discuss the ways in which our approach is simplified for the special case of linear identification in Section 3.3.7.

The above construction depends on a *surrogate* state sequence $\{\tilde{x}_t\}_{t=1}^T$. While it is not assumed that these are *true* internal states, the more accurate they are the more effective our approach will be. Methods for generating state estimates from input-output data include subspace methods for linear systems [241]. For nonlinear systems, state estimation is more challenging and solutions can be quite case specific. Possible strategies include: subspace methods in the case of weakly nonlinear systems, c.f. Section 3.6; exploiting physical or structural knowledge, c.f. Section 3.5; alternating between model-based state estimation and model refinement, e.g. Expectation Maximization [237]; and using truncated histories of inputs and outputs (as in NARX) [208].

For what follows, it is convenient to introduce the following 'lifted' representation of (3.6). Let $\mathcal{G}(\rho) = \text{blkdiag}(G_1, \ldots, G_T)$, $\eta(\rho) = [\eta_1', \ldots, \eta_T']'$, $\epsilon(\rho) = \left[0, \epsilon_1', \ldots, \epsilon_{T-1}'\right]'$ and

$$\mathcal{F}(\rho) = \begin{bmatrix} E_1 & 0 & 0 & \ldots \\ -F_1 & E_2 & 0 & \ddots \\ 0 & -F_2 & E_3 & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}. \tag{3.7}$$

The upper bound in (3.6) can then be more compactly expressed as $\hat{J}_\lambda(\rho) = \sup_\Delta J_\lambda(\rho, \Delta)$, where

$$J_\lambda(\rho, \Delta) = \|\mathcal{G}(\rho)\Delta + \eta(\rho)\|^2 - 2\Delta'(\mathcal{F}(\rho)\Delta - \epsilon(\rho)). \tag{3.8}$$

3.2.5 Optimization with general-purpose solvers

Minimization of $\hat{J}_\lambda(\rho)$ can be formulated as the following SDP, compatible with any general-purpose SDP solver:

$$\min_\rho \quad s \tag{3.9a}$$

$$\text{s.t.} \quad \begin{bmatrix} s & \epsilon(\rho)' & \eta(\rho)' \\ \epsilon(\rho) & \mathcal{F}(\rho) + \mathcal{F}(\rho)' & \mathcal{G}(\rho)' \\ \eta(\rho) & \mathcal{G}(\rho) & I_{Tn_y} \end{bmatrix} \succeq 0 \tag{3.9b}$$

where s is a slack variable. If no structural properties (e.g. sparsity) of (3.9b) are exploited by the solver, then each iteration of a primal-dual interior point method requires

$$O\left(\max\{n_\rho(n_x + n_y)^3 T^3,\ n_\rho^2(n_x + n_y)^2 T^2\}\right)$$

operations to solve, c.f., e.g., [125, §2], where n_ρ is the number of free model parameters, and T is the number of data points in the training set. In a typical system identification scenario, the model and parameter dimensions n_x, n_y, and n_ρ remain moderate in size while the data set length T may be very large. This implies $O(T^3)$ complexity, which will be demonstrated empirically in Section 3.4.1.

3.3 Specialized algorithm

In this section we present the main contribution of this chapter: an efficient, scalable algorithm for the problem $\min_\rho \hat{J}_\lambda(\rho)$ where $\hat{J}_\lambda(\rho)$ (c.f. (3.6)) is a convex upper bound on linearized simulation error. Specifically, we present an interior-point algorithm for which the complexity of each Newton iteration grows *linearly* with the number of data points, T. See Algorithm 3 for a complete listing.

3.3.1 Stable model set and cost function

We begin by defining the set of stable models that we wish to search over, first introduced in [230]. Let $\mathcal{P}$ denote the set of all models ρ of the form (3.2) for which $\exists\ P \in \mathbb{S}^{n_x}_{++}$ and $\mu > 0$ such that the matrix inequality

$$\begin{aligned} m(\rho, P, x, u) :=& F(x,u)P^{-1}F(x,u) - E(x) - E(x)' + \\ & P + \mu I_{n_x} + G(x,u)'G(x,u) \preceq 0 \end{aligned} \tag{3.10}$$

holds for all $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$ where $E(x) = \nabla_x e(x)$, $F(x,u) = \nabla_x f(x,u)$ and $G(x,u) = \nabla_x g(x,u)$. The inequality (3.10) may be interpreted as a contraction condition [135] with the metric $E(x)'P^{-1}E(x)$, and guarantees global incremental ℓ^2 stability of the model ρ. The set $\mathcal{P}$ is nonconvex, and so we introduce a SOS approximation in Section 3.3.2.

The cost function that we seek to minimize is, of course, $\hat{J}_\lambda(\rho)$, given by (3.6). As the Lagrangian $J_\lambda(\rho, \Delta)$, defined in (3.8), is quadratic in Δ it is finite if and only if

$$W := \mathcal{G}'\mathcal{G} - \mathcal{F} - \mathcal{F}' \leq 0, \tag{3.11}$$

i.e., $J_\lambda(\rho)$ is concave. Imposing strict negativite-definiteness ensures robustness and a unique maximizing Δ. It turns out that $\rho \in \mathcal{P}$ is sufficient to guarantee $W < 0$. Specifically, we have the following result from [232, Theorem 6]:

Lemma 3.1. Given any arbitrary sequence of vectors $\{\tilde{x}_t\}_{t=1}^T$, $W := \mathcal{G}'\mathcal{G} - \mathcal{F} - \mathcal{F}' < 0$ for $\rho \in \mathcal{P}$, and thus $\hat{J}_\lambda(\rho)$ is finite.

The key point is that we can guarantee $W < 0$, i.e. a large LMI that grows linearly in dimension with T, by enforcing (3.10), a low-dimensional convex constraint that does not grow with T. This property is essential to the scalability of our approach.

3.3.2 Convex parametrization of stable models

In this section, we detail how the set of stable models,

$$\mathcal{P} = \{\rho \in \mathbb{R}^{n_\rho} : \exists P \in \mathbb{S}^{n_x}_{++} \text{ s.t. } m(\rho, P, x, u) \preceq 0 \ \forall\ x, u\}$$

can be approximated with sum-of-squares (SOS) programming [172]. By an application of the Schur complement, (3.10) is equivalent to the infinite family of LMIs:

$$M(\rho, P, x, u) = \tag{3.12}$$
$$\begin{bmatrix} E(x) + E(x)' - P - \mu I_{n_x} & F(x,u)' & G(x,u)' \\ F(x,u) & P & 0 \\ G(x,u) & 0 & I_{n_y} \end{bmatrix} \succeq 0.$$

Introducing $v \in \mathbb{R}^{2n_x+n_y}$, and $z = [x', u', v']'$, we define the linearly parametrized scalar polynomial

$$p(z) := v' M(\rho, P, x, u) v.$$

Then the condition $M(\rho, P, x, u) \succeq 0\ \forall x, u$ is equivalent to $p(z) \geq 0\ \forall z$. Testing nonnegativity of a general multivariate polynomial is known to be NP-hard. However, constraining $p(z)$ to be SOS gives tractable *sufficient conditions* for nonnegativity [172].

Our goal is to find a representation of $p(z)$ of the form

$$p(z) = \omega(z)' Q \omega(z) =: \sum_{i=1}^{n_q} c_i(Q) z^{\beta_i}, \text{ where } Q \in \mathbb{S}^{n_\omega}_{+}, \tag{3.13}$$

and $\omega : \mathbb{R}^{n_z} \mapsto \mathbb{R}^{n_\omega}$ is a vector of n_ω monomials and Q is the Gram matrix. Careful selection of the basis monomials ω can simplify the SOS program, e.g., reduce the number of constraints and decision variables. Tools such as the Newton polytope [134], and facial reduction [175], can be used to generate an effective basis. For the examples in this chapter, we used the toolbox [233] for monomial selection.

Introducing $\theta = [\rho', \mathrm{s2v}(P)', \mathrm{s2v}(Q)']'$, the equality constraints in (3.13) can be expressed as $A_e\theta = b_e$. Similarly, positive semidefiniteness of the Gram matrix can be encoded as $S(\theta) := \mathrm{mat}(A_s\theta) := Q \succeq 0$. Then $p(z) \in \mathrm{SOS}$ is equivalent to $\theta \in \Theta$, where

$$\Theta = \{\theta : S(\theta) = \mathrm{mat}\,(A_s\theta) \succeq 0,\ A_e\theta = b_e\}. \tag{3.14}$$

To develop a primal-only interior point method, c.f. Section 3.3.3, we first eliminate the equality constraints in Θ, by constructing a general solution to $A_e\theta = b_e$,

$$\theta(\nu) = \theta^* + N_e\nu \tag{3.15}$$

where $A_e\theta^* = b_e$, N_e is a basis for the nullspace of A_e, and ν denotes our new decision

variables. The particular solution θ^* can be obtained, e.g., from the semidefinite feasibility problem:

$$S(\theta) = \mathrm{mat}\,(A_s\theta) \succeq 0,\ \ A_e\theta = b_e. \tag{3.16}$$

With the parameterization (3.15) the model set Θ reduces to $\{\nu : S(\nu) \succeq 0\}$, and our optimization problem becomes

$$\min_{\nu}\ \hat{J}_\lambda(\nu) \text{ s.t. } S(\nu) \succeq 0. \tag{3.17}$$

Here, with mild abuse of notation, we have used $\hat{J}_\lambda(\nu)$ as shorthand for $\hat{J}_\lambda(\theta(\nu))$, which is understood to denote $\hat{J}_\lambda(\rho(\nu))$. Similarly, $S(\nu)$ is shorthand for $S(\theta(\nu))$.

Before moving on, we remark that $A_e\theta = b_e$ in (3.14) will typically contain many constraints of the form $Q(i,j) = [\rho', \mathrm{s2v}(P)']c_{ij}$, where c_{ij} is a vector of constants. For such constraints, it is possible to eliminate the variable $Q(i,j)$ by *directly* parametrizing the Gram matrix. Although the resulting formulation is mathematically equivalent, we have observed that direct parametrization of the Gram matrix can improve numerical conditioning in practice.

3.3.3 Path-following interior point method

The algorithm we propose solves (3.17) via a (primal-only) path following interior point, or *barrier*, method; see, e.g., [162]. Primal-dual interior point methods are generally expected to be more efficient than barrier methods on standard SDPs [242]. Despite this, we employ a primal-only method for the following reasons:

i. The LR approach requires minimization of a smooth *nonlinear* function of the semidefinite cone. Unlike standard SDPs, the dual function does not have a simple explicit representation. Lifting to a standard-form SDP involves introducing a large number of additional variables (see Section 3.2.5).

ii. We can exploit structural properties of $\hat{J}_\lambda(\nu)$ to dramatically simplify the computation of the gradient and Hessian.

iii. We can exploit the fact that $S(\nu) \succeq 0$ guarantees $W < 0$ and finiteness of $\hat{J}_\lambda(\nu)$, c.f. Lemma 3.1, which leads to a much smaller LMI than the formulation outlined in Section 3.2.5.

iv. Using a primal-only interior point method, we can guarantee model stability at every iteration. In standard primal-dual methods, the iterates are not necessarily feasible. This permits early stopping (without compromising model stability), a well-known regularization method that has long been used in system identification, c.f., e.g., [208].

v. As we will see in Section 3.4.2, our proposed algorithm is more accurate (i.e. returns models with lower cost $\hat{J}_\lambda$) than primal-dual methods, due to the numerical problems encountered by general-purpose solvers for large datasets.

The key idea in a path-following interior point method is the introduction a barrier function that tends towards infinity at the boundary of the feasible set. We use the standard choice for the LMI constraint $S(\nu) \succeq 0$ [162], i.e.,

$$\phi(\nu) = \begin{cases} -\log\det S(\nu) & S(\nu) > 0 \\ \infty & S(\nu) \not\succ 0 \end{cases}.$$

The barrier function, weighted by a scalar τ, is then added to the objective $\hat{J}_\lambda(\nu)$ and we solve (using a damped Newton method) a sequence of *unconstrained* optimization problems

$$\min_\nu \{f_\tau(\nu) := \hat{J}_\lambda(\nu) + \tau\phi(\nu)\} \tag{3.18}$$

for decreasing τ, exploiting the insights enumerated above.

3.3.4 Gradient computation

In this section, we show that the gradient $\nabla\hat{J}_\lambda(\nu)$ can be computed very efficiently, which is one of the key properties exploited by our solver. The gradient of $f_\tau(\nu)$ is given by

$$\nabla f_\tau(\nu) = \nabla\hat{J}_\lambda(\nu) + \tau\nabla\phi(\nu). \tag{3.19}$$

We consider each term separately, beginning with $\nabla\hat{J}_\lambda(\nu)$. Recalling our parametrization of $\theta(\nu)$ in (3.15), we have

$$\frac{\partial\hat{J}_\lambda(\theta(\nu))}{\partial\nu} = \frac{\partial\hat{J}_\lambda}{\partial\theta}\frac{\partial\theta}{\partial\nu} = \frac{\partial\hat{J}_\lambda}{\partial\theta}N_e \tag{3.20}$$

by the chain rule. We now present a simple formula for the gradient of $\hat{J}_\lambda(\theta)$ w.r.t. θ at a particular parameter $\theta^\dagger \in \Theta$. From Lemma 3.1, the bound is given by $\hat{J}_\lambda(\theta) = \max_\Delta J_\lambda(\theta,\Delta) = J_\lambda(\theta,\Delta^*(\theta))$ where

$$\Delta^*(\theta) = \arg\max_\Delta \; J_\lambda(\theta,\Delta) = -W^{-1}(\mathcal{G}'\eta + \epsilon). \tag{3.21}$$

The gradient is then given by

$$\frac{\partial\hat{J}_\lambda}{\partial\theta} = \frac{\partial J_\lambda}{\partial\theta} + \frac{\partial J_\lambda}{\partial\Delta}\frac{\partial\Delta^*}{\partial\theta}.$$

As $\Delta^*(\theta^\dagger)$ maximizes the smooth function $J_\lambda(\theta^\dagger, \Delta)$, we have $\frac{\partial J_\lambda}{\partial \Delta} = 0$ at $\Delta = \Delta^*(\theta^\dagger)$, and so

$$\frac{\partial \hat{J}_\lambda}{\partial \theta} = \left.\frac{\partial J_\lambda}{\partial \theta}\right|_{\theta=\theta^\dagger, \Delta=\Delta^*(\theta^\dagger)}. \tag{3.22}$$

The key point is that $\frac{\partial \Delta^*}{\partial \theta}$ need not be computed to calculate the gradient of $\hat{J}_\lambda(\theta^\dagger)$, which now reduces to

$$\frac{\partial \hat{J}_\lambda}{\partial \theta(i)} = 2(\mathcal{G}\Delta^* + \eta)'(\mathcal{G}_i\Delta^* + \eta_i) - 2\Delta^{*\prime}(\mathcal{F}_i\Delta^* - \epsilon_i) \tag{3.23}$$

where $\mathcal{G}_i, \eta_i, \mathcal{F}_i, \epsilon_i$ denote $\partial\mathcal{G}/\partial\theta(i), \partial\eta/\partial\theta(i), \partial\mathcal{F}/\partial\theta(i), \partial\epsilon/\partial\theta(i)$, respectively.

We now turn our attention to $\nabla\phi(\nu)$. The chain rule gives

$$\frac{\partial \phi(\theta(\nu))}{\partial \nu} = \frac{\partial \phi(\theta)}{\partial \theta}\frac{\partial \theta}{\partial \nu} = \frac{\partial \phi(\theta)}{\partial \theta} N_e. \tag{3.24}$$

The gradient of $\phi(\theta)$ w.r.t. θ is straightforward to compute, as $S(\theta)$ is affine in θ; specifically, $S(\theta) = \mathrm{mat}(A_s\theta)$. Recall that for $g(Z) = \log\det Z$, where $Z \in \mathbb{S}_{++}$, we have $\nabla g = Z^{-1}$, and so by the chain rule we have

$$\frac{\partial \phi(\theta)}{\partial \theta} = \left[\frac{\partial \phi}{\partial \theta(1)}, \ldots, \frac{\partial \phi}{\partial \theta(n_\theta)}\right] = -\mathrm{vec}(S(\theta)^{-1})' A_s. \tag{3.25}$$

3.3.5 Hessian computation

The Hessian of $f_\tau(\nu)$ w.r.t ν is given by

$$\nabla^2 f_\tau(\nu) = \nabla^2 \hat{J}_\lambda(\nu) + \tau\nabla^2\phi(\nu). \tag{3.26}$$

By the chain rule we have

$$\frac{\partial^2 \hat{J}_\lambda(\theta(\nu))}{\partial \nu^2} = \frac{\partial \theta}{\partial \nu}'\frac{\partial^2 \hat{J}_\lambda(\theta)}{\partial \theta^2}\frac{\partial \theta}{\partial \nu} = N_e'\frac{\partial^2 \hat{J}_\lambda(\theta)}{\partial \theta^2} N_e \tag{3.27}$$

To compute the Hessian $\nabla^2\hat{J}_\lambda$ we differentiate (3.22), yielding

$$\frac{\partial^2 \hat{J}_\lambda}{\partial \theta^2} = \frac{\partial^2 J_\lambda}{\partial \theta^2} + \frac{\partial^2 J_\lambda}{\partial \Delta \partial \theta}\frac{\partial \Delta^*}{\partial \theta}. \tag{3.28}$$

Notice that $\frac{\partial^2 \Delta^*}{\partial \theta^2}$ does not appear in (3.28), for the same reason that $\frac{\partial \Delta^*}{\partial \theta}$ does not appear in $\nabla\hat{J}_\lambda$, namely: $\frac{\partial J_\lambda}{\partial \Delta}(\theta, \Delta^*(\theta)) = 0$ for all $\theta \in \Theta$. This property simplifies the computation of $\nabla^2\hat{J}_\lambda$. The first two quantities in (3.28) are given by

$$\frac{\partial^2 J_\lambda}{\partial \theta(j)\partial \theta(i)} = 2(\mathcal{G}_j\Delta^* + \eta_j)'(\mathcal{G}_i\Delta^* + \eta_i), \tag{3.29a}$$

$$\frac{\partial^2 J_\lambda}{\partial \Delta \partial \theta(i)} = \begin{array}{l} 2\Delta^{*\prime}\left(\mathcal{G}'\mathcal{G}_i + \mathcal{G}_i'\mathcal{G} - \mathcal{F}_i' - \mathcal{F}_i\right) + \\ 2\left(\eta'\mathcal{G}_i + \eta_i'\mathcal{G} + \epsilon_i'\right). \end{array} \tag{3.29b}$$

To compute $\frac{\partial \Delta^*}{\partial \theta}$ we rewrite the linear system (3.21) as

$$W\Delta^* = w \tag{3.30}$$

where $W = \mathcal{G}'\mathcal{G} - \mathcal{F}' - \mathcal{F}$ and $w = -\mathcal{G}'\eta - \epsilon$. Application of the product rule to (3.30) yields

$$W\frac{\partial \Delta^*}{\partial \theta(i)} = \left(\frac{\partial w}{\partial \theta(i)} - \frac{\partial W}{\partial \theta(i)}\Delta^*\right), \tag{3.31}$$

from which we can solve for $\frac{\partial \Delta^*}{\partial \theta(i)} \in \mathbb{R}^{Tn_x}$.

We now turn our attention to the Hessian of the barrier, $\nabla^2\phi(\nu)$, which, like the gradient, is straightforward to compute. By the chain rule, we have

$$\frac{\partial^2 \phi(\theta(\nu))}{\partial \nu^2} = \frac{\partial \theta}{\partial \nu}'\frac{\partial^2 \phi(\theta)}{\partial \theta^2}\frac{\partial \theta}{\partial \nu} = N_e'\frac{\partial^2 \phi(\theta)}{\partial \theta^2}N_e. \tag{3.32}$$

While $\nabla^2\phi(\theta)$ is easy to compute, it is somewhat cumbersome to express. Let $\mathcal{B} : \mathbb{S}^n \mapsto \mathbb{S}^{n^2}$ denote the function that maps a symmetric matrix $Z \in \mathbb{S}^n$ to the $n \times n$ block matrix, in which the $(i,j)^{\text{th}}$ block is given by $Z(:,j)Z(:,i)'$, where $Z(:,i)$ denotes the ith column of Z. Then, by the chain rule, the Hessian of the barrier function is given by

$$\nabla^2\phi = \begin{bmatrix} \frac{\partial^2\phi}{\partial\theta(1)^2} & \frac{\partial^2\phi}{\partial\theta(1)\partial\theta(2)} & \cdots \\ \vdots & & \ddots \end{bmatrix} = A_s'\mathcal{B}(S(\theta)^{-1})A_s. \tag{3.33}$$

3.3.6 Stopping criteria

For each τ, the 'Newton iterations' (L7-20) terminate when at least one of the following convergence criteria is satisfied: i) change in $f_\tau(\nu)$ is less than a prescribed tolerance, δ_f; ii) the maximum absolute value of an element of $\nabla f_\tau(\nu)$ is less than δ_g; iii) the step size αd_k is less than δ_f. The 'barrier iterations' (and thus, the algorithm) terminate when the change in $\hat{J}_\lambda(\nu)$ is less than a prescribed tolerance, δ_J. Recommended values for these parameters are summarized in Table 3.1.

3.3.7 Special case: Identification of LTI systems

We conclude this section by making explicit the ways in which our proposed algorithm is simplified when applied to the special case of LTI systems. Throughout this section, we use

Table 3.1 – Parameter values for Algorithm 3.

Parameter	Description	Value
τ_0	Initial barrier weight	10^4
β	Barrier weight division factor	10
δ_f	Newton objective tolerance	10^{-10}
δ_g	Newton gradient tolerance	10^{-10}
δ_J	Objective convergence tolerance	10^{-11}
`maxit`	Max no. of Newton iterations	10^4

Algorithm 3 MIN-LAGRANGIAN

Initialize $\theta_0 = \theta^*$, where θ^* is given by (3.16)
Initialize $\nu = 0$
Initialize τ_0, c.f. Table 3.1
while $|\hat{J}_\lambda(\nu_j) - \hat{J}_\lambda(\nu_{j-1})| > \delta_J$ **do**
    $\nu_k \leftarrow \nu_j$
    Set $f_\tau(\nu) = \hat{J}_\lambda(\nu) + \tau_j \phi(\nu)$
    for $k = 1 :$ `maxit` **do**
        Compute $\nabla \hat{J}_\lambda(\nu_k)$ using (3.20) and (3.23)
        Compute $\nabla \phi(\nu_k)$ using (3.24) and (3.25)
        Form $\nabla f_\tau(\nu_k)$ using (3.19)
        Compute $\nabla^2 \hat{J}_\lambda(\nu_k)$ using (3.27), (3.28), (3.29), (3.31)
        Compute $\nabla^2 \phi(\nu_k)$ using (3.32) and (3.33)
        Form $\nabla^2 f_\tau(\nu_k)$ using (3.26)
        Solve $d_k = \nabla^2 f_\tau(\nu_k)^{-1} \nabla f_\tau(\nu_k)$
        Compute the step length α by a backtracking line search to satisfy the Wolfe conditions.
        Update the parameter estimate: $\nu_{k+1} = \nu_k + \alpha d_k$
        if $|f_\tau(\nu_{k+1}) - f_\tau(\nu_k)| < \delta_f$ **or** $\|\nabla f_\tau(\nu_{k+1})\|_\infty < \delta_g$ **or** $\|\alpha d_k\|_\infty < \delta_f$ **then**
            $\nu_j \leftarrow \nu_k$ and **break**
        end if
    end for
    Set $\tau_{j+1} = \tau_j / \beta$ for some constant β
end while
return $\theta = \theta_0 + N_e \nu_j$

the specific implicit representation of LTI systems

$$Ex_{t+1} = Fx_t + Ku_t \tag{3.34a}$$
$$y_t = Cx_t + Du_t \tag{3.34b}$$

where $E \in \mathbb{R}^{n_x \times n_x}$, $F \in \mathbb{R}^{n_x \times n_x}$ and $K \in \mathbb{R}^{n_x \times n_u}$. There are two key simplifications in the linear case. First, linearized simulation error $\mathcal{E}^0$ and simulation error $\mathcal{E}^{\text{se}}$ are equivalent. To see this clearly, observe that for linear models $\nabla_x\, e(x) = E$, $\nabla_x\, f(x,u) = F$, $\nabla_x\, g(x,u) = C$, $\epsilon_t = F\tilde{x}_t + K\tilde{u}_t - E\tilde{x}_{t+1}$ and $\eta_t = C\tilde{x}_t + D\tilde{u}_t - \tilde{y}_t$. Substituting these identities into the definition of $\mathcal{E}^0$ with $\Delta_t = x_t - \tilde{x}_t$, c.f. Section 3.2.4, we obtain $\mathcal{E}^0 = \sum_{t=1}^{T} \|G_t\Delta_t + \eta_t\|^2 = \sum_{t=1}^{T} \|Cx_t + D\tilde{u}_t - \tilde{y}_t\|^2$ subject to the constraints $\Delta_t = 0 \iff x_t = \tilde{x}_t$ and $E_{t+1}\Delta_{t+1} = F_t\Delta_t + \epsilon_t \iff Ex_{t+1} = Fx_t + K\tilde{u}_t$, i.e., linearized simulation error $\mathcal{E}^0$ equals simulation error $\mathcal{E}^{\text{se}}$.

Second, there is no conservatism in the stability constraint: the stability condition (3.12) reduces to

$$M_l(\rho, P) = \begin{bmatrix} E + E' - P + \mu I & F' & C' \\ F & P & 0 \\ C & 0 & I \end{bmatrix} > 0. \tag{3.35}$$

As (3.35) represents a LMI, there is no need for SOS approximation, as in the nonlinear case. In fact, $\Theta_l := \{\rho, P : M_l(\rho, P) > 0\}$ defines a convex parametrization of *all* stable LTI systems, c.f. [141, Lemma 4], i.e., (3.35) is necessary and sufficient for stability.

3.4 Computational complexity

In this section we examine the computational complexity of the proposed algorithm with respect to the length of the data set T. We will show that the per-iteration cost of the proposed algorithm grows linearly with T, a significant improvement over the $O(T^3)$ per-iteration complexity of general-purpose SDP solvers, c.f. Section 3.2.5. This does not result in a *complete* complexity analysis, since we do not bound the *number* of iterations required. However, it is generally observed empirically that the number of iterations required grows very mildly with the number of variables [242], and we confirm this in the next subsection. In what follows, (Ln) refers to line n of Algorithm 3.

3.4.1 Scalability with respect to data length

In this subsection we establish that computational complexity of the gradient (L8) and Hessian (L11) of $\hat{J}_\lambda(\nu)$ both scale linearly with T. The gradient (L9) and Hessian (L12) of the barrier function $\phi(\nu)$, as well as the calculation of the search direction (L14), do not depend on T. Computation of the gradient $\nabla \hat{J}_\lambda$ requires:

- one application of the chain rule, as in (3.20) which does not grow with T,
- n_θ applications of formula (3.23). Notice, from (3.7), that $\mathcal{G}$ and $\mathcal{F}$, along with the derivatives $\mathcal{G}_i$ and $\mathcal{F}_i$, are sparse banded matrices. This implies that the products $\mathcal{G}\Delta^*$, $\mathcal{G}_i\Delta^*$ and $\Delta^{*'}\mathcal{F}_i\Delta^*$ in (3.23) can be computed with $O(T)$ arithmetic operations. As the model (3.2) is linearly parametrized, the gradients $\mathcal{G}_i, \eta_i, \mathcal{F}_i$ and ϵ_i can be precomputed off-line.
- The most expensive operation would appear to be the computation of Δ^* by solving the linear system (3.30). However, as W is block diagonal, Hermitian, and sign-definite we can employ the block Thomas algorithm [187, Section 3.8.3], to compute Δ^* with $O(T)$ operations.

To compute each of the $n_\theta(n_\theta+1)/2$ unique elements of $\nabla^2\hat{J}_\lambda$, we require:

- one application of (3.29a), requiring $O(T)$ operations due to block diagonality of $\mathcal{G}_i$, c.f. (3.7),
- one application of (3.29b), requiring $O(T)$ operations due to block diagonality of $\mathcal{G}$, $\mathcal{G}_i$ and $\mathcal{F}_i$, c.f. (3.7),
- the solution to (3.31) for $i = 1, \ldots, n_\theta$, requiring $O(T)$ operations as W is block tridiagonal, Hermitian and sign-definite.

To summarize, the complexity of computing each Newton step of the proposed algorithm is therefore $O(T)$.

Before moving on, we remark that computation of the Hessian is the most expensive part of each iteration. For identification of 'large scale' systems (e.g. models of high dimension n_x), it is possible to use only gradient information, if moderate-accuracy is acceptable, e.g., gradient descent or BFGS approximation of the Hessian, as in [235].

3.4.2 Empirical results

In this section we provide an empirical comparison between our proposed algorithm and general-purpose solvers. Both methods solve the same convex optimization problem,

$$\min_{\nu} \ \hat{J}_\lambda(\nu) \text{ s.t. } S(\nu) \succeq 0,$$

as in (3.17). All computations were carried out with an Intel i7 (3.40GHz, 8GB RAM).

We begin with a nonlinear example. Figure 3.1(a) presents computation times for identification of a SISO nonlinear model of the form (3.37), with $n_x = 4$, $\deg_x(e) = \deg_x(f) = 3$, and $\deg_x(g) = 1$. Specifically, we compare our proposed algorithm to Mosek v7.0.0.119

Table 3.2 – Computation time (in seconds, to 3 s.f.) for varying model order n_x and $T = 400$, averaged over 5 trials.

Model size, n_x	2	4	6	8
Specialized algorithm	0.339	2.74	8.74	34.9
Mosek 7.0.0.119	162	882	2550	7340

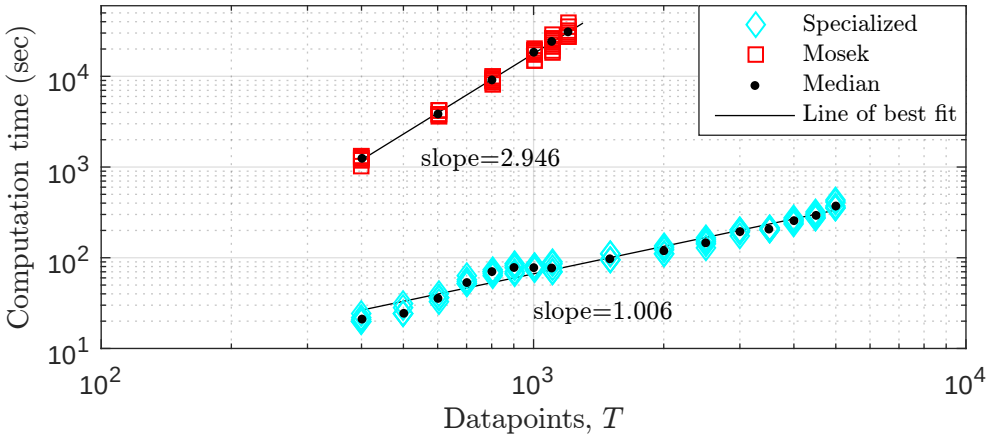
(using Yalmip [133] for SDP formulation), which in our experience is the best currently available general-purpose SDP solver. Problem data is generated by simulation of the nonlinear mass-spring-damper depicted in Figure 3.2 over time intervals of increasing length T. As the focus of this section is algorithmic scalability, we refer the reader to Section 3.5.1 for simulation details. Examining Figure 3.1(a), it is clear that the specialized algorithm exhibits better scalability compared to Mosek. In fact, for the specialized algorithm, the slope of the line of best is 1.006 indicating approx. linear growth with T, whereas the slope for Mosek is 2.946, indicating approx. cubic growth. This is consistent with the analysis of Section 3.4.1 and Section 3.2.5. Furthermore, for $T > 1200$, Mosek reports an `out of memory` error and fails to return a solution.

Before moving on, we note that in many of the trials depicted in Figure 3.1(a), Mosek encountered numerical problems, and often reported `unknown` as the final solution status. In such cases, one cannot have confidence in the feasibility (much less, optimality) of the solution. In contrast, our primal only interior-point method ensures feasibility of the solution (i.e. stability of the identified model) at every iteration. Furthermore, for every trial depicted in Figure 3.1(a), the objective value $\hat{J}_\lambda$ attained by our proposed algorithm was lower than the value obtained by Mosek.

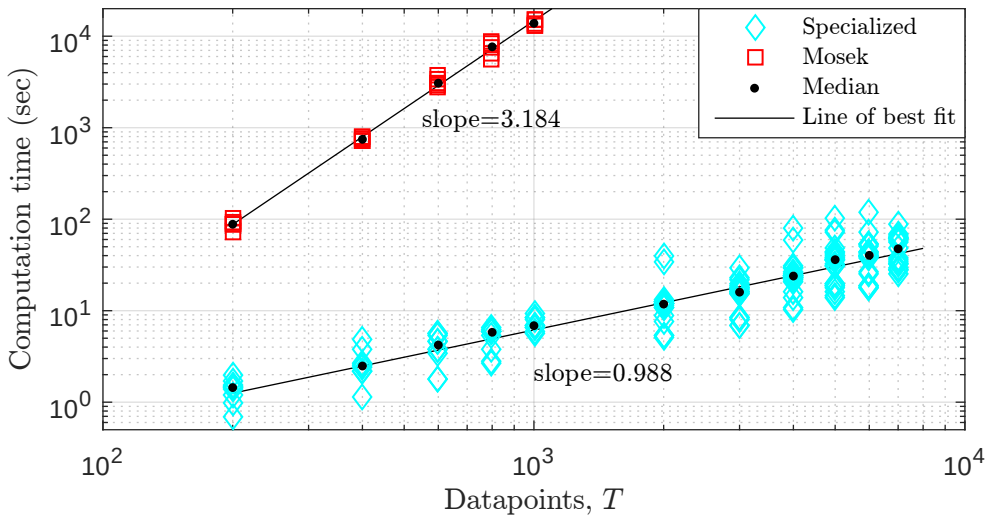
Next, we consider a linear example. Figure 3.1(b) presents computation times for identification of 4th order SISO LTI systems, again comparing our proposed algorithm to Mosek. In each trial, the true system was randomly generated using Matlab's `drss` function, and simulated for T timesteps, excited by a white noise input. The output was corrupted by additive white noise to give a SNR of 17dB, and N4SID [241] was used to obtain the state estimates $\{\tilde{x}_t\}_{t=1}^T$. As in the nonlinear example, the results support the claim that scalability of the specialized algorithm linear w.r.t. T, while Mosek is cubic, although there is slightly more variability in computation time due to the randomly generated test systems. Finally, Table 3.2 records computation times for varying model order n_x, with the length of the dataset held constant at $T = 400$ in all trials.

3.4.3 Relationship to other specialized solvers

Recall from Section 3.3.3, one of the main motivations for optimizing $\hat{J}_\lambda$ directly was avoiding the lifted representation (3.9) required by general-purpose solvers. In this lifted formu-



(a) Nonlinear identification.



(b) Linear identification.

Figure 3.1 – Computation times for solving $\min_\nu \ \hat{J}_\lambda(\nu)$ s.t. $S(\nu) \succeq 0$, as in (3.17), via two methods: our proposed algorithm (Specialized) and a general-purpose solver (Mosek). In (a) the identified SISO nonlinear model is of the form (3.37) with $n_x = 4$, $\deg_x(e) = \deg_x(f) = 3$ and $\deg_x(g) = 1$. For each value of T, 10 trials (each with different random noise and input realizations) were conducted. In (b) the identified 4th order SISO LTI model is randomly generated for each trial. For each value of T, 5 and 20 trials were conducted for Mosek and Specialized, respectively.

lation, the dimension of the LMI (3.9b), repeated here for convenience,

$$\begin{bmatrix} s & \epsilon(\rho)' & \eta(\rho)' \\ \epsilon(\rho) & \mathcal{F}(\rho)+\mathcal{F}(\rho)' & \mathcal{G}(\rho)' \\ \eta(\rho) & \mathcal{G}(\rho) & I_{Tn_y} \end{bmatrix} \succeq 0,$$

grows linearly with the number of data points, T, leading to worst-case per-iteration computational complexity that is cubic in T.

Though large, the LMI (3.9b) is high structured. In fact, as $\mathcal{F}$ and $\mathcal{G}$ are block Toeplitz and block diagonal, respectively, (3.9b) has a sparsity pattern characterized by a *chordal* graph. Since early 2000s [67, 158], there has been considerable research into exploiting chordal sparsity in semidefinite programming, c.f. Section 2.2.3 for a brief discussion. One such example is the recent work [6], which presents nonsymmetric interior-point methods for optimization over semidefinite cones with chordal sparsity patterns. Specifically, a number of algorithms for computing the search direction (i.e. solving the Newton equations) in primal scaling and dual scaling methods are derived, based largely on the zero fill-in Cholesky factorization for matrices with chordal sparsity, c.f. [6, §4], and (2.63) in Chapter 2. The authors utilize these algorithms in a feasible-start primal scaling method, and empirically demonstrate superior per-iteration computational complexity compared to other SDP solvers on a variety of problems. Specifically, the cost of each iteration is shown to grow linearly with the dimension of the LMI when the sparsity pattern is chordal, compared to quadratic growth for the alternative solvers.

The solver demonstrated in [6] and the algorithm we propose in this chapter have per-iteration complexity that scales linearly with problem size; although [6] is of course more generally applicable. The difference is, [6] exploits properties of matrices with chordal sparsity, whereas we exploit structural properties of Lagrangian relaxation, namely, simplified expressions for the gradient and Hessian, c.f. Section 3.3.3. Comparison of these two methods for minimization of $\hat{J}_\lambda$ is the subject of current research. Such a comparison is necessarily empirical in nature, and shall be concerned with differences in computation time, as well as quality of the identified models.

3.5 Case study: Mechanical system with nonlinear spring

In this section, we consider identification of a mechanical system with nonlinear spring stiffness. Accurate modeling of such systems is critical in several application areas, e.g. microelectromechanical systems (MEMS) [151] and precision motion control [167].

3.5.1 System Description

A schematic of the system is shown in Figure 3.2. The springs have a nonlinear characteristic given by:

$$k(s) = k \tan\left(\frac{\pi s}{2 \times 1.25}\right), \; s \in [-1.25, 1.25]. \tag{3.36}$$

To generate training data, the system is simulated for 100 seconds with `ode45`, excited by a superposition of sinusoidal forces, each with randomized frequency, phase and amplitude. We sample the input force $\tilde{f}$ and the displacement of the two masses, $s^{(1)}$ and $s^{(2)}$, at 10Hz, to give discrete time data $\tilde{f}_t = \tilde{f}(t \times T_s)$ and $s_t^{(i)} = s^{(i)}(t \times T_s)$, $i = \{1, 2\}$, $T_s = 0.1$. We then corrupt the displacement data with additive Gaussian noise $\tilde{s}_t^{(i)} = s_t^{(i)} + w_t^{(i)}$, $w_t^{(i)} \sim \mathcal{N}(0, 10^{-4})$, $i \in \{1, 2\}$, to simulate measurement errors, giving a signal-to-noise ratio (SNR) of approx. 34dB. Our goal is to model the dynamics from the input force to the position of the second mass, i.e., $\{\tilde{u}_t, \tilde{y}_t\}_{t=1}^T = \{\tilde{f}_t, \tilde{s}_t^{(2)}\}_{t=1}^T$ with $T = 10^3$. To estimate the internal states $\{\tilde{x}_t\}_{t=1}^T$, used in the construction of the Lagrange multipliers, c.f. Section 3.2.4, we take

$$\tilde{x}_t = \left[\tilde{s}_t^{(1)}, \; \tilde{s}_t^{(2)}, \; \frac{\tilde{s}_{t+1}^{(1)} - \tilde{s}_{t-1}^{(1)}}{T_s}, \; \frac{\tilde{s}_{t+1}^{(2)} - \tilde{s}_{t-1}^{(2)}}{T_s}\right]',$$

i.e., we exploit our knowledge of the system structure and approximate the velocities by the central difference.

In the following case studies, we will apply Lagrangian relaxation to implicit models of the form (3.2), with

$$e : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x} = [e_1(x), \ldots, e_{n_x}(x)]', \tag{3.37a}$$

$$f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \mapsto \mathbb{R}^{n_x} = [f_1(x, u), \ldots, f_{n_x}(x, u)]', \tag{3.37b}$$

$$g : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \mapsto \mathbb{R}^{n_y} = [g_1(x, u), \ldots, g_{n_y}(x, u)]'. \tag{3.37c}$$

Each function e_i, f_i and g_i is a scalar valued, multivariate polynomial, the degree of which will be specified for each application example. We will use the term "degree n", and the notation $\deg_x(p) = n$, to refer to a polynomial p containing all possible monomials in x up to degree n, e.g., if $n_x = 2$ then "e_1 *is degree* 2", or $\deg_x(e_1) = 2$, implies

$$e_1(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2,$$

where $\{\theta_i\}_{i=0}^5$ are the parameters to be identified.

Performance of identified models shall be quantified by the normalized simulation error, $\frac{\sum_t \|\tilde{y}_t - y_t\|^2}{\sum_t \|\tilde{y}_t\|^2}$, where y_t denotes the simulated output of the model and $\tilde{y}_t$ denotes measured output from the system of interest.

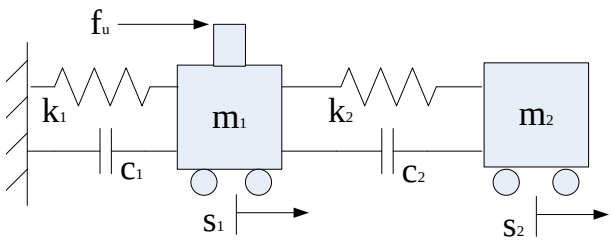


Figure 3.2 – Mass-spring-damper system, with parameters $m_1 = 0.5\ kg$, $m_2 = 0.1\ kg$, $c_1 = 0.01\ Nsm^{-1}$, $c_2 = 0.1Nsm^{-1}$. The spring has the nonlinear force-displacement curve (3.36) with $k_1 = 2$, $k_2 = 1$. The measured control input is force f_u and the measured system outputs are the displacements s_1 and s_2.

3.5.2 Comparison to RIE and equation error

We first compare the Lagrangian relaxation approach to two other methods that utilize the same model structure but alternative convex surrogates for simulation error. The first is minimization of the Local Robust Identification Error (RIE) [230], which also gives a convex upper bound on simulation error, and was developed as a tractable approximation to Lagrangian relaxation. The second is minimization of equation error (EE), i.e.,

$$\min_{\theta} \sum_{t=1}^{T} \|\eta_t\|^2 + \sum_{t=1}^{T-1} \|\epsilon_t\|^2 \text{ s.t. } E(x) + E(x)' \in \text{SOS}, \tag{3.38}$$

where the SOS constraint ensures that the identified model is well-posed (i.e. $e(\cdot)$ is a bijection). Equation error is a form of one-step-ahead prediction error, frequently used in system identification [129] and, for the case of linear systems, is exactly the algorithm of [113].

Identified models are of the form (3.37), with $\{f_i(x,u)\}_{i=1}^4$ affine in u. The results are presented in Figure 3.3, for models of increasing complexity. Figure 3.3(a) depicts performance on training data, for 30 different training data realizations. Figure 3.3(b) plots the performance of these 30 different instances of each model, for a single realization of validation data. Computation times are listed in Table 3.3.

Table 3.3 – Mean computation times (in seconds, to 3 sig. fig) for the methods applied in the 30 experimental trials depicted in Figure 3.3. The parenthesized numbers refer to degree of e, f, g, respectively.

Model	$(1,1,1)$	$(3,1,1)$	$(3,3,1)$	$(5,3,1)$
LR	4.45	44.4	67.3	1280
RIE	9.47	20.6	26.4	172
EE	3.60×10^{-3}	4.19×10^{-2}	4.25×10^{-2}	4.28

Table 3.4 – Median values (to 3 sig. fig.) of the convex upper bounds on simulation error for the 30 experimental trials presented in Figure 3.3(a). The parenthesized numbers refer to degree of e, f, g, respectively.

Model	$(1,1,1)$	$(3,1,1)$	$(3,3,1)$	$(5,3,1)$
Bound, LR	12.1	5.08	4.25	2.89
Bound, RIE	15.5	12.0	11.3	9.04

It is clear that in terms of model fidelity, LR is the best, followed by RIE, with EE worst. In terms of computation time, the ranking is reversed. This is perhaps unsurprising: mini-

mizing EE is essentially least-squares with a small SDP constraint, and RIE was proposed as a simpler alternative to LR in [232].

We observe that for both LR and RIE, model fidelity improves (i.e. simulation error decreases) monotonically with increasing model complexity. In contrast, performance of models fit by minimization of EE is more erratic and exhibits large variance. EE is susceptible to the well-known bias-variance tradeoff: comparing EE(1,1,1) to EE(5,3,1), we see that increasing model complexity reduces median error at the expense of large variance. In this situation, a standard remedy would be regressor pruning [24].

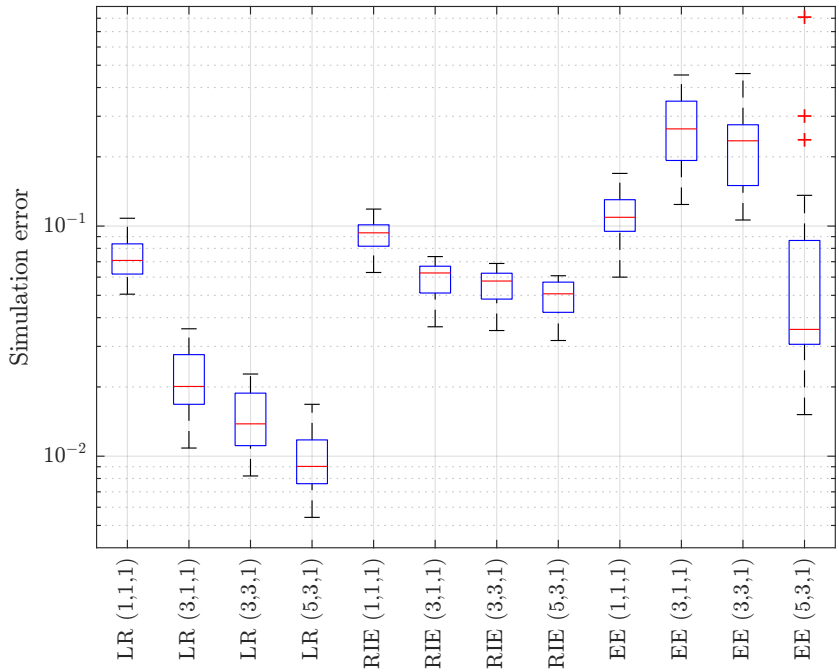
Both LR and RIE models achieve lower median error as model complexity increases without any increase in variance. It should be noted that the increase in complexity from (3,3,1) to (5,3,1) is significant; these models contain 271 and 1785 parameters, respectively. Given that only 2000 input/output datapoints were used for identification, this is evidence of the regularizing effect of model stability constraints and "robust" simulation error bounds.

Finally, in Table 3.4 we record the median values of the minimized convex upper bounds on simulation error, for the experimental trials in Figure 3.3(a). for both LR and RIE the tightness of the bound improves (i.e. the value decreases) with increasing model complexity. This is entirely as expected, as increased model complexity corresponds to increased size of the feasible convex set over which the convex bound is minimized. Furthermore, the bound from LR is always tighter than that from RIE. This is guaranteed by construction of the LR, c.f. [232, Theorem 6]

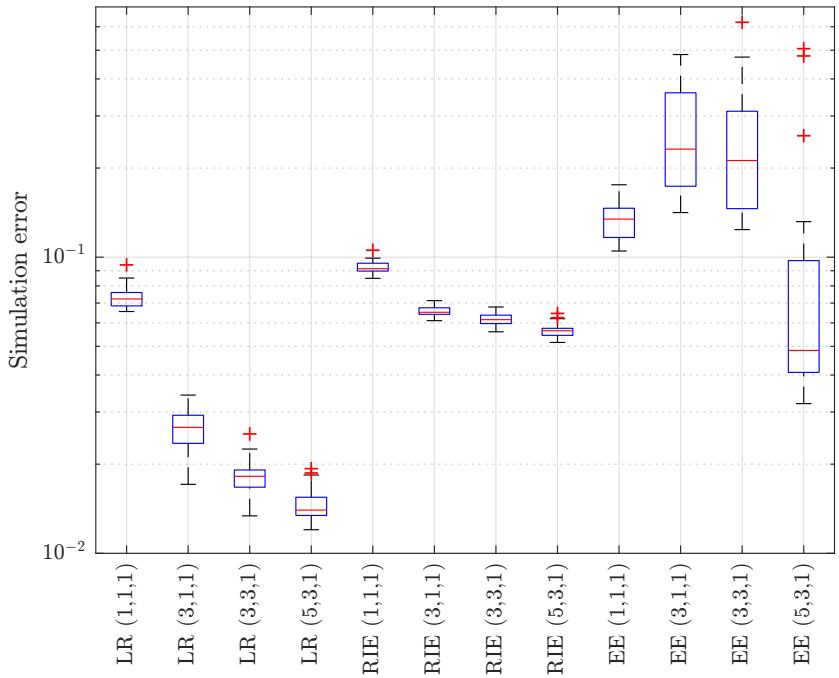
3.5.3 Comparison to Nonlinear ARX

Next, we compare our algorithm a standard approach: Nonlinear AutoRegressive models with eXogenous inputs (NARX), as implemented in the Matlab System Identification Toolbox. In particular, we compare the following identification methods:

- *LR* - The proposed Lagrangian relaxation algorithm, applied to a model of the form (3.37) with (e,f,g) of degree (3,3,1) respectively.
- *Poly* - Nonlinear regressors are all monomials in $\{\tilde{s}_n^{(1)}, \tilde{s}_n^{(2)}\}$ (for $n = t-1, t-2$) up to degree 3; `focus` = `simulation`.
- *Sig** - sigmoid nonlinearity; `nlreg`=`search` to select regressors; `focus` = `prediction`.
- *Sig* - sigmoid nonlinearity; all nonlinear regressors used; `focus` = `simulation`.
- *Wav** - wavelet nonlinearity; `nlreg`=`search` to select regressors; `focus` = `prediction`.
- *Wav* - wavelet nonlinearity; all nonlinear regressors used; `focus` = `simulation`.



(a) Simulation error on training data (30 realizations).



(b) Simulation error on validation data.

Figure 3.3 – Comparison of different methods for fitting polynomial state-space models: Lagrangian relaxation (proposed method, LR); Local Robust Identification Error (RIE), c.f. [230]; equation error (EE), c.f. (3.38). Parenthesized numbers denote the degrees of the polynomials (e,f,g) for models of the form (3.37). Refer to Section 3.5.1 for experimental details.

Table 3.5 – Computation times (in seconds, to 3 sig. fig.) for the methods applied in the 30 experimental trials depicted in Figure 3.4.

Method	*LR*	*Poly*	*Sig**	*Sig*	*Wav**	*Wav*
Mean	63.2	1690	4930	58.3	2080	53.7
Std. Dev.	3.87	486	64.9	28.1	12.8	43.1

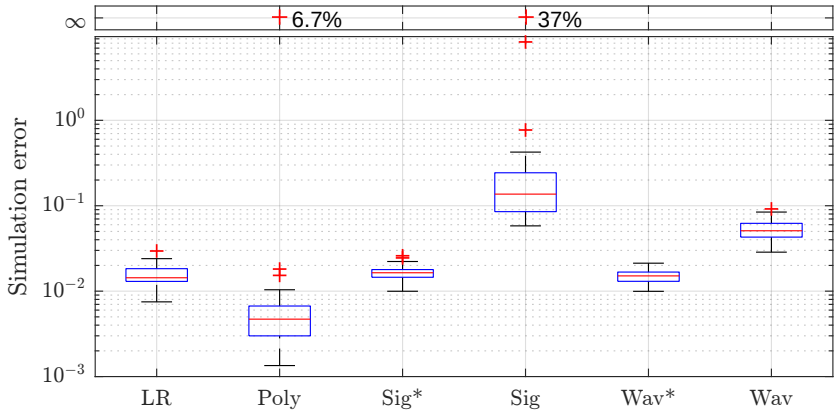
Each NARX model uses six regressors $\{y_n^{(1)}, y_n^{(2)}, u_n\}_{n=t-1}^{t-2}$, with $\{y_n^{(i)}, u_n\} = \{\tilde{s}_n^{(i)}, \tilde{u}_n\}$ for training. The `focus` property was set so as to produce the best performance for each model. This was important for the *Poly* model, where `simulation` performed much better than `prediction`, but less so for the others, where the `focus` property had little influence.

For each of the methods tested, 30 models were attained by fitting to 30 randomly generated training datasets; c.f. Section 3.5.1. Performance of these models on training data is depicted in Figure 3.4(a). For validation, we randomly generate a *single* new dataset and compute the simulation error of each of the 30 models; the results are presented in Figure 3.4(b). Many NARX models were unstable, and the simulations diverged. To keep the scale of Figure 3.4 meaningful, we collect these at the top as ∞ simulation error, and the box plots are generated using only the stable models. Note that the *same models* are being simulated in Figures 3.4 (a) and (b), but different proportions of models were divergent. This is because (local) stability of a nonlinear model is trajectory-dependent. On the other hand, the global incremental stability constraint (3.35) for LR ensures stability for *all* possible inputs.

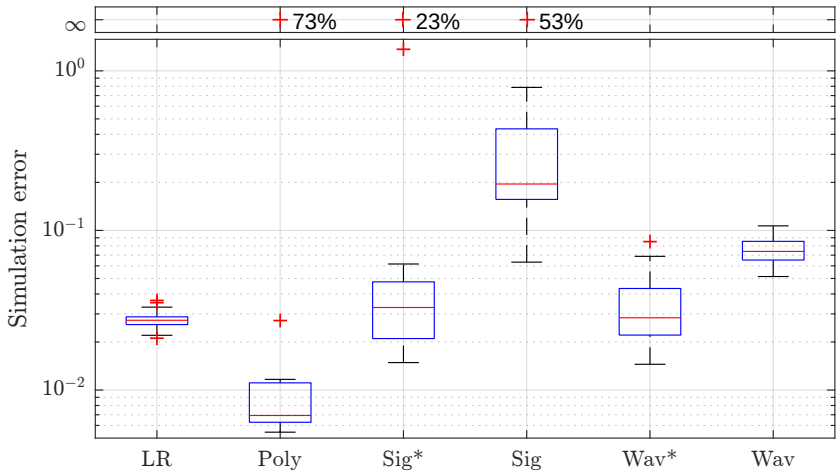
Some interesting observations can be made from Figure 3.4. Foremost, we note that *LR* outperforms NARX, achieving the lower median error than all other methods. The apparent lower median of *Poly* is is not a real effect: since 73% of models diverged it could be said that for *Poly* the median simulation error is infinity.

The computation times are recorded in Table 3.5. Computationally, *LR* is comparable with *Sig* and *Wav*, although *LR* achieves significantly lower (i.e. better) simulation error. Only *Wav** has similar median error to *LR*, though with larger variance, but it took around 30 times longer to compute due to the costly subset selection process. Before moving on, we note that even better performance can be attained by LR, at the expense of greater computational effort, if we are willing to use a more complicated model, e.g. LR(5,3,1) in Figure 3.3. Notice that LR(5,3,1) is still twice as fast to fit compared to *Wav**, c.f. Table 3.3.

Comparing *LR* to the subset selection methods *Sigmoid** and *Wavelet**, we observe that the *variance* of simulation error on validation data is much lower for *LR*. We suggest that this is due to the large variation in the structure (i.e. selected regressors) of models from subset selection. Table 3.6 reports the frequency with which individual regressors were



(a) Training data, 30 models per method.



(b) Validation data, 30 models per method.

Figure 3.4 – Comparison of proposed method (LR) to various nonlinear ARX models; c.f. Section 3.5.3 for a complete description of the models and methods. The percentages at infinite error denote the proportion of trials for which the simulated model diverged.

Table 3.6 – Frequency with which certain nonlinear regressors were chosen by Matlab's subset selection algorithm (i.e. `nlreg` set to `search`) during the 30 experimental trials depicted in Figure 3.4.

Regressor	$y_{t-1}^{(1)}$	$y_{t-2}^{(1)}$	$y_{t-1}^{(2)}$	$y_{t-2}^{(2)}$	u_{t-1}	u_{t-2}
Wavelet*, y_1	77%	0%	17%	3%	7%	20%
Wavelet*, y_2	37%	27%	30%	13%	93%	97%
Sigmoid*, y_1	80%	10%	17%	10%	30%	33%
Sigmoid*, y_2	33%	43%	27%	30%	67%	57%

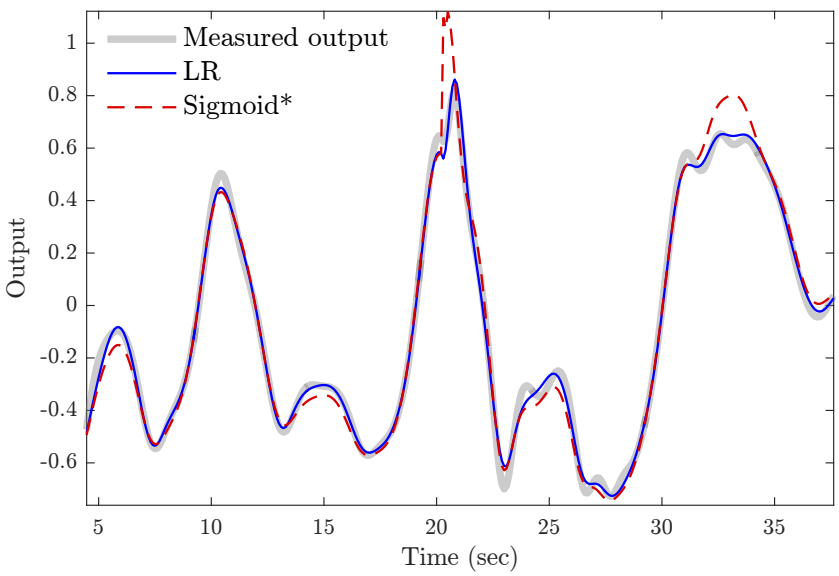


Figure 3.5 – Simulated performance on validation data for one of the trials in Figure 3.4. *LR* denotes a 4th order state-space model fit with our proposed algorithm. *Sigmoid** denotes a nonlinear ARX model with sigmoid net nonlinearity and regressors chosen automatically by Matlab's subset selection algorithm; see Section 3.5.3 for details. Normalized simulation error for *LR* and *Sigmoid** are 2.02×10^{-2} and 3.71×10^{-2}, respectively.

chosen by Matlab's subset selection algorithm. Notice that there isn't a single regressor that was selected in 100% of trials. Since subset selection is inherently nonsmooth, and small variations in the training data can lead to large differences in model structure (i.e. selected regressors), having an adverse effect on the ability of these models to generalize. By contrast, our proposed LR algorithm involves a minimizing a *smooth convex* function over a convex set, and small changes in the problem data do not result in large changes in the identified model.

3.6 Case study: Two tank system

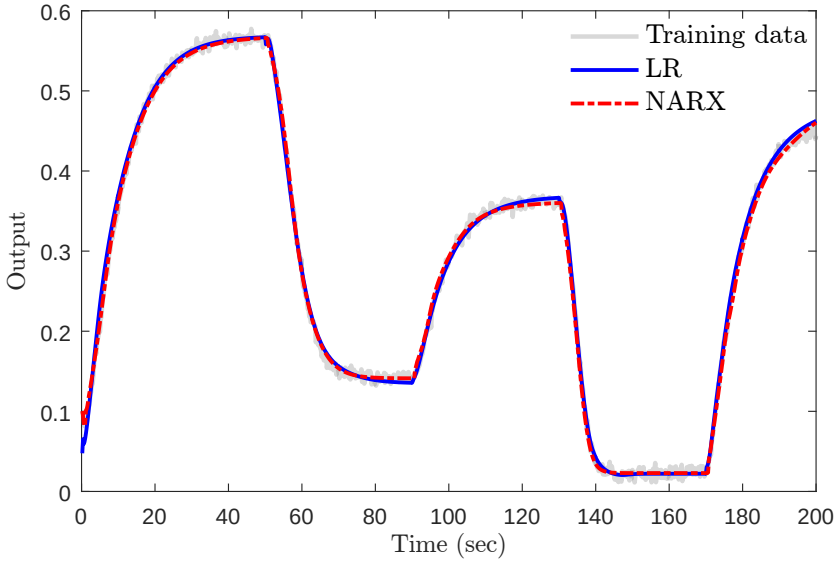
In this section, we seek to model a system consisting of two interconnected tanks. The input is the voltage $\tilde{u}$ (V) applied to a pump, which delivers fluid to Tank One. Fluid then flows from an outlet in the bottom of Tank One to Tank Two. System output is the depth $\tilde{y}$ (m) of fluid in Tank Two. These signals are sampled at 5Hz to produce the discrete-time training dataset $\{\tilde{u}_t, \tilde{y}_t\}_{t=1}^{T}$, where $T = 10^3$. For further details and access to the problem data, c.f. [147].

We compare our proposed Lagrangian relaxation method to the best performing NARX model from [147], comprising 8 linear regressors $\{\tilde{y}_{t-1}, \ldots, \tilde{y}_{t-5}, \tilde{u}_{t-1}, \ldots, \tilde{u}_{t-3}\}$ and 2 nonlinear regressors $\{\tilde{y}_{t-4}, \tilde{u}_{t-3}\}$ with 12 unit wavelet nonlinearities. The polynomial model fit with Lagrangian relaxation is of the form (3.37) with $n_x = 3$, $\{e_i\}_{i=1}^{3}$ degree 5, $\{f_i = f_i^x(x) + f_i^u(u)\}_{i=1}^{3}$ with $\{f_i^x\}_{i=1}^{3}$ degree 3 and $\{f_i^u\}_{i=1}^{3}$ degree 4, $g = g^x(x) + g^u(u)$ with g^x degree 3 and g^u degree 4. To estimate the internal states $\{\tilde{x}_t\}_{t=1}^{T}$, used in the construction of the Lagrange multipliers, we apply the subspace algorithm of [241, Section 4.3.1], with $n_x = 3$.

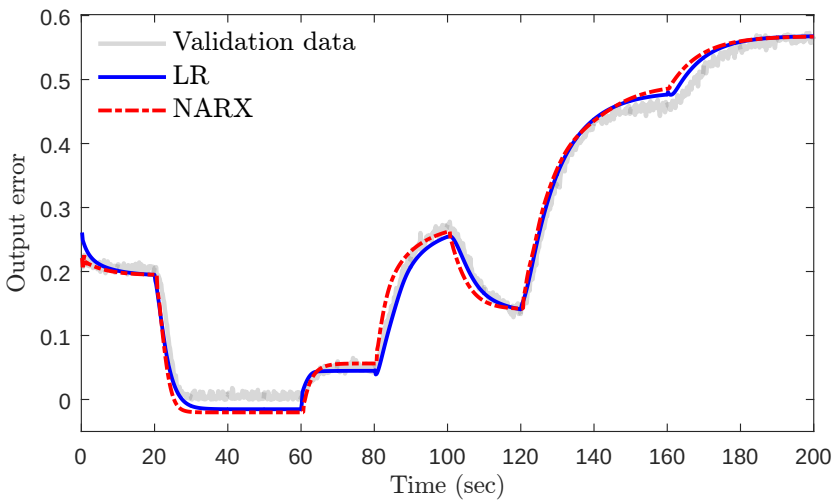
The simulated performance of each model is depicted in Figure 3.6 and recorded in Table 3.7. We observe that LR performs significantly better (49% improvement) on validation data, compared to the best NARX model.

Table 3.7 – Normalized simulation error for training and validation data from the two tank system.

Method	LR	NARX
Training	3.21×10^{-4}	3.62×10^{-4}
Validation	2.52×10^{-3}	4.91×10^{-3}



(a) Simulated output on training data.



(b) Simulated output on validation data.

Figure 3.6 – Simulated performance for a 3rd order state-space model fit with our proposed algorithm compared to a NARX model; see Section 3.6 for details. True data is collected from a two tank system [147].

3.7 Case study: bias in linear system identification

To illustrate the performance of our proposed algorithm on a wide variety of linear models, we first conducted the following numerical experiment: Matlab's `drss` function was used to randomly generate forty 8th order LTI SISO systems. Each system was excited with white noise and simulated for $T = 400$ time steps to generate input/output data $\{\tilde{u}_t, \tilde{y}_t\}_{t=1}^{T}$. The algorithm of [155] was used obtain an approximate state sequence $\{\tilde{x}_t\}_{t=1}^{T}$ in a balanced

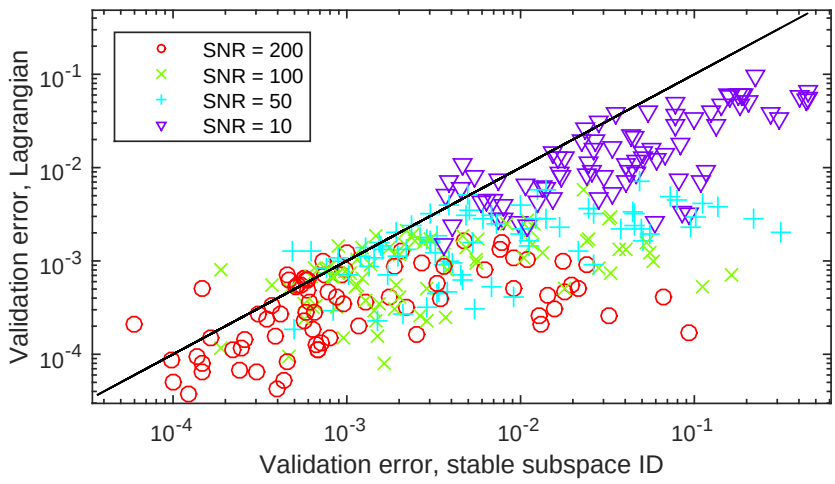


Figure 3.7 – Performance of our proposed algorithm compared with stable subspace ID for the identification of forty 8th order SISO models, randomly generated by Matlab's `drss` function.

basis.

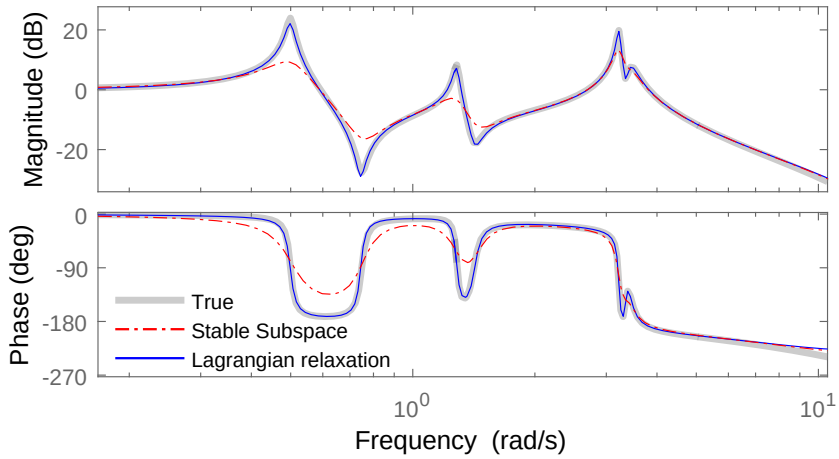
We then fit 8th order linear models to the data using two methods: i) our proposed algorithm, and ii) minimization equation error, weighted by $P \in \mathbb{S}_{++}$, subject to model stability constraints, i.e.,

$$\min_{\theta} \sum_{t=1}^{T} \|\tilde{y}_t - C\tilde{x}_t - D\tilde{u}_t\|^2 + \|P\tilde{x}_{t+1} - \mathcal{A}\tilde{x}_t - \mathcal{B}\tilde{u}_t\|^2 \tag{3.39}$$

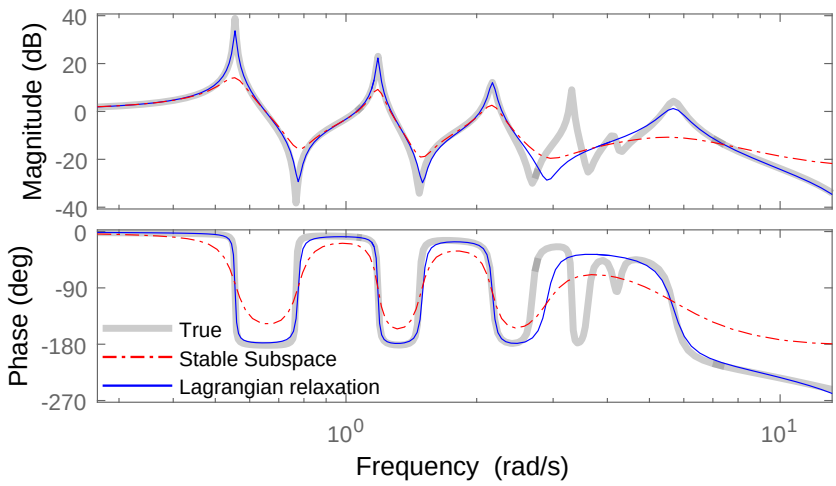
$$\text{s.t.} \quad \begin{bmatrix} P - \delta I & \mathcal{A} \\ \mathcal{A}' & P \end{bmatrix} \succeq 0, \tag{3.40}$$

where $\theta = \{P, \mathcal{A}, \mathcal{B}, C, D\}$. A stable LTI model can then be recovered as $A = P^{-1}\mathcal{A}$ and $B = P^{-1}\mathcal{B}$. This method is henceforth referred to as 'stable subspace ID'. This process was repeated eight times for each model, over four different SNRs. The results of this experiment are shown in Figure 3.7, which records the validation error of each identified model. It is clear that models identified with our proposed algorithm outperform those from stable subspace ID in the majority (86%) of trials.

It has been observed by several authors that guaranteeing stability in system identification is often associated with a bias towards models that are "too stable" [138, 141, 229] and [113]. To gain further insight into this effect, we consider identification of a flexible beam, which serves as a useful model of cantilever structures arising in many engineering applications. In particular, we fit 8th order models to a 8th order (4-link) beam; the Bode plot for this system is given in Figure 3.8(a). The subspace algorithm [155] was used obtain an approximate state sequence $\{\tilde{x}_t\}_{t=1}^{T}$. Figure 3.9 plots identified pole locations for decreasing SNR. Observe

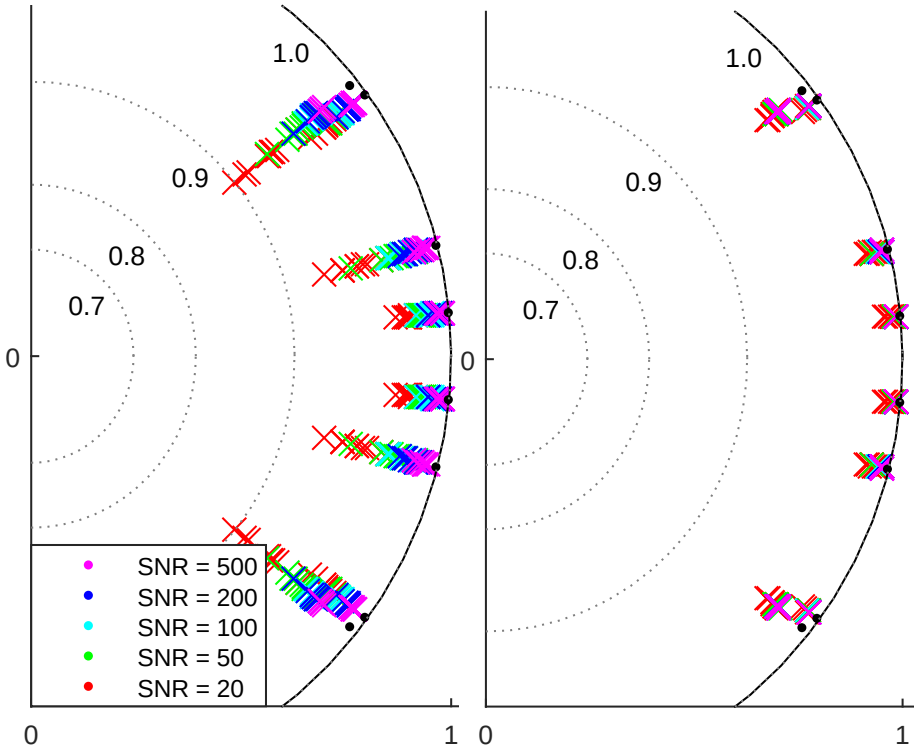


(a) 8th order model fit to 8th order true system.



(b) 8th order model fit to 12th order true system (undermodeling).

Figure 3.8 – Bode plots for the true flexible beam model (gray), and 8th order models identified by Lagrangian relaxation (blue) and stable subspace ID (red). In (a), the true system is 8th order, while in (b) the true system is 12th order; i.e. undermodeling is present. The output SNR was 100 (20dB).



(a) Stable subspace ID. **(b)** Lagrangian relaxation.

Figure 3.9 – Pole locations of 8th order models fit to an 8th order flexible beam; c.f. Figure 3.8 for the Bode plot. The small dots denote the poles of the true model, '×' the poles of identified models.

that the poles of models identified by stable subspace ID have been shifted considerably towards the center of the unit circle, compared to those of the models from Lagrangian relaxation.

Figure 3.8(a) presents Bode plots for identified models from one of these trials. The inability of the model from stable subspace ID to capture the resonant peaks – and associated phase shifts – is a consequence of the poles being pulled in towards the origin.

In most real applications, there is some degree of *undermodeling*: i.e. the model identified is of lower order than the true system. To examine performance in this situation, we repeated the above experiments but fit 8th-order models is fit to data from a 12th order system, representing a six-link beam. The resulting Bode plots are shown in Figure 3.8(b). It is clear that LR does a good job of capturing four resonant peaks (as expected with an 8th-order model), and a reasonable job of interpolating through the remaining two. The particular peaks that are captured depend on the spectra of the forcing input. Stable Subspace again fails to capture the resonance.

3.8 Conclusion

In this chapter, we have developed interior point algorithms for minimization of Lagrangian relaxation of simulation error, and linearized simulation error, for linear and nonlinear dynamical systems, respectively. In both settings, stability of the identified model is guaranteed. The algorithms take advantage of special structure in the problem to reduce computational complexity from cubic to linear in the data length, compared to a generic SDP solver.

Equipped with this specialized algorithm, we demonstrate superior performance of the proposed method over several established methods, and discuss the apparent regularizing effect of stability constraints and robust fidelity bounds.

Chapter 4

Maximum likelihood identification of stable systems

The Maximum Likelihood criterion is used extensively in a wide range of statistical inference problems. In fact, when the model parametrization includes a description of the true system, the ML estimator is optimal in the sense that it asymptotically achieves the Cramér-Rao lower bound; c.f. [129, §7.4]. As we have stressed, for applications in which the long-term, open-loop predictive power of the model is important, stability of the identified model is vital. Of the existing methods designed to ensure stability, surveyed in Section 2.3, all depart from (and thus fail to inherit the desirable properties of) the ML framework.

In this chapter, we incorporate model stability constraints into maximum likelihood identification of dynamical systems. The approach draws upon the underlying similarity between Lagrangian relaxation and Expectation Maximization; both of which are techniques to generate bounds that are more easily optimized than the cost functions they approximate. More specifically, the EM algorithm is an iterative approach to ML estimation in which estimates of the unknown latent variables, typically unobserved system states, are used to construct tractable lower bounds on the likelihood. We use Lagrangian relaxation to derive alternative bounds on the likelihood, that have advantage of being able to be optimized over convex parameterizations of stable models.

Furthermore, we also propose a novel formulation of the EM algorithm in which system *disturbances*, rather than system states, are taken as the latent variables. We show that the latent states formulation provides tighter bounds on the likelihood when the effect of disturbances is more significant than the effect of measurement noise. Conversely, the latent disturbances formulation gives tighter bounds when the situation is reversed. We also show that use of latent disturbances gives the most broadly applicable formulation of EM for identification of models with rank-deficient disturbance covariance, so-called *singular state space models*.

Connection to other chapters

One of the key tools used in this chapter is Lagrangian relaxation. Specifically, our latent disturbances formulation of EM involves the Lagrangian relaxation of many simultaneous simulation error minimization problems. Without the efficient custom interior point methods developed in Chapter 3, this approach would be computationally intractable. In this sense, the latent disturbances formulation developed in this chapter can be viewed as an important application area for the specialized algorithms developed in Chapter 3.

On the other hand, the Lagrangian relaxation of simulation error studied in Chapter 3 has two potential shortcomings, depending on the application. First of all, the method applies only to deterministic models; no attempt is made to explicitly model disturbances or measurement noise. Modeling these random processes can be important in some applications, e.g., if the model is to be used for state inference or design of control systems that are robust to disturbances. Second, the constructions optimized in Chapter 3, especially linearized simulation error, depend upon an approximate state sequence, $\tilde{x}_{1:T}$. In many applications, internal system states cannot be measured directly. Various strategies and ad-hoc solutions to this problem exist; however, in general, there are no mature techniques for estimation of internal states of a nonlinear dynamical system in the absence of an approximate model, research on various embedding theorems notwithstanding, c.f., [226][165][220][221]. This is in contrast to identification of linear systems, where subspace methods fulfill this role. In this sense, the EM based algorithms proposed in this chapter can be viewed as an iterative approach to generating approximate state sequences for the Lagrangian relaxations constructed in Chapter 3.

Publications

This material presented in this chapter also appears in the following publications:

> **J. Umenberger**, J. Wågberg, I.R. Manchester, T.B. Schön. Maximum likelihood identification of stable linear dynamical systems. *Automatica*. 2017. *Conditionally accepted for publication.*

> **J. Umenberger**, J. Wågberg, I.R. Manchester, T.B. Schön. On identification via EM with latent disturbances and Lagrangian relaxation. In *Proceedings of the IFAC Symposium on System Identification (SYSID)*. 2015.

4.1 Introduction

This chapter is concerned with identification of discrete-time linear Gaussian state space (LGSS) models of the form,

$$x_{t+1} = Ax_t + Bu_t + w_t, \tag{4.1a}$$

$$y_t = Cx_t + Du_t + v_t, \tag{4.1b}$$

where $x_t \in \mathbb{R}^{n_x}$ denotes the system state, and $u_t \in \mathbb{R}^{n_u}$, $y_t \in \mathbb{R}^{n_y}$ denote the observed input and output, respectively (henceforth, resp.). The disturbances, $w_t \in \mathbb{R}^{n_w}$ and measurement noise, v_t, are modeled as zero mean Gaussian white noise processes, while the uncertainty in the initial condition x_1 is modeled by a Gaussian distribution, i.e.

$$w_t \sim \mathcal{N}(0, \Sigma_w),\ v_t \sim \mathcal{N}(0, \Sigma_v),\ x_1 \sim \mathcal{N}(\mu, \Sigma_1). \tag{4.2}$$

For convenience, all unknown model parameters are denoted by the variable

$$\theta = \{\mu, \Sigma_1, \Sigma_w, \Sigma_v, A, B, C, D\}.$$

Despite the simplicity of (4.1), identification of LTI systems is complicated by (at least) two factors: *latent variables* and *model stability*, the latter being a desirable property in many applications. Typically, observed data consists of inputs and (noisy) outputs only; the internal states or exogenous disturbances are *latent* or 'hidden'. Bilinearity of (4.1) in x and θ renders the search for model parameters nonconvex. Similarly, the set of stable matrices, $\mathcal{S}$, is also nonconvex.

Various strategies have been developed to deal with this 'hidden data'. *Marginalization*, for instance, involves integrating out (i.e. marginalizing over) the latent variables, leaving θ as the only quantity to be estimated. This approach is adopted by prediction error methods [129, 130] (PEM) and the Metropolis-Hastings algorithm [87, 152].

Alternatively, one may treat the latent variables as additional quantities to be estimated together with the model parameters. Such a strategy is termed *data augmentation*, and examples include subspace methods [114, 239], and the Expectation Maximization (EM) algorithm [51, 71, 203, 207]. The augmentation together with appropriate priors also allows for closed form expressions in a Gibbs sampler [70, 259], (as a special case of the Metropolis-Hastings algorithm).

Recently, a new family of methods have been developed in which one *supremizes over* the latent variables, with an appropriate multiplier, to obtain convex upper bounds for quality-of-fit cost functions, such as output error; c.f. Section 2.4 of this thesis, and the references within. An important technique employed in this approach is Lagrangian relaxation [119] (also used in combinatorial optimization, and closely related to the S-procedure [181, 263]

in robust control), which replaces difficult constrained optimization problems with tractable convex approximations.

In this chapter, we seek the maximum likelihood (ML) estimate of the model parameters θ, given measurements $u_{1:T}$ and $y_{1:T}$, subject to model stability constraints, i.e.

$$\hat{\theta}^{\mathrm{ML}} = \arg\max_{\theta} p_\theta(u_{1:T}, y_{1:T}) \text{ s.t. } A \in \mathcal{S}. \tag{4.3}$$

ML methods have been studied extensively and enjoy desirable properties, such as asymptotic efficiency; see, e.g., [129, Chapters 7 and 9]. Stability of identified state space models has been investigated by a number of authors, using tools such as regularization [238], linear matrix inequality (LMI) and polytopic parametrizations of stable models [113, 154, 230], and modifications to the shifted state matrix in subspace algorithms [138]. However, the resulting methods do not fall within, nor inherit the desirable properties of, the ML framework; e.g. [138, 230] are known to bias the estimated model, and even unconstrained subspace methods are generally considered to be less accurate than PEM [59].

The work in this chapter draws on the underlying similarities between EM and Lagrangian relaxation to incorporate model stability constraints into the ML framework. The EM algorithm is an iterative approach to ML estimation, in which estimates of the latent variables are used to construct tractable lower bounds to the likelihood. We use Lagrangian relaxation to derive alternative bounds on the likelihood, that have advantage of being able to be optimized over a convex parameterization of all stable linear models, using standard techniques such as semidefinite programming (SDP).

In this chapter, we treat both the latent states and latent disturbances formulation of EM, leading to two algorithms: EM with latent States & Lagrangian relaxation (EMSL), and EM with latent Disturbances & Lagrangian relaxation (EMDL). The former represents the *de facto* choice of latent variables; however, we show that the latter can lead to higher fidelity bounds on the likelihood, when the effect of measurement noise is more significant than that of the disturbances. We also show that latent disturbances lead to the most broadly applicable formulation of EM for identification of singular state space models.

4.2 Preliminaries

4.2.1 Notation

Specific notation used in this chapter is as follows. The cone of real, symmetric nonnegative (positive) definite matrices is denoted by $\mathbb{S}^n_+$ ($\mathbb{S}^n_{++}$). The $n \times n$ identity matrix is denoted I_n. Let $\mathrm{vec} : \mathbb{R}^{m\times n} \mapsto \mathbb{R}^{mn}$ denote the function that stacks the columns of a matrix to produce a column vector. The Kronecker product is denoted $\otimes$. The transpose of a matrix A is denoted A'. For a vector a, $|a|^2_Q$ is shorthand for $a'Qa$. Time series data $\{x_t\}_{t=a}^b$ is denoted $x_{a:b}$ where $a, b \in \mathbb{N}$. A random variable x distributed according to the multivariate normal

distribution, with mean μ and covariance Σ, is denoted $x \sim \mathcal{N}(\mu, \Sigma)$. We use $a(\theta) \propto b(\theta)$ to mean $b(\theta) = c_1 a(\theta) + c_2$ where c_1, c_2 are constants that do not affect the minimizing value of θ when optimizing $a(\theta)$. The log likelihood function is denoted $L_\theta(y_{1:T}) := \log p_\theta(u_{1:T}, y_{1:T})$. The spectral radius (magnitude of largest eigenvalue) of a matrix A is $r_{\text{sp}}(A)$.

4.2.2 The minorization-maximization principle

The minorization-maximization (MM) principle [95, 166] is an iterative approach to optimization problems of the form $\max_\theta \; f(\theta)$. Given an objective function $f(\theta)$ (not necessarily a likelihood), at each iteration of an MM algorithm we first build a *tight* lower bound $b(\theta, \theta_k)$ satisfying

$$f(\theta) \geq b(\theta, \theta_k) \; \forall \; \theta \text{ and } f(\theta_k) = b(\theta_k, \theta_k),$$

i.e. we *minorize* f by b. Then we optimize $b(\theta, \theta_k)$ w.r.t. θ to obtain θ_{k+1} such that $f(\theta_{k+1}) \geq f(\theta_k)$. The principle is useful when direct optimization of f is challenging, but optimization of b is tractable (e.g. concave). In the following two subsections, we present EM and Lagrangian relaxation as special cases of the MM principle, for problems involving missing data. Each of these algorithms is predicated on the assumption that there exists latent variables, z, such that optimization of $f(\theta)$ would be more straightforward if z were known.

4.2.3 The Expectation Maximization algorithm

The EM algorithm [51] applies the MM principle to ML estimation, i.e., $f(\theta) = \log p_\theta(u_{1:T}, y_{1:T}) := L_\theta(y_{1:T})$. Each iteration of the algorithm consists of two steps: the expectation (E) step computes the *auxiliary* function

$$Q(\theta, \theta_k) := \int L_\theta(y_{1:T}, Z) p_{\theta_k}(Z|y_{1:T}) \; dZ \tag{4.4a}$$

$$= \mathrm{E}_{\theta_k}\big[L_\theta(y_{1:T}, Z)|y_{1:T}\big], \tag{4.4b}$$

which is then maximized in lieu of the likelihood function during the maximization (M) step. The auxiliary function can be shown to satisfy the following inequality

$$L_\theta(y_{1:T}) - L_{\theta_k}(y_{1:T}) \geq Q(\theta, \theta_k) - Q(\theta_k, \theta_k) \tag{4.5}$$

and so the new parameter estimate θ_{k+1} obtained by maximization of $Q(\theta, \theta_k)$ is guaranteed to be of equal or greater likelihood than θ_k. In this sense, EM may be thought of as a specific MM recipe for building lower bounds $Q(\theta, \theta_k)$ to the objective $L_\theta(y_{1:T})$, in ML estimation problems involving latent variables.

Remark 4.1. Strictly speaking $Q(\theta, \theta_k)$ does not minorize $L_\theta(y_{1:T})$. Rather, the *change* in $Q(\theta, \theta_k)$ lower bounds the *change* in $L_\theta(y_{1:T})$; c.f. (4.5). Nevertheless, with some abuse of

terminology, we will refer to $Q(\theta, \theta_k)$ as a lower bound, as shorthand for the relationship in (4.5).

4.2.4 Lagrangian relaxation

The technique of Lagrangian relaxation applies the MM principle to *constrained* optimization problems of the form

$$\min_{\theta,z} \; J(\theta, z) \text{ s.t. } F(\theta, z) = 0, \tag{4.6}$$

i.e. $f(\theta) = \min_z J(\theta, z)$ s.t. $F(\theta, z) = 0$. Here $J(\theta, z)$ is a cost function assumed to be convex in θ, and $F(\theta, z)$, assumed affine in θ, encodes the constraints. Notice that we present the problem as cost minimization, rather than objective maximization, and consequently develop upper bounds; however, this difference in superficial.

Unlike EM, in which we estimate z, Lagrangian relaxation *supremizes* over the latent variables to generate the bound. Specifically, the relaxation of (4.6) takes the form

$$\bar{J}_\lambda(\theta) = \sup_z \; J(\theta, z) - \lambda(z)' F(\theta, z), \tag{4.7}$$

where $\lambda(z)$ may be interpreted as a Lagrange multiplier. For arbitrary λ, the function $\bar{J}_\lambda(\theta)$ has two key properties:

1) It is convex in θ. Recall that J and F are convex and affine in θ, respectively. As such, $\bar{J}_\lambda(\theta)$ is the supremum of an infinite family of convex functions, and is therefore convex in θ; see §3.2.3 of [28].

2) It is an upper bound for the original problem (4.6). Given θ, let z^* be any z such that $F(\theta, z^*) = 0$. Then

$$J(\theta, z^*) + \lambda F(\theta, z^*) = J(\theta, z^*) \geq f(\theta),$$

which implies that the supremum over all z can be no smaller; i.e. $\bar{J}_\lambda(\theta)$ is an upper bound for $f(\theta)$.

The original optimization problem (4.6) may then be approximated by the convex program $\min_\theta \; \bar{J}_\lambda(\theta)$.

4.3 EM for linear dynamical systems

In the application of EM to the identification of dynamical systems, there are two natural choices of latent variables: systems states, $x_{1:T}$, and initial conditions and disturbances

$\{x_1, w_{1:T}\}$. In this section, we recap the latent states case, detail the latent disturbances formulation, and elucidate the key differences between the two.

The EM algorithm begins from an initial estimate of θ. As with any iterative method, it can be desirable to incorporate as much prior knowledge about the system as possible when initializing the algorithm. In this chapter, in the absence of prior information, we initialize with Lagrangian relaxation of simulation error [236], to ensure stability of the initial model.

4.3.1 EM with latent states

Latent states are the *de facto* choice of latent variables in the identification of dynamical systems. Consequently, this formulation has been studied extensively, c.f. [71]. Here we recap the essential details, to pave the way for the introduction of stability guarantees in §4.4.1. Choosing latent states yields a joint likelihood function of the form

$$p_\theta(y_{1:T}, x_{1:T}) = \left[\prod_{t=1}^{T} p_\theta(y_t|x_t)\right]\left[\prod_{t=1}^{T-1} p_\theta(x_{t+1}|x_t)\right] p_\theta(x_1). \tag{4.8}$$

The E step computes the *auxiliary* function,

$$Q^{\mathrm{s}}(\theta, \theta_k) = \mathrm{E}_{\theta_k}\left[\log p_\theta(y_{1:T}, x_{1:T}) \middle| y_{1:T}\right] \tag{4.9}$$

which decomposes as

$$\begin{aligned} Q^{\mathrm{s}}(\theta, \theta_k) = {} & \underbrace{\mathrm{E}_{\theta_k}\left[\log p_\theta(x_1) \middle| y_{1:T}\right]}_{\propto -Q_1^{\mathrm{s}}(\theta,\theta_k)} + \\ & \underbrace{\sum_{t=1}^{T} \mathrm{E}_{\theta_k}\left[\log p_\theta(y_t|x_t) \middle| y_{1:T}\right]}_{\propto -Q_2^{\mathrm{s}}(\theta,\theta_k)} + \underbrace{\sum_{t=1}^{T} \mathrm{E}_{\theta_k}\left[\log p_\theta(x_{t+1}|x_t) \middle| y_{1:T}\right]}_{\propto -Q_3^{\mathrm{s}}(\theta,\theta_k)} \end{aligned} \tag{4.10}$$

Notice that $-Q^{\mathrm{s}} \propto Q_1^{\mathrm{s}} + Q_2^{\mathrm{s}} + Q_3^{\mathrm{s}}$. It is more convenient to discuss maximization of Q^{s} in terms of minimization of $\sum_{i=1}^{3} Q_i^s$. As $-Q^{\mathrm{s}}$ is convex in θ, minimization is straightforward and reduces to linear least squares; c.f [71, Lemma 3.3]. Global minimizers of Q_1^{s}, Q_2^{s} and Q_3^{s} are given by

$$\mu = \hat{x}_{1|T}, \qquad \Sigma_1 = \widehat{\Sigma}_{1|T}, \tag{4.11a}$$

$$[C\ D] = \Phi_{yz}\Phi_{zz}^{-1}, \quad \Sigma_v = \Phi_{yy} - \Phi_{yz}\Phi_{zz}^{-1}\Phi_{yz}, \tag{4.11b}$$

$$[A\ B] = \Phi_{xz}\Phi_{zz}^{-1}, \quad \Sigma_w = \Phi_{xx} - \Phi_{xz}\Phi_{zz}^{-1}\Phi_{xz}, \tag{4.11c}$$

resp., where $z_t = [x_t', u_t']'$, $\Phi_{yy} = \frac{1}{T}\sum_{t=1}^{T} y_t y_t'$, and

$$\hat{x}_{1|T} = \mathrm{E}_{\theta_k}\left[x_1 \middle| y_{1:T}\right], \qquad \widehat{\Sigma}_{1|T} = \mathrm{Var}_{\theta_k}\left[x_1 | y_{1:T}\right], \tag{4.12}$$

$$\Phi_{yz} = \frac{1}{T}\sum_{t=1}^{T} \mathrm{E}_{\theta_k}\left[y_t z_t' \middle| y_{1:T}\right], \; \Phi_{xz} = \frac{1}{T}\sum_{t=1}^{T} \mathrm{E}_{\theta_k}\left[x_{t+1} z_t' \middle| y_{1:T}\right]$$
$$\Phi_{zz} = \frac{1}{T}\sum_{t=1}^{T} \mathrm{E}_{\theta_k}\left[z_t z_t' \middle| y_{1:T}\right], \; \Phi_{xx} = \frac{1}{T}\sum_{t=2}^{T+1} \mathrm{E}_{\theta_k}\left[x_t x_t' \middle| y_{1:T}\right].$$

The quantities in (4.12) can be computed by the RTS smoother [194]; c.f. [71, Lemma 3.2] for details. A numerically robust square-root implementation of the smoothing algorithm should be used for accuracy, e.g. [71, §4].

4.3.2 EM with latent disturbances

In the latent disturbances formulation of EM, it is convenient to work with the more general parametrization

$$x_{t+1} = Ax_t + Bu_t + Gw_t, \tag{4.13}$$

of LGSS model dynamics. This permits identification of singular state-space models, in which $n_w < n_x$, as discussed in §4.5.1. When using latent disturbances, we set $\Sigma_w = I$ and $\theta = \{\mu, \Sigma_1, \Sigma_v, A, B, C, D, G\}$. To avoid cumbersome notation, we use the same variable θ to group parameters in both the latent states and disturbances formulations; the contents of θ can be easily inferred from the context. Choosing latent disturbances yields a joint likelihood function of the form

$$p_\theta(y_{1:T}, x_1, w_{1:T}) = \left[\prod_{t=1}^{T} p_\theta(y_t | w_{1:t-1}, x_1)\right] p_\theta(w_{1:T}) p_\theta(x_1). \tag{4.14}$$

Analogously to (4.10), the auxiliary function

$$Q^{\mathrm{d}}(\theta, \theta_k) = \mathrm{E}_{\theta_k}\left[\log p_\theta(y_{1:T}, x_1, w_{1:T}) | y_{1:T}\right] \tag{4.15}$$

conveniently decomposes as

$$\begin{aligned} Q^{\mathrm{d}}(\theta, \theta_k) = &\underbrace{\mathrm{E}_{\theta_k}\left[\log p_\theta(x_1) | y_{1:T}\right]}_{\propto -Q_1^{\mathrm{d}}(\theta,\theta_k)} + \underbrace{\mathrm{E}_{\theta_k}\left[\log p_\theta(w_{1:T}) | y_{1:T}\right]}_{\propto -Q_2^{\mathrm{d}}(\theta_k)} \\ &+ \underbrace{\mathrm{E}_{\theta_k}\left[\log p_\theta(y_{1:T} | x_1, w_{1:T}) | y_{1:T}\right]}_{\propto -Q_3^{\mathrm{d}}(\theta,\theta_k)}. \end{aligned} \tag{4.16}$$

The following lemma details the computation of $Q^{\mathrm{d}}(\theta, \theta_k)$. For clarity of exposition, we introduce the following *lifted* form of the dynamics in (4.13),

$$Y = \bar{C}\bar{H}Z + (\bar{C}\bar{N} + \bar{D})U + V,$$

where $Y = \mathrm{vec}(y_{1:T})$, $U = \mathrm{vec}(u_{1:T})$, $V = \mathrm{vec}(v_{1:T})$, $Z = \mathrm{vec}([x_1, w_{1:T-1}])$,

$$\bar{H} = \begin{bmatrix} I & 0 & 0 & 0 & \dots & 0 \\ A & G & 0 & 0 & \dots & 0 \\ A^2 & A & G & 0 & \dots & 0 \\ \vdots & & & & \ddots & \vdots \\ A^{T-1} & A^{T-2}G & A^{T-3}G & & \dots & G \end{bmatrix},$$

$$\bar{N} = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ B & 0 & 0 & 0 & \dots & 0 \\ AB & B & 0 & 0 & \dots & 0 \\ \vdots & & & & \ddots & \vdots \\ A^{T-2}B & A^{T-3}B & \dots & AB & B & 0 \end{bmatrix},$$

$$\bar{C} = I_T \otimes C \text{ and } \bar{D} = I_T \otimes D. \tag{4.17}$$

Lemma 4.1. The auxiliary function $Q^{\mathrm{d}}(\theta, \theta_k)$ defined in (4.15) is given by

$$\begin{aligned} -Q^{\mathrm{d}}(\theta, \theta_k) \propto & \log\det\Sigma_1 + |\hat{x}_{1|T} - \mu|^2_{\Sigma_1^{-1}} + \mathrm{tr}\big(\Sigma_1^{-1}\widehat{\Sigma}_{1|T}\big) + \\ & T\log\det\Sigma_v + \mathrm{tr}(\Sigma_Y^{-1}(\bar{C}\bar{H}\Omega\bar{H}'\bar{C}' + \hat{\Delta}\hat{\Delta}')), \end{aligned}$$

where $\hat{x}_{1|T} = \mathrm{E}_{\theta_k}\left[x_1|y_{1:T}\right]$ and $\widehat{\Sigma}_{1|T} = \mathrm{Var}_{\theta_k}\big[x_1|y_{1:T}\big]$ as in (4.12), and

$$\hat{Z} = \mathrm{E}_{\theta_k}\left[Z|y_{1:T}\right], \tag{4.18a}$$

$$\Omega = \mathrm{Var}_{\theta_k}\left[Z|y_{1:T}\right], \tag{4.18b}$$

$$\mu_Y := \mathrm{E}_\theta\big[Y|Z\big] = \bar{C}\bar{H}Z + (\bar{C}\bar{N} + \bar{D})U, \tag{4.19a}$$

$$\Sigma_Y := \mathrm{Var}_\theta\big[Y|Z\big] = I_T \otimes \Sigma_v, \tag{4.19b}$$

$$\hat{\Delta} = \mathrm{E}_{\theta_k}\big[Y - \mu_Y|y_{1:T}\big] = Y - \bar{C}\bar{H}\hat{Z} - (\bar{C}\bar{N} + \bar{D})U. \tag{4.20}$$

Proof. Refer to 4.8.1. □

For the LGSS models considered in this work, $\hat{Z}$ and Ω can be computed in closed form by standard *disturbance* smoothers; see, e.g., [53, §4.5]. Once again, it is prudent to use square-root implementations of these smoothing algorithms, given in [53, §6.3], for numerical robustness (i.e. nonnegative definiteness of covariances).

We now turn our attention to the M step, i.e. minimization of $-Q^{\mathrm{d}} \propto Q_1^{\mathrm{d}} + Q_2^{\mathrm{d}} + Q_3^{\mathrm{d}}$. It is clear that Q_1^{d} and Q_1^{s} take the same form, so $\mu = \hat{x}_{1|T}$ and $\Sigma_1 = \widehat{\Sigma}_{1|T}$ globally minimize $Q_1^{\mathrm{d}}(\theta, \theta_k)$, as in (4.11a). Minimization of $Q_2^{\mathrm{d}}(\theta_k)$ is unnecessary, as it is constant w.r.t. θ. Minimization of Q_3^{d}, however, is a more challenging problem. Indeed, from Lemma 4.1, it

is clear that the quantities $\bar{H}$ and $\bar{N}$ render $Q_3^{\mathrm{d}}(\theta, \theta_k)$ a nonconvex function of the model parameters.

To summarize, the computations involved in each iteration of the latent disturbances formulation of EM are straightforward, with the exception of minimization of $Q_3^{\mathrm{d}}(\theta, \theta_k)$, which is nonconvex.

A common heuristic for terminating the EM algorithm is to cease iterations once the change in likelihood falls below a certain tolerance δ, i.e.

$$L_{\theta_{k+1}}(y_{1:T}) - L_{\theta_k}(y_{1:T}) < \delta. \tag{4.21}$$

Alternatively, one can simply run the algorithm for a finite number of iterations, chosen so as to attain a model of sufficient quality; this is the approach taken, e.g., in [71, 258].

4.4 Convex M step with guaranteed model stability

In this section we incorporate model stability constraints into the EM framework. The set $\mathcal{S}$ of stable A matrices is nonconvex; however, by using Lagrangian relaxation we build convex bounds on $-Q(\theta, \theta_k)$, which we optimize over a convex parametrization of all stable linear models.

4.4.1 Ensuring stability with latent states

We begin with the latent states formulation. The global minimizers (4.11a) and (4.11b) of Q_1^{s} and Q_2^{s}, resp., remain unchanged, as they do not influence stability. To optimize Q_3^{s}, it is convenient to work with the representation:

$$Q_3^{\mathrm{s}}(\theta, \theta_k) = \sum_{t=1}^{M^{\mathrm{s}}} |\tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t|^2_{\Sigma_w^{-1}} + T \log\det \Sigma_w \tag{4.22}$$

where $M^{\mathrm{s}} = 2n_x + n_u$, and $\tilde{x}$ and $\tilde{u}$ satisfy

$$\sum_{t=1}^{M^{\mathrm{s}}} \begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{x}_t \\ \tilde{u}_t \end{bmatrix} \begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{x}_t \\ \tilde{u}_t \end{bmatrix}' = T \begin{bmatrix} \Phi_{xx} & \Phi_{xz}' \\ \Phi_{xz} & \Phi_{zz} \end{bmatrix} = \Phi^{\mathrm{s}}. \tag{4.23}$$

Our goal is to minimize Q_3^{s} subject to model stability constraints. The main challenge is nonconvexity of the set $\mathcal{S}$ of (Schur) stable matrices, i.e., the Lyapunov condition $A'PA - P < 0$ is not jointly convex in A and $P \in \mathbb{S}_{++}^{n_x}$. One can circumvent this difficulty by introducing an equivalent *implicit* representation of the dynamics, e.g.,

$$Ex_{t+1} = Fx_t + Ku_t. \tag{4.24}$$

In what follows, let $\theta^s = \{E, F, K, \Sigma_w\}$.

Lemma 4.2. A matrix $A \in \mathbb{R}^{n_x \times n_x}$ is Schur stable iff there exists $E \in \mathbb{R}^{n_x \times n_x}$ and $P \in \mathbb{S}^{n_x}_{++}$ such that the LMI

$$S^{\mathrm{s}}(\theta^s) := \begin{bmatrix} E + E' - P - I & F' \\ F & P \end{bmatrix} \succeq 0 \tag{4.25}$$

holds with $F = EA$.

Proof. This result is a trivial modification of Lemma 4 and Corollary 5 in [141, Section 3.2]. □

By Lemma 4.2, $\Theta^s = \{\theta^s : \exists P \in \mathbb{S}^{n_x}_{++},\ S^{\mathrm{s}}(\theta^s) \succeq 0\}$ defines a convex parametrization of all stable linear systems. Note also that (4.25) implies $E + E' \succ 0$, which ensures that the implicit dynamics in (4.24) are well-posed, i.e. $A = E^{-1}F$.

The challenge now becomes the optimization of Q^{s}_3 with models in the implicit form (4.24). Simply solving

$$\min_{\theta^s \in \Theta^s} \sum\nolimits_{t=1}^{M^{\mathrm{s}}} |E\tilde{x}_{t+1} - F\tilde{x}_t - K\tilde{u}_t|^2_{\Sigma_w^{-1}} + T \log\det \Sigma_w$$

is insufficient, as there is no guarantee that this will reduce Q^{s}_3. We proceed by using Lagrangian relaxation to build a convex upper bound on Q^{s}_3. For clarity of exposition, let us temporarily ignore the $\log\det\Sigma_w$ term, as well as the summation, in (4.22) and consider the nonconvex problem

$$\min_{A,B,\Sigma_w} \ |\tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t|^2_{\Sigma_w^{-1}} \quad \text{s.t. } A \in \mathcal{S} \tag{4.26}$$

for some t. Problem (4.26) is completely equivalent to

$$\min_{x_{t+1}, \theta^s \in \Theta^s} \ |\tilde{x}_{t+1} - x_{t+1}|^2_{\Sigma_w^{-1}} \text{ s.t. } Ex_{t+1} = F\tilde{x}_t + K\tilde{u}_t \tag{4.27}$$

as both (4.26) and (4.27) have the same objective and feasible set. Introducing $\Delta = \tilde{x}_{t+1} - x_{t+1}$ and $\epsilon_t = E\tilde{x}_{t+1} - F\tilde{x}_t - K\tilde{u}_t$, the Lagrangian relaxation of (4.27) is given by

$$\bar{J}^s_\lambda(\theta^s, t) = \sup_{\Delta} \ |\Delta|^2_{\Sigma_w^{-1}} - \lambda(\Delta)' \left(E\Delta - \epsilon_t\right) \tag{4.28}$$

for some multiplier $\lambda(\Delta)$, c.f. Section 4.2.4. As $\bar{J}^s_\lambda(\theta^s, t)$ upper bounds (4.26), we can construct a convex upper bound for Q^{s}_3 by combining $\sum_t \bar{J}^s_\lambda(\theta^s, t)$ with a linear bound on the concave $\log\det\Sigma_w$ term.[1]

Lemma 4.3. Consider the function

$$\bar{Q}^{\mathrm{s}}_3(\theta^s) := \sum\nolimits_{t=1}^{M^{\mathrm{s}}} \bar{J}^s_\lambda(\theta^s, t) + T\mathrm{tr}\left(\Sigma_{w_k}^{-1}\Sigma_w\right), \tag{4.29}$$

[1] The linear bound on $\log\det\Sigma_w$ is $\mathrm{tr}\left(\Sigma_{w_k}^{-1}\Sigma_w\right) + \log\det\Sigma_{w_k} + n_x$, but we exclude the constant terms from (4.29) for brevity.

where Σ_{w_k} is the estimate of Σ_w stored in θ_k. $\bar{Q}_3^{\mathrm{s}}(\theta^s)$ is a convex upper bound on $Q_3^{\mathrm{s}}(\theta, \theta_k)$.

The function $\bar{Q}_3^{\mathrm{s}}$ is a convex upper bound on Q_3^{s} for any multiplier, λ. However, to be suitable for EM, i.e. for (4.5) to hold, we require $\bar{Q}_3^{\mathrm{s}}$ to be tight at θ_k, i.e. $\bar{Q}_3^{\mathrm{s}}(\theta_k^s) = Q_3^{\mathrm{s}}(\theta_k, \theta_k)$. The following lemma provides a choice of multiplier that ensures this property.

Lemma 4.4. For each $\bar{J}_\lambda^s(\theta_k^s, t)$ in (4.29), $t = 1, \ldots, M^{\mathrm{s}}$, let $\lambda(\Delta) = 2H\Delta$ where $H = (P')^{-1}\Sigma_w^{-1}$ is such that $\theta_k^s = \{P, PA_k, PB_k, \Sigma_{w_k}\} \in \Theta^s$. Then $\bar{Q}_3^{\mathrm{s}}(\theta_k^s) = Q_3^{\mathrm{s}}(\theta_k, \theta_k)$, i.e. the bound is tight at θ_k.

Proof. Refer to 4.8.2. □

Before leaving the latent states case we note that the upper bound $\bar{Q}_3^{\mathrm{s}}$ can be optimized as the following SDP:

$$\begin{aligned} \min_{R, \theta^s \in \Theta^s} \quad & \operatorname{tr}(R\Phi^{\mathrm{s}}) + T\operatorname{tr}\left(\Sigma_{w_k}^{-1}\Sigma_w\right) \\ \text{s.t.} \quad & \begin{bmatrix} R & E_K^{F'}H & 0 \\ H'E_K^F & H'E + E'H & I \\ 0 & I & \Sigma_w \end{bmatrix} \succeq 0, \end{aligned} \tag{4.30}$$

where $R \in \mathbb{S}^{2n_x+n_u}$ is a slack variable, $E_K^F = [E, -F, -K]$, and Φ^{s} is the empirical covariance matrix in (4.23). A complete summary of the approach is given in Algorithm 4.

Remark 4.2. To ensure model stability, it is only necessary to solve (4.30) if the spectral radius of A_{ls} is too large, where A_{ls} is the least squares solution from (4.11c), i.e., if $r_{\mathrm{sp}}(A_{\mathrm{ls}}) > 1 - \delta$ for some-user selected $\delta > 0$, c.f. (3.2) of Algorithm 4.

Algorithm 4 EM with latent States and Lagrangian relaxation (EMSL)

1. Set $k = 0$ and initialize θ_k such that $A_k \in \mathcal{S}$ and $L_{\theta_k}(y_{1:T})$ is finite; c.f. §4.3.
2. **Expectation (E) Step:** compute (4.12).
3. **Maximization (M) Step:**
   (3.1) Update θ_{k+1} with least squares, as in (4.11).
   (3.2) If $r_{\mathrm{sp}}(A_{k+1}) > 1 - \delta$ for user-chosen $\delta > 0$, solve $\theta_{k+1}^s = \arg\min_{\theta^s \in \Theta^s} \bar{Q}_3^{\mathrm{s}}(\theta^s)$, and update θ_{k+1} with $A_{k+1} = E_{k+1}^{-1}F_{k+1}$, $B_{k+1} = E_{k+1}^{-1}K_{k+1}$, and $\Sigma_{w_{k+1}}$
4. Evaluate termination criteria, e.g. (4.21). If `false`, $k \leftarrow k + 1$ and return to step 2.

4.4.2 Convex bounds with stability guarantees for latent disturbances

We now turn our attention to the latent disturbances formulation. The developments in this section follow the same pattern as §4.4.1, however, the computations are more involved. As in the latent states case, the global minimizer (4.11a) of Q_1^{d} remains unchanged, and so we concentrate on optimization of Q_3^{d} subject to a model stability constraint, $A \in \mathcal{S}$. It is convenient to conceptualize $Q_3^{\mathrm{d}}(\theta, \theta_k)$ in terms of *simulation error*, defined as

$$\mathcal{E}(\theta, U, Y, Z) := \sum_{t=1}^{T} |y_t - Cx_t - Du_t|^2_{\Sigma_v^{-1}}, \tag{4.31}$$

where $\mathrm{vec}(x_{1:T}) = \bar{N}U + \bar{H}Z$, i.e., the simulated states.

Lemma 4.5. With simulation error defined as in (4.31), $Q_3^{\mathrm{d}}(\theta, \theta_k)$ in (4.16) is equivalent to:

$$Q_3^{\mathrm{d}}(\theta, \theta_k) = \mathcal{E}(\theta, U, Y, \hat{Z}) + \sum_{j=1}^{M^{\mathrm{d}}} \mathcal{E}(\theta, 0, 0, Z^j) + T \log\det \Sigma_v \tag{4.32}$$

where $M^{\mathrm{d}} = n_x + (T-1)n_w$, and $Z^j \in \mathbb{R}^{n_x + (T-1)n_w}$ are such that $\sum_{j=1}^{M^{\mathrm{d}}} ZZ^{j'} = \Omega$.

Proof. Refer to 4.8.3. □

Our task $\min_\theta Q_3^{\mathrm{d}}$ s.t. $A \in \mathcal{S}$ is challenging due to nonconvexity of both the objective and feasible set. As in §4.4.1, we circumvent the latter by introducing an implicit representation of the dynamics in (4.13),

$$Ex_{t+1} = Fx_t + Ku_t + Lw_t. \tag{4.33}$$

Setting $\theta^d = \{E, F, K, L, \Sigma_v\}$ we can define the convex set of stable models $\Theta^d = \{\theta^d : \exists P \in \mathbb{S}^{n_x}_{++},\ S^{\mathrm{d}}(\theta^d) \succeq 0\}$, with

$$S^{\mathrm{d}}(\theta^s) := \begin{bmatrix} E + E' - P - \delta I & F' & C' \\ F & P & 0 \\ C & 0 & \Sigma_w \end{bmatrix} \succeq 0, \tag{4.34}$$

for $\delta > 0$. We use the LMI (4.34), instead of (4.25), to ensure finiteness of the supremum in (4.37), c.f. 4.8.5.

To optimize Q_3^{d} with models in the implicit form (4.33), we use Lagrangian relaxation to build a convex upper bound on Q_3^{d}. For clarity of exposition, let us temporarily ignore the

$\log\det\Sigma_v$ term, and summation, in (4.32) and concentrate on minimization of simulation error

$$\min_{\theta}\ \mathcal{E}(\theta, U, Y, Z) \text{ s.t. } A \in \mathcal{S}. \tag{4.35}$$

Problem (4.35) is completely equivalent to

$$\min_{\Delta, \theta^d \in \Theta^d}\ |Y - \bar{C}\Delta - \bar{D}U|^2_{\Sigma_Y^{-1}} \text{ s.t. } \bar{E}\Delta = \bar{\epsilon}, \tag{4.36}$$

as both problems have the same objective and feasible set. In (4.36), $\Delta \in \mathbb{R}^{Tn_x}$ denotes the states $x_{1:T}$ that we optimize over, $\bar{E} \in \mathbb{R}^{Tn_x \times Tn_x}$ and $\bar{\epsilon} \in \mathbb{R}^{Tn_x}$ are given by

$$\begin{bmatrix} E & 0 & \dots & & \\ -F & E & 0 & & \\ 0 & -F & E & 0 & \\ \vdots & & & \ddots & \ddots \end{bmatrix} \ \& \ \begin{bmatrix} Ex_1 \\ Ku_1 + Lw_1 \\ \vdots \\ Ku_{T-1} + Lw_{T-1} \end{bmatrix}$$

resp., and $\bar{C}$, $\bar{D}$, Σ_Y are defined in (4.17). The Lagrangian relaxation of (4.36) is given by

$$\bar{J}^d_\lambda(\theta^d, U, Y, Z) = \sup_{\Delta}\ |Y - \bar{C}\Delta - \bar{D}U|^2_{\Sigma_Y^{-1}} - \lambda(\Delta)'\left(\bar{E}\Delta - \bar{\epsilon}\right) \tag{4.37}$$

for some multiplier $\lambda(\Delta) \in \mathbb{R}^{Tn_x}$, c.f. §4.2.4. As $\bar{J}^d_\lambda$ upper bounds (4.26), we can construct a convex upper bound for Q^{d}_3 by replacing each simulation error term in (4.32) with the appropriate bound:

Lemma 4.6. Consider the following function

$$\begin{aligned} \bar{Q}^{\mathrm{d}}_3(\theta^d) := \bar{J}_{\lambda^0}(\theta^d, U, Y, \hat{Z}) + \sum_{j=1}^{M^{\mathrm{d}}} \bar{J}_{\lambda^j}(\theta^d, 0, 0, Z^j) \\ + T\mathrm{tr}\left(\Sigma^{-1}_{v_k}\Sigma_v\right), \end{aligned} \tag{4.38}$$

where $\sum_{j=1}^{M^{\mathrm{d}}} Z^j Z^{j'} = \Omega$ as in Lemma 4.5, and Σ_{v_k} is the estimate of Σ_v stored in θ_k. $\bar{Q}^{\mathrm{d}}_3(\theta^d)$ is a convex upper bound for $Q^{\mathrm{d}}_3(\theta, \theta_k)$.

Proof. Refer to 4.8.4. □

Notice, from (4.38), that $\bar{Q}^{\mathrm{d}}_3$ depends on $M^{\mathrm{d}} + 1$ multipliers, $\{\lambda^j\}_{j=0}^{M^{\mathrm{d}}}$, unlike $\bar{Q}^{\mathrm{s}}_3$. Although $\bar{Q}^{\mathrm{d}}_3$ upper bounds Q^{d}_3 for any choice of λ, as in §4.4.1 we require $\bar{Q}^{\mathrm{d}}_3$ to be a tight bound such that (4.5) holds, i.e., we need $\bar{Q}^{\mathrm{d}}_3(\theta^d_k) = Q^{\mathrm{d}}_3(\theta_k, \theta_k)$. To obtain such a set of multipliers $\{\lambda^j\}_{j=0}^{M^{\mathrm{d}}}$, we propose the following two-stage approach. At the kth iteration,

i. For each of the $j = 0, \dots, M^{\mathrm{d}}$ bounds $\bar{J}_{\lambda^j}(\theta^d)$ that comprise $\bar{Q}^{\mathrm{d}}_3(\theta^d)$, c.f. (4.38), solve

the convex program

$$\begin{aligned} E_k^j = \arg\min_{E} \quad & \bar{J}_{\lambda_\Delta^j}(\theta^d) \\ \text{s.t.} \quad & \theta^d = \{E, EA_k, EB_k, EG_k, \Sigma_{v_k}\} \in \Theta^d \end{aligned}$$

where $\lambda_\Delta^j = 2\Delta$.

ii. Set $\lambda^j = 2\left(\Delta + h^j\right)$ with $h^j \in \mathbb{R}^{Tn_x}$ given by

$$h^j = (\bar{E}_k^{j\prime})^{-1}\left(\Psi_k^j(\bar{E}_k^j)^{-1}\bar{\epsilon}_k^j + \bar{\epsilon}_k^j - \bar{C}_k'\bar{\Sigma}_{Y,k}^{-1}(Y - \bar{D}_k U)\right) \tag{4.39}$$

where $\Psi_k^j = \bar{C}_k'\bar{\Sigma}_{Y,k}^{-1}\bar{C}_k - \bar{E}_k^j - \bar{E}_k^{j\prime}$. Here $\bar{E}_k^j$ and $\bar{\epsilon}_k^j$ denote $\bar{E}$ and $\bar{\epsilon}$, resp., built with $E = E_k^j$, $F = E_k^j A_k$, $K = E_k^j B_k$, and $L = E_k^j G_k$. $\bar{C}_k$ and $\bar{D}_k$ denote $\bar{C}$ and $\bar{D}_k$, resp., built with $C = C_k$, $D = D_k$.

The following lemma guarantees that the multipliers generated by this two-stage procedure give a 'tight' bound:

Lemma 4.7. Given $\theta_k^d \in \Theta^d$, let $\{\lambda^j\}_{j=0}^{M^{\mathrm{d}}}$ in (4.38) take the form $\lambda(\Delta)^j = 2(\Delta + h^j)$ with h^j given by (4.39). Then $\bar{Q}_3^{\mathrm{d}}(\theta_k^d) = Q_3^{\mathrm{d}}(\theta_k, \theta_k)$, i.e. the bound is tight at θ_k.

Proof. Refer to 4.8.5. □

A complete summary of the latent disturbances approach to EM with stability constraints is given in Algorithm 5.

Remark 4.3. This EMDL formulation includes, as a special case, models in innovations form, c.f. [129, §4.3]. For such models, innovations replace disturbances in (4.1a) and the latent variables reduce to the initial state, x_1. EM in this setting was studied in [258]. The difference between [258] and our approach is the M step: in [258] $Q(\theta, \theta_k)$ is optimized directly with a quasi-Newton method; we optimize a convex upper bound on $-Q(\theta, \theta_k)$ over a convex parametrization of stable models.

4.4.3 Correlated disturbances and measurement noise

For clarity of exposition, we have considered models in which there is no correlation between disturbances and measurement noise. However, the methods we have presented readily extend to the correlated case, i.e.

$$\begin{bmatrix} w_t \\ v_t \end{bmatrix} \sim \mathcal{N}\left(0, \Sigma_{s_c}\right), \quad \Sigma_{s_c} = \begin{bmatrix} \Sigma_w & \Sigma_{wv} \\ \Sigma_{wv}' & \Sigma_v \end{bmatrix}. \tag{4.40}$$

Algorithm 5 EM with latent Disturbances and Lagrangian relaxation (EMDL)

1. Set $k = 0$ and initialize θ_k such that $A_k \in \mathcal{S}$ and $L_{\theta_k}(y_{1:T})$ is finite; c.f. §4.3.
2. **Expectation (E) Step:**
   (2.1) Compute $\hat{x}_{1|T}$ and $\widehat{\Sigma}_{1|T}$ as in (4.12).
   (2.2) Compute $\hat{Z}$ and Ω as in (4.18).
3. **Maximization (M) Step:**
   (3.1) Update $\{\mu, \Sigma_1\}_{k+1} = \{\hat{x}_{1|T}, \widehat{\Sigma}_{1|T}\}$.
   (3.2) Assemble $\{\lambda^j\}_{j=0}^{M^{\mathrm{d}}}$ of the form $\lambda^j = 2(\Delta + h^j)$ by computing $\{h^j\}_{j=0}^{M^{\mathrm{d}}}$ with (4.39).
   (3.3) Obtain $\theta^d_{k+1} = \arg\min_{\theta^d \in \Theta^d} \bar{Q}^{\mathrm{d}}_3(\theta^d)$.
   (3.4) Update θ_{k+1} with $A_{k+1} = E^{-1}_{k+1} F_{k+1}$, $B_{k+1} = E^{-1}_{k+1} K_{k+1}$, $G_{k+1} = E^{-1}_{k+1} L_{k+1}$, and $\Sigma_{v_{k+1}}$.
4. Evaluate termination criteria, e.g. (4.21). If `false`, $k \leftarrow k + 1$ and return to step 2.

With latent states, the joint likelihood becomes

$$p_\theta(y_{1:T}, x_{1:T}) = \left[\prod_{t=1}^{T} p_\theta(y_t, x_{t+1}|x_t)\right] p_\theta(x_1),$$

with $-Q^{\mathrm{s}}(\theta, \theta_k) \propto Q_1^{\mathrm{s}}(\theta, \theta_k) + Q_{\mathrm{c}}^{\mathrm{s}}(\theta, \theta_k)$ where

$$\begin{aligned} -Q_{\mathrm{c}}^{\mathrm{s}}(\theta, \theta_k) \propto & \sum_{t=1}^{M_{\mathrm{c}}^{\mathrm{s}}} \left| \begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{y}_t \end{bmatrix} - \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \tilde{x}_t \\ \tilde{u}_t \end{bmatrix} \right|^2_{\Sigma_{s_c}^{-1}} \\ & + T \log\det \Sigma_{s_c}. \end{aligned} \tag{4.41}$$

Here $M_{\mathrm{c}}^{\mathrm{s}} = 2n_x + n_y + n_u$, and $\tilde{x}$, $\tilde{y}$, $\tilde{u}$ satisfy

$$\sum_{t=1}^{M_{\mathrm{c}}^{\mathrm{s}}} \tilde{\zeta}_t \tilde{\zeta}_t' = \sum_{t=1}^{T} \mathrm{E}_{\theta_k} \left[\zeta_t \zeta_t' \middle| y_{1:T} \right],$$

where $\tilde{\zeta}_t = \left[\tilde{x}_{t+1}', \tilde{y}_t', \tilde{x}_t', \tilde{u}_t'\right]'$ and $\zeta_t = \left[x_{t+1}', y_t', x_t', u_t'\right]'$. Clearly, (4.41) has the same form as (4.22), and so the Lagrangian relaxation of §4.4.1 is applicable.

Similarly, in the latent disturbances formulation the joint likelihood can be factorized as

$$p_\theta(y_{1:T}, x_1, w_{1:T}) = \prod_{t=1}^{T} p_\theta(y_t, w_t|x_1, w_{1:t-1}) p(x_1),$$

with $p(y_t, w_t|x_1, w_{1:t-1})$ given by

$$\mathcal{N}\left(\begin{bmatrix} Cx_t + Du_t \\ 0 \end{bmatrix}, \Sigma_{s_d} \right), \quad \Sigma_{s_d} = \begin{bmatrix} \Sigma_v & \Sigma_{wv} \\ \Sigma_{wv}' & I \end{bmatrix},$$

where $\mathrm{vec}(x_{1:T}) = \bar{N}U + \bar{H}Z$. Introducing $y_t^c = [y_t', w_t']'$, $C^c = [C', 0']'$, and $D^c = [D', 0']'$ we have

$$\begin{aligned} & \log p_\theta(y_{1:T}, w_{1:T}|x_1) \propto \\ & \sum_{t=1}^{T} |y_t^c - C^c x_t - D^c u_t|^2_{\Sigma_{s_d}^{-1}} + T \log\det \Sigma_{s_d}. \end{aligned} \tag{4.42}$$

The resemblance to (4.31) is apparent. Computing the expected value of (4.42) leads to a quantity that takes the same form as Q_3^{d} in (4.32), to which the Lagrangian relaxation of §4.4.2 is applicable.

4.5 On the choice of latent variables

4.5.1 Singular state space models

In applications it may be known *a priori* that the dimension of the disturbance is less than that of the state variable, i.e. $n_w < n_x$. For example, consider a mechanical system in which the disturbances are forces or torques. There are typically fewer disturbance forces than state variables (as force directly affects acceleration, but not position or velocity, in continuous time dynamics), and so G is rank-deficient. As the transition density $p_\theta(x_{t+1}|x_t) = \mathcal{N}(Ax_t + Bu_t, GG')$ no longer admits a closed form representation, when GG' is singular, it is well known that the standard EM algorithms based on latent states are no longer directly applicable.

Modifications of the EM algorithm have been proposed to circumvent this difficulty. The work of [214] introduced a perturbation model with full-rank process noise covariance, and proved that the EM iterations remain well behaved when the perturbation is set to zero. However, this approach was restricted to models in which the disturbances and measurement noise are uncorrelated. A subsequent paper [215] addressed the case of correlated state and measurement noise, but only considered models in innovations form; extension to the case of models in general form was left to future work. Furthermore, this method requires the variance of the initial state (i.e. Σ_1) to be excluded from the estimated parameters, θ.

The latent disturbances formulation of EM, c.f. §4.3.2, provides the most general solution to the difficulties associated with singular state space models. Specifically, with latent disturbances we can handle rank deficient G, with the possibility of correlated state and measurement noise (c.f. §4.4.3), as well as unknown initial conditions (μ, Σ_1), for models not necessarily in innovations form, c.f. Remark 4.3. When using latent disturbances we work with the joint likelihood function $p_\theta(y_{1:T}, x_1, w_{1:T})$, given in (4.14). Comparing (4.14) to (4.8), we observe that the problematic transition density is replaced by the joint distribution of disturbances $p_\theta(w_{1:T})$ which remains well-defined in the singular case, $n_w < n_x$.

4.5.2 Absence of disturbances or measurement noise

In this section, we study the auxiliary function $Q(\theta, \theta_k)$ in the limit cases of $G = 0$ and $\Sigma_v = 0$, for different choices of latent variables. These results will offer insight into the behavior of the EM algorithm as a function of disturbance magnitude, which is explored in §4.5.3.

Proposition 4.1. Consider a model of the form (4.1), and let θ be such that $\Sigma_w = 0$, i.e. disturbances are omitted from the model. The auxiliary function built on latent states, $Q^{\mathrm{s}}(\theta, \theta_k)$, is undefined when $A \neq A_k$ or $B \neq B_k$.

Proof. Refer to 4.8.6. □

Proposition 4.2. Consider a model of the form (4.1), with dynamics of the form (4.13), and let θ be such that $G = 0$, i.e. disturbances are omitted from the model. Furthermore, suppose $\Sigma_1 = 0$; i.e. the initial conditions $x_1 = \mu$ are modeled without uncertainty. Then $L_\theta(y_{1:T}, x_1) = Q^{\mathrm{d}}(\theta, \theta_k)$ for all θ, θ_k; i.e., the auxiliary function built on latent disturbances, $Q^{\mathrm{d}}(\theta, \theta_k)$, reduces to the log likelihood.

Proof. Refer to 4.8.7. □

Proposition 4.3. Consider a first order model of the form (4.1), and let θ be such that $\Sigma_v = 0$, i.e. output noise is omitted from the model. The auxiliary function built on latent disturbances, $Q^{\mathrm{d}}(\theta, \theta_k)$, is undefined for $\theta \neq \theta_k$, i.e. the domain of $Q(\theta, \theta_k)$ collapses to a single point, $\theta = \theta_k$.

Proof. Refer to 4.8.8. □

Proposition 4.4. Consider a first order model of the form (4.1), and let θ be such that $\Sigma_v = 0$, i.e. output noise is omitted from the model. Let $Q^{\mathrm{s}}(\theta, \theta_k)$ denote the auxiliary function built on latent states, then:

i. $Q^{\mathrm{s}}(\theta, \theta_k)$ is undefined for all θ such that $C \neq C_k$ or $D \neq D_k$.

ii. $Q^{\mathrm{s}}(\theta, \theta_k) = L_\theta(y_{1:T})$ for all θ such that $C = C_k$ and $D = D_k$.

Proof. Refer to 4.8.9. □

4.5.3 Influence of disturbance magnitude on bound fidelity

In this section, we empirically investigate the fidelity of $Q(\theta, \theta_k)$ as a bound on $L_\theta(y_{1:T})$, as a function of the magnitude of the disturbances, $w_{1:T}$, and the choice of latent variables. The results are presented in Figure 4.1, which depicts Q^{s}, Q^{d} and $L_\theta(y_{1:T})$ for a first order ($n_x = 1$) LGSS model, with $A = 0.7$, $B = 0.3$, $C = 0.1$, $D = 0.01$, and $GG' = \Sigma_w$. Each bound, Q^{s} and Q^{d}, is plotted as a function of the single unknown scalar parameter $\theta = A$. Note that $\mathcal{S} = \{A : -1 < A < 1\}$ is convex for $n_x = 1$; this is not true for $n_x > 1$.

We begin with the case of 'small' disturbances (i.e. $\Sigma_w \ll \Sigma_v$) as depicted in Figure 4.1(a), and observe the following: $Q^{\mathrm{d}}(\theta, \theta_k)$ represents $L_\theta(y_{1:T})$ with high fidelity, whereas $Q^{\mathrm{s}}(\theta, \theta_k)$ is localized about θ_k. Such an observation is not without precedent. For instance, in the latent states formulation of [203, Section 10] it was noted that an initial disturbance covariance estimate $\Sigma_w = 0$ results in $\theta_k = \theta_0$ for all k; i.e. the model parameters are not improved. This suggests that $Q^{\mathrm{s}}(\theta, \theta_k)$ fails to accurately represent $L_\theta(y_{1:T})$, except at $\theta = \theta_0$.

Proposition 4.1 makes this observation more precise: in the 1D case of Figure 4.1(a), when $\Sigma_w = 0$, $Q^{\mathrm{s}}(\theta, \theta_k)$ is undefined for $A \neq A_k$. Taken together, Figure 4.1(a) and Proposition

4.1 suggest that as Σ_w becomes smaller (relative to Σ_v) the bound $Q^{\mathrm{s}}(\theta, \theta_k)$ becomes more localized about θ_k; the domain collapses to a single point, $\theta = \theta_k$, when $\Sigma_w = 0$. Conversely, as Σ_w (and Σ_1) decrease, $Q^{\mathrm{d}}(\theta, \theta_k)$ becomes an increasingly accurate representation of the log likelihood, eventually reproducing $L_\theta(y_{1:T})$ *exactly*, when Σ_w (and Σ_1) are identically zero, as in Proposition 4.2.

Turning our attention to the case of 'large' disturbances (i.e. $\Sigma_w \gg \Sigma_v$) as depicted in Figure 4.1(b), we observe the opposite behavior: $Q^{\mathrm{s}}(\theta, \theta_k)$ faithfully represents the log likelihood, whereas $Q^{\mathrm{d}}(\theta, \theta_k)$ appears to be localized about θ_k. Once more, studying the limiting case $\Sigma_v = 0$ offers insight into this behavior: Proposition 4.3 states that when $\Sigma_v = 0$, $Q^{\mathrm{d}}(\theta, \theta_k)$ is undefined for $A \neq A_k$. Taken together, Figure 4.1(b) and Proposition 4.3 suggest that as Σ_v decreases (i.e. as Σ_w increases relative to Σ_v), the bound $Q^{\mathrm{d}}(\theta, \theta_k)$ becomes more localized about θ_k; the domain collapses to a single point, $\theta = \theta_k$, when $\Sigma_v = 0$. Conversely, for this 1D experiment with $\theta = A$, Proposition 4.4 states that $Q^{\mathrm{s}}(\theta, \theta_k)$ will reproduce $L_\theta(y_{1:T})$ *exactly*, when Σ_v is identically zero. Indeed, in Figure 4.1(b) with $\Sigma_v \ll \Sigma_w$, we observe $Q^{\mathrm{s}}(\theta, \theta_k)$ representing the likelihood faithfully.

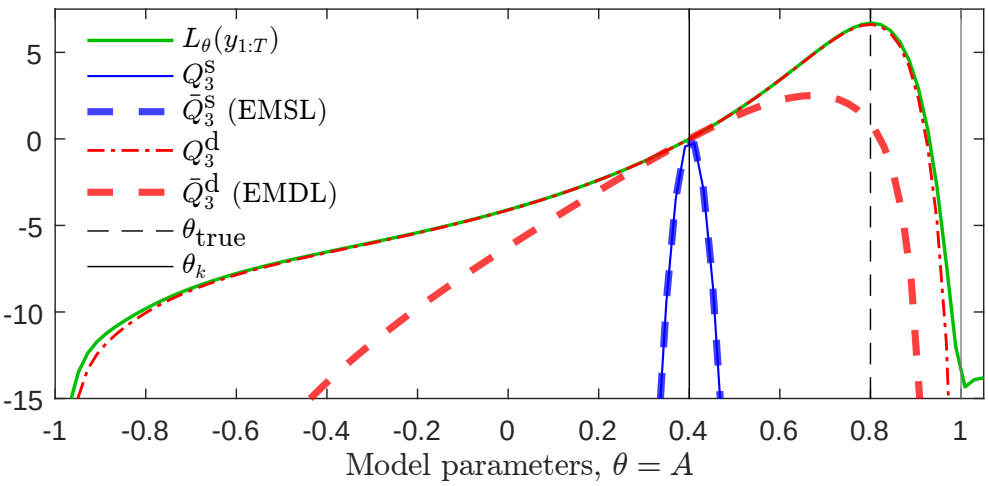
To summarize: in the case of 'small disturbances' (i.e. $\Sigma_w \ll \Sigma_v$), $Q^{\mathrm{d}}(\theta, \theta_k)$ will tend to bound $L_\theta(y_{1:T})$ with greater fidelity, compared to $Q^{\mathrm{s}}(\theta, \theta_k)$. In the case of 'large disturbances' (i.e. $\Sigma_w \gg \Sigma_v$) the converse is true.

We conclude this section by drawing attention to the fidelity of the bounds from Lagrangian relaxation, i.e. $\bar{Q}_3^{\mathrm{d}}$ and $\bar{Q}_3^{\mathrm{s}}$. In Figure 4.1(a), $\bar{Q}_3^{\mathrm{d}}$ provides an effective bound on the likelihood, despite $L_\theta(y_{1:T})$ not being concave in the neighborhood of θ_k. In Figure 4.1(b), $\bar{Q}_3^{\mathrm{s}}$ almost perfectly reproduces Q^{s}, except at the boundary of the feasible set, $\mathcal{S}$, where it tends towards $-\infty$ as desired, unlike Q^{s}, which remains finite for unstable models ($A > 1$).

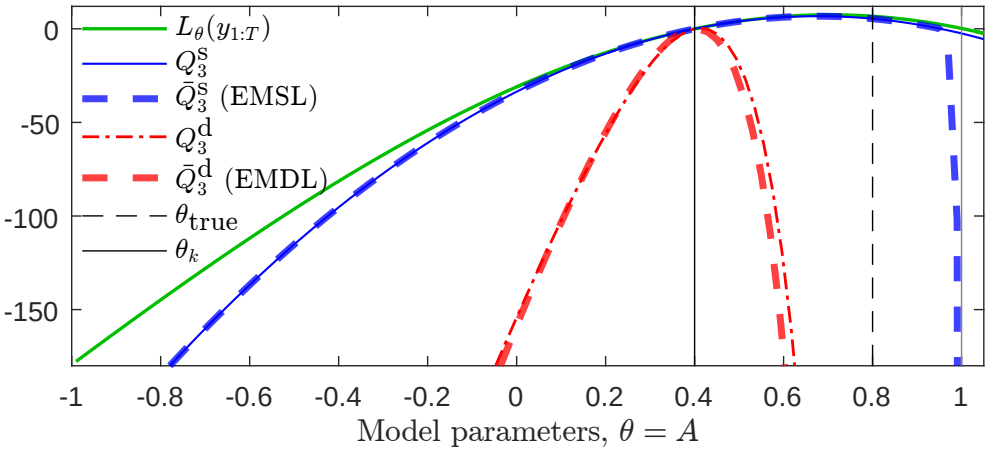
4.6 Numerical experiments

4.6.1 Stability of the identified model

This section provides empirical evidence of the value of the model stability constraints introduced in §4.4. We present examples of model instability arising in the standard unconstrained latent states formulation, during identification of models depicted in Figure 4.4. Figure 4.2 considers the case where measurement noise is more significant than the disturbances, for singular state space models; Figure 4.3 treats the 'significant disturbances' case with full rank covariance. We make the following observations. First, model instability can arise even when the true spectral radius is far from unity; c.f. Figure 4.2(b) and 4.3(b) concerning identification of the overdamped System 2 in Figure 4.4. Secondly, the consequences of instability are varied; e.g. in Figure 4.2(b), model instability leads to failure of the latent states algorithm due to poor numerical conditioning, whereas in Figure 4.3(a) the spectral radius of the identified model hovers above unity for thousands of iterations.



(a) 'Small' disturbances: $\Sigma_w = 1 \times 10^{-3}$ and $\Sigma_v = 1 \times 10^{-2}$.

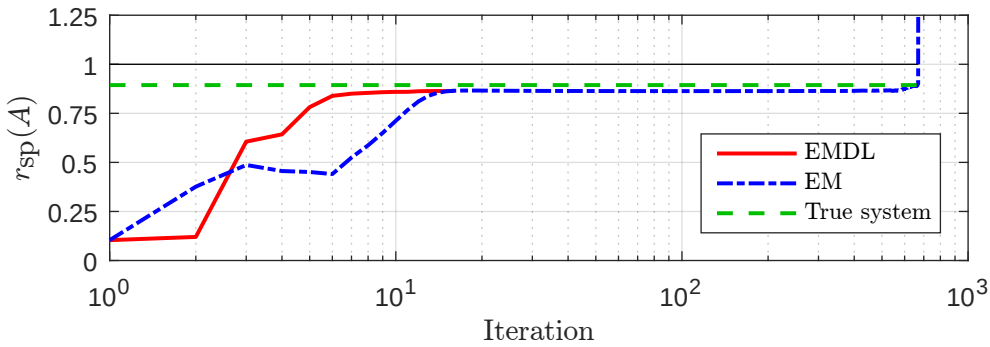


(b) 'Large' disturbances: $\Sigma_w = 10$ and $\Sigma_v = 1 \times 10^{-2}$.

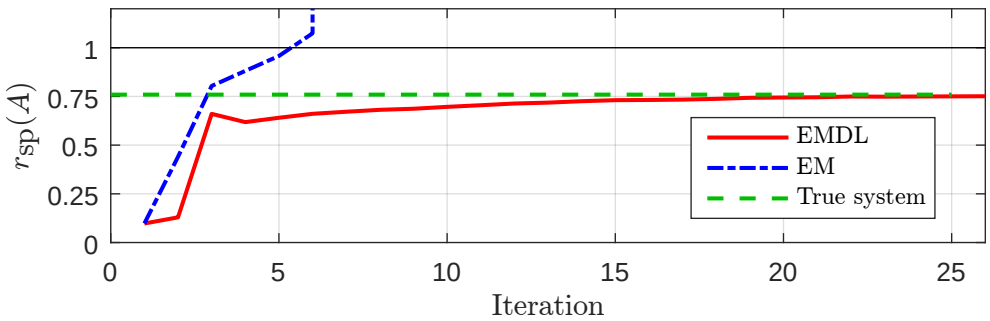
Figure 4.1 – Bounds on the log likelihood $L_\theta(y_{1:T})$ of a 1st order system with a single unknown scalar parameter, A. Q^{s} and Q^{d} denote the bounds based on latent states and disturbances resp., c.f. (4.10) and (4.16). $\bar{Q}_3^{\mathrm{s}}$ and $\bar{Q}_3^{\mathrm{d}}$ denote the bounds from Lagrangian relaxation, using latent states and disturbances resp., c.f. (4.29) and (4.38).

Such a model may achieve adequate performance on training data, yet behave unreliably should the unstable modes be excited during validation.

To supplement the results in Figures 4.2 and 4.3, we randomly generated 1500 stable SISO systems, of varying order, with Matlab's `drss` function, and report instability of the identified models in Table 4.1. Specifically, to generate problem data each model was simulated for $T = 2n_\theta$ time steps (where n_θ is the number of parameters in the model) with Σ_v set to give a SNR of 20dB, and GG' of rank 1 with eigenvalue 10^{-4}. The latent states algorithm [214] was then run for 60 seconds, randomly initialized with `drss`. The proportion of trials for which the identified model was unstable for at least one iteration is recorded in Table 4.1.

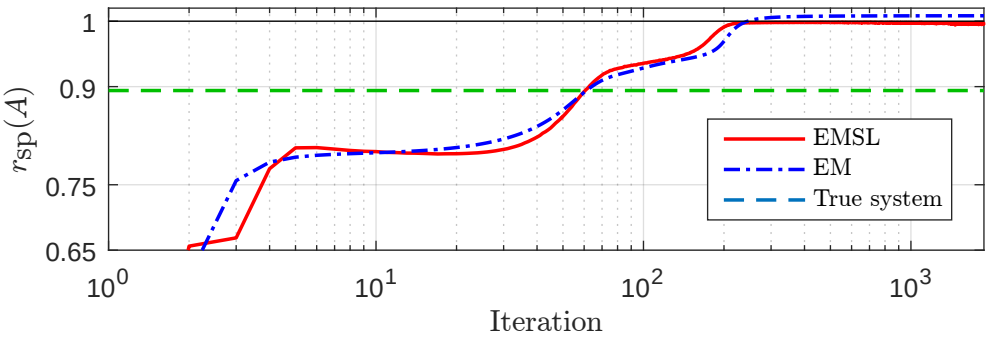


(a) System 1, underdamped: $\Sigma_v = 1.7 \times 10^{-3}$.

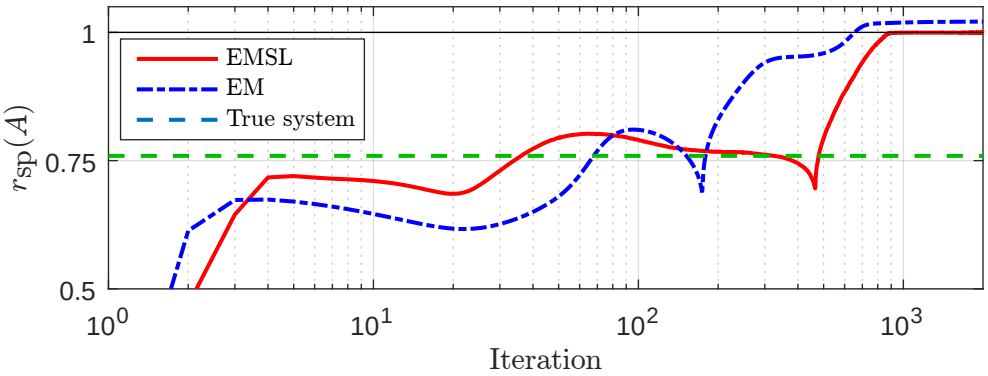


(b) System 2, overdamped: $\Sigma_v = 0.012$.

Figure 4.2 – Spectral radius of identified models at each iteration of two methods: our latent disturbances method (EMDL), and the latent states method [214] (EM); c.f. Figure 4.4 for Bode plots of true systems. For each model, the disturbance covariance is singular with $G = [0, \sqrt{10^{-5}}, 0, 0]'$. In each case, both algorithms were initialized with the same randomly generated model from `drss`. Models were trained with $T = 75$ and 100 datapoints, in (a) and (b), resp.



(a) System 1, underdamped: $\Sigma_w = I_4$, $\Sigma_v = 1.7 \times 10^{-3}$.



(b) System 2, overdamped: $\Sigma_w = I_4$, $\Sigma_v = 0.012$.

Figure 4.3 – Spectral radius of identified models at each iteration of two methods: our stable latent states method (EMSL), and the latent states method [71] (EM); c.f. Figure 4.4 for Bode plots of true systems. For each model, the disturbance covariance is full rank. In each case, both algorithms were initialized with the same randomly generated model from `drss`. Models were trained with $T = 75$ datapoints.

Table 4.1 – Proportion of trials for which the identified model was unstable for at least one iteration, using the latent states algorithm [214]. 300 trials were conducted for each model order; the true SISO models were generated with `drss`; c.f. Section 4.6.1 for details.

Model size, n_x	2	4	6	8	10
Unstable model	32%	32%	36%	48%	47%

4.6.2 Convergence rate and computation time

In this section, we demonstrate that although the per-iteration complexity of our latent disturbances formulation (EMDL), c.f. Algorithm 5, is much greater, total computation time remains competitive with conventional latent states methods in cases where measurement noise dominates disturbances. This is due to the higher fidelity bounds on likelihood achieved using latent disturbances, e.g. Figure 4.1(a), meaning fewer iterations are required

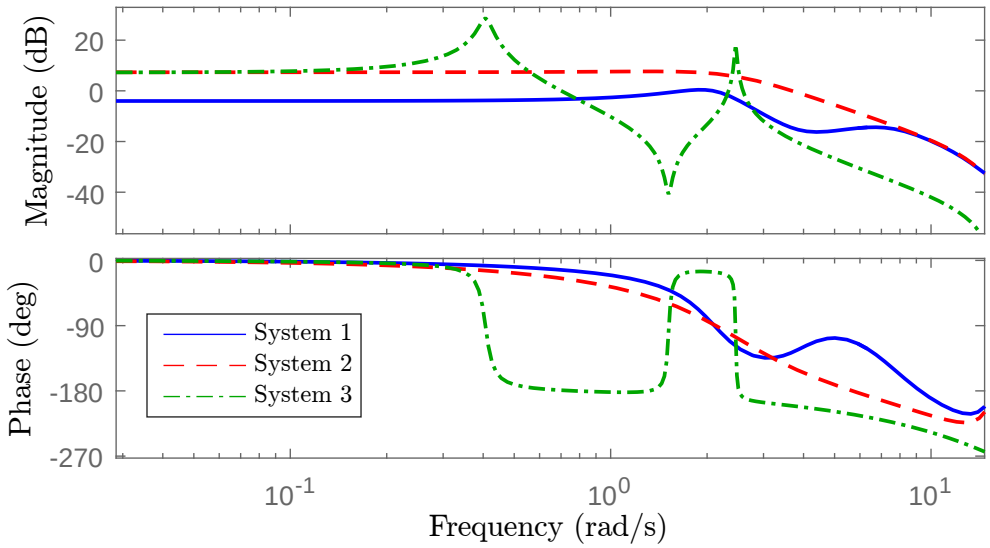


Figure 4.4 – Bode plots of 4th order systems used in the numerical experiments of Section 4.6.

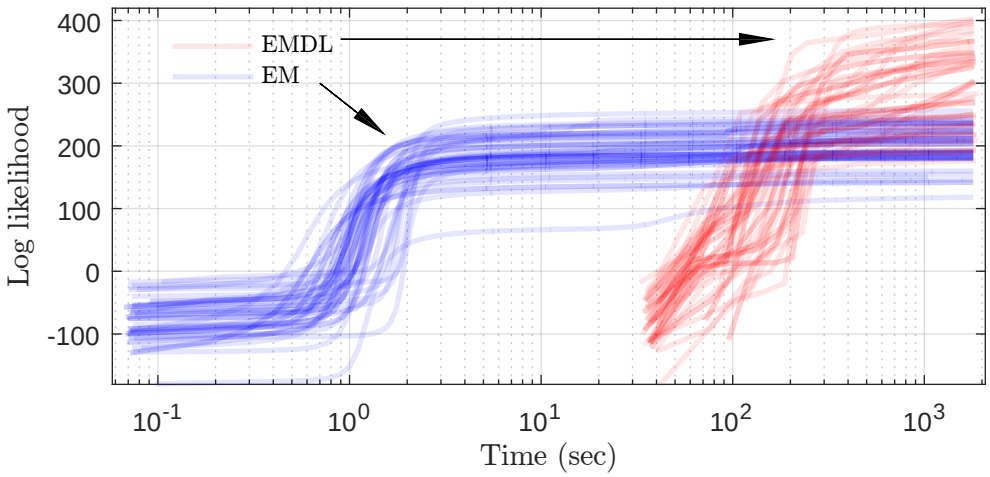
to significantly improve the likelihood. To illustrate this, we identify three 4th order linear models, the Bode plots for which are given in Figure 4.4. Each of these systems has GG' of rank 1, with eigenvalue 10^{-5}. We set Σ_v to give a signal-to-noise ratio (SNR) of approx. 20dB, which means Σ_v is two to three orders of magnitude larger than GG'. The experiment consists of 50 trials; in a single trial we repeat the following process for each system in Figure 4.4. First we simulate the system for $T = 250$ time steps, excited by $u_t \sim \mathcal{N}(0, 1)$, to generate problem data $u_{1:T}$ and $y_{1:T}$. We then run EMDL and [214] for 30 minutes. Each algorithm is initialized with the same model (randomly generated for each trial), with system matrices close to zero.

Figure 4.5 presents the log likelihood as a function of computation time. The longer per-iteration time of EMDL is immediately apparently; tens of seconds elapse before the first iteration completes. After 200 seconds of computation, EMDL is approx. equal to (or greater than) EM, and after 30 minutes EMDL has surpassed EM in almost all cases. These higher likelihoods correspond to more accurate models, as revealed by Figure 4.6, which plots the H_∞ and prediction error (on validation data) of the systems identified in Figure 4.5.

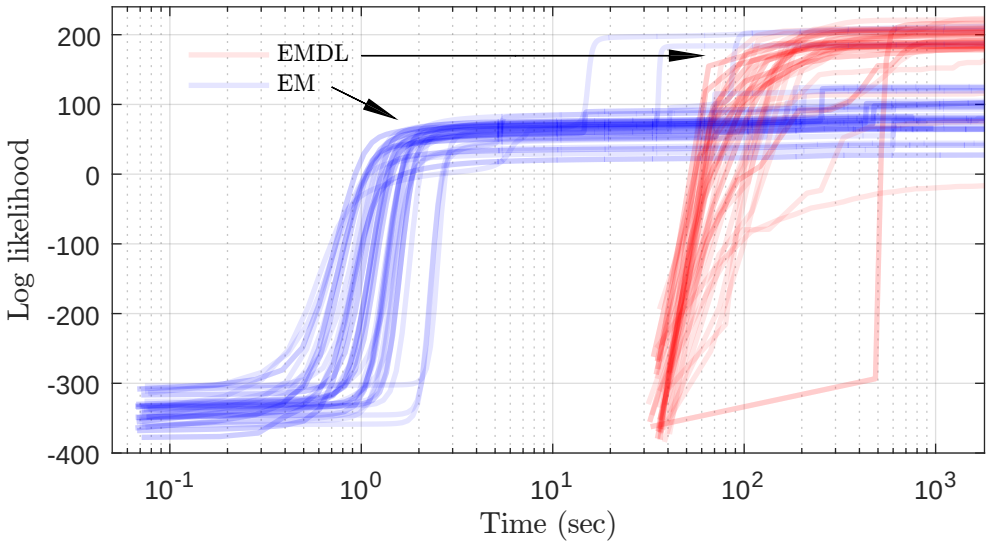
For the highly resonant System 3, EM exhibited poor performance (in both formulations) due to capture in local maxima. This is evident in Figure 4.7(b) where the H_∞ error is close to unity (again, for both formulations) indicating only marginal improvement over no dynamics at all. Similarly, with a 'warm start' (initialization with a model from Lagrangian relaxation, c.f. [236]), we observe little improvement in prediction and H_∞ error, c.f. Figures 4.7(a) and (b), respectively.

Finally, we analyse the performance of EMDL as a function of the number of datapoints used for training, T. Figure 4.8 presents H_∞ and prediction error for increasing T, for identification of System 2. System 2 was selected because capture in local maxima is less common, allowing us to study the asymptotic behavior of the global maximum more reliably. Both EMDL and the latent states algorithm [214] were run for 30 minutes in each trial. For EMDL, we observe an increase in accuracy (i.e., a decrease in both H_∞ and prediction error) for increasing T. In fact, for $T \geq 200$ the prediction error of the identified model is approximately equal to that of the true model. For the latent states algorithm, this trend is much less pronounced. The weak performance of latent states, along with the latent disturbances outliers, appears to be due to capture in local maxima, as model quality fails to improve in these cases, even after many additional iterations.

Table 4.2 records the mean computation time for a single iteration of the experimental trials carried out in Figure 4.8. Latent states methods, including EMSL, scale linearly with T, as the cost is dominated by the filtering and smoothing operations in the E step. In principle, EMDL, is $O(T^2)$, as optimization of each of the $M^{\text{d}} + 1$ bounds in (4.38) requires $O(T)$, [236]. In practice, all M^{d} singular values of Ω are not typically required for accurate approximation of $Q_3(\theta, \theta_k)$.



(a) System 1, smooth resonant peaks.

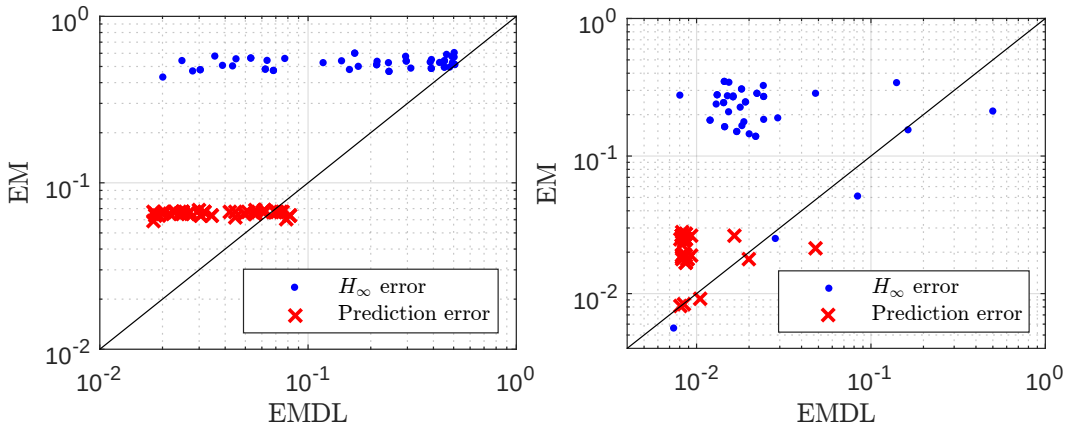


(b) System 2, overdamped.

Figure 4.5 – Log likelihood as a function of computation time for EMDL and the latent states formulation of [214] (EM). 50 trials were carried out for each system, and $T = 250$ datapoints were used for fitting. Bode plots for System 1 and 2 are depicted in Figure 4.4.

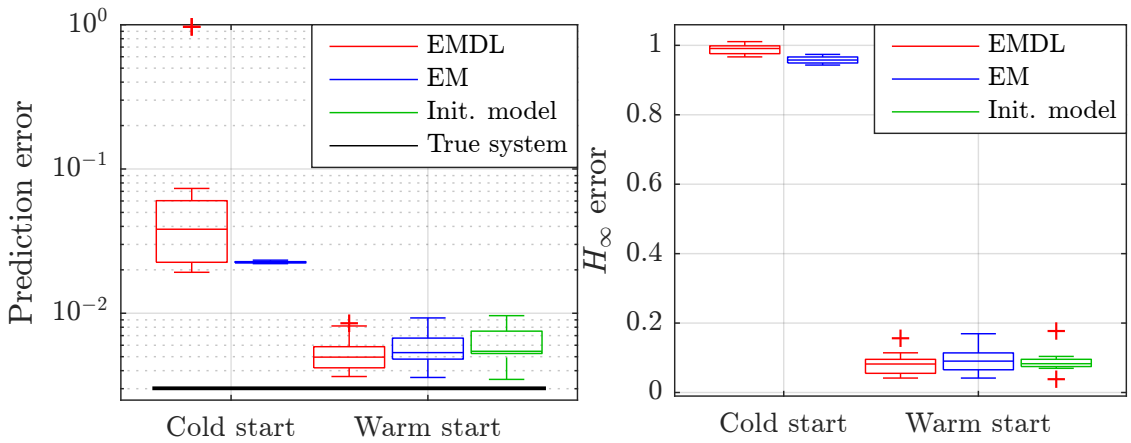
Table 4.2 – Mean per-iteration computation time (in seconds, to 3 sig. fig.) for the trials in Figure 4.8. EMSL and EMDL denote Algorithm 4 and Algorithm 5, respectively. EMSL is included for reference.

Data length, T	50	100	150	200	250
EM [214]	0.028	0.0541	0.08	0.103	0.126
EMSL	0.197	0.214	0.235	0.254	0.271
EMDL	37.5	40.7	48.9	54.6	65.6



(a) System 1. **(b)** System 2.

Figure 4.6 – Prediction error (on validation data) and H_∞ error for EMDL and the latent states formulation of [214] (EM). The systems used are those reached at the conclusion of the trials depicted in Figure 4.5.

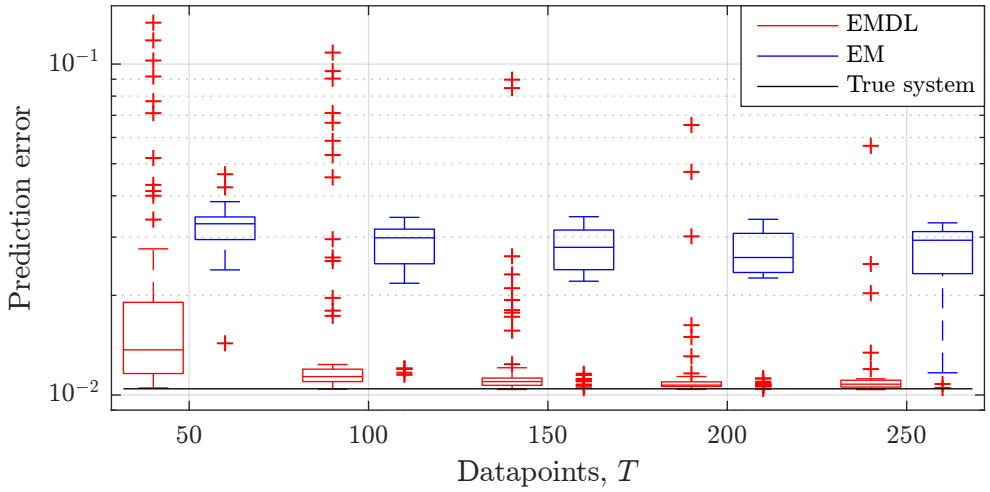


(a) Prediction error. **(b)** H_∞ error.

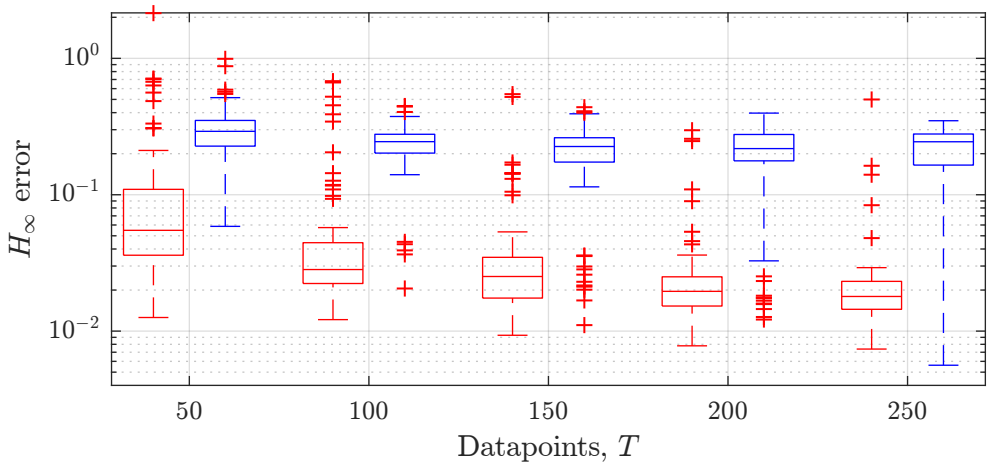
Figure 4.7 – Prediction error (on validation data) and H_∞ error for (EMDL) and the latent states formulation of [214] (EM), after identification of the highly resonant System 3, c.f. Figure 4.4. 'Cold start' and 'warm start' denote initialization with a random model (with system matrices close to zero) and a model from Lagrangian relaxation [236], respectively.

4.7 Conclusion

This chapter has incorporated model stability constraints into the maximum likelihood identification of linear dynamical systems. By combining the EM algorithm and Lagrangian relaxation, we construct tight convex bounds on the (negative) likelihood, that can be optimized over a convex parametrization of all stable linear systems, with semidefinite programing. The key practical outcomes of this work are as follows. The *de facto* choice of latent states leads to the simplest algorithms, as well as higher fidelity bounds on the



(a) Prediction error, on validation data



(b) H_∞ error.

Figure 4.8 – Prediction error (on validation data) and H_∞ error for EMDL and the latent states formulation of [214] (EM). The true model is the overdamped System 2, c.f. Figure 4.4. For each T, 50 trials were carried out, and each algorithm was run for 30 minutes.

likelihood when disturbances are more significant than measurement noise (i.e. $\Sigma_w \gg \Sigma_v$). Concerning software implementation, incorporating stability constraints into standard latent *states* algorithms is straightforward: if the identified model becomes unstable, simply replace the usual M step (4.11c) with the convex program (4.30), to continue the search over a convex set of stable models. On the other hand, when measurement noise is more significant than disturbances it may be advisable to formulate EM with latent *disturbances*. Although the per-iteration computational complexity of the ensuing algorithm is greater, the improved fidelity of the bounds on the likelihood can lead to faster convergence and more accurate models (e.g. by avoiding local maxima). Furthermore, latent disturbances lead to the most broadly applicable formulation of EM for identification of singular state space models.

4.8 Proofs

4.8.1 Proof of Lemma 4.1

The first term in (4.16), Q_1^{d}, is identical to Q_1^{s}, so we focus on Q_3^{d}. The p.d.f. $p_\theta(y_{1:T}|x_1, w_{1:T})$ is given by $p_\theta(Y|Z) = \mathcal{N}(Y; \mu_Y, \Sigma_Y)$, where μ_Y and Σ_Y are given in (4.19). Q_3^{d} may then be expressed as

$$\begin{aligned}&\mathrm{E}_{\theta_k}\left[\log \mathcal{N}(Y; \mu_Y, \Sigma_Y)|y_{1:T}\right]\\ &= -\frac{Tn_y}{2}\log 2\pi - \log\det\Sigma_Y - \mathrm{E}_{\theta_k}\left[|Y-\mu_Y|^2_{\Sigma_Y^{-1}}|y_{1:T}\right].\end{aligned}$$

Letting $\hat{Z}$ and Ω, defined in (4.18), denote the mean and covariance (respectively) of $p_{\theta_k}(x_1, w_{1:T-1}|y_{1:T})$, gives

$$Q_3(\theta, \theta_k) \propto -T\log\det\Sigma_v - \mathrm{tr}(\Sigma_Y^{-1}(\bar{C}\bar{H}\Omega\bar{H}'\bar{C}' + \hat{\Delta}\hat{\Delta}')), \tag{4.43}$$

where $\hat{\Delta} = \mathrm{E}_{\theta_k}\left[Y - \mu_Y|y_{1:T}\right]$ is defined in (4.20). □

4.8.2 Proof of Lemma 4.4

Evaluating the supremum in (4.28) yields

$$\bar{J}_\lambda^s(\theta^s, t) = \epsilon_t' H\left(H'E + E'H - \Sigma_w^{-1}\right)^{-1} H'\epsilon_t. \tag{4.44}$$

Let $e_t = \tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t$, such that $\epsilon_t = Ee_t$. Then substituting $H = (E_k')^{-1}\Sigma_w^{-1}$ into (4.44) gives $\bar{J}_\lambda^s(\theta_k^s, t) = e_t'\Sigma_w^{-1}e_t$, i.e. (4.28) is tight to (4.26) at θ_k. □

4.8.3 Proof of Lemma 4.5

First consider the $\mathrm{tr}(\Sigma_Y\hat{\Delta}\hat{\Delta}')$ term in (4.43). From (4.20), $\hat{\Delta}$ is clearly the difference between the measured output $y_{1:T}$ and the simulated output of the model with the expected value of the latent disturbances, i.e. $\hat{Z}$. Therefore, $\mathrm{tr}(\Sigma_Y\hat{\Delta}\hat{\Delta}') = \sum_{t=1}^T |y_t - Cx_t - Du_t|^2_{\Sigma_v^{-1}}$ where $\mathrm{vec}(x_{1:T}) = \bar{N}U + \bar{H}\bar{Z}$. Next, consider $\mathrm{tr}(\Sigma_Y^{-1}\bar{C}\bar{H}\Omega\bar{H}'\bar{C}')$. Decomposing $\Omega = \sum_{j=1}^{M^{\mathrm{d}}} Z^j Z^{j'}$ leads to

$$\mathrm{tr}(\Sigma_Y^{-1}\bar{C}\bar{H}\Omega\bar{H}'\bar{C}') = \sum_{j=1}^{M^{\mathrm{d}}}|\bar{C}\bar{H}w_j|^2_{\Sigma_Y^{-1}} = \sum_{j=1}^{M^{\mathrm{d}}}\sum_{t=1}^{T}|Cx_t^j|^2_{\Sigma_v^{-1}}$$

where $\mathrm{vec}(x_{1:T}^j) = \bar{H}Z^j$., i.e. the sum of M^{s} simulation error problems with $Y = 0$, $U = 0$ and $Z = Z^j$. □

4.8.4 Proof of Lemma 4.6

As $\bar{Q}_3^{\mathrm{d}}(\eta)$ is defined by a summation of convex functions, it is itself a convex function. Summation of the following inequalities

$$\begin{aligned}
&\bar{J}_{\lambda^0}(\eta, u_{1:T}, y_{1:T}, \hat{x}_{1|T}, \hat{w}_{1:T}) \geq \mathcal{E}(\eta, u_{1:T}, y_{1:T}, \hat{x}_{1|T}, \hat{w}_{1:T}),\\
&\bar{J}_{\lambda^j}(\eta, 0, 0, x_1^j, w_{1:T}^j) \geq \mathcal{E}(\eta, 0, 0, x_1^j, w_{1:T}^j), \quad j = 1, \ldots, T,\\
&\mathrm{tr}(\Sigma_{v_k}^{-1}\Sigma_v) + \log\det\Sigma_{v_k} + n_y \geq \log\det\Sigma_v,
\end{aligned}$$

gives $\bar{Q}_3^{\mathrm{d}}(\eta) \geq -Q_3(\beta, \theta_k)$. Notice that $n_y + \log\det\Sigma_{v_k} + \mathrm{tr}(\Sigma_{v_k}^{-1}\Sigma_v)$ is an affine upper bound on the concave term $\log\det\Sigma_v$, which is tight at our current best estimate of the covariance, Σ_{v_k}. □

4.8.5 Proof of Lemma 4.7

For $\theta_k^d \in \Theta^d$, we have $\Psi_k < 0$, c.f. [232, Theorem 6]. The Lagrangian in (4.37) is then concave in Δ, so the supremizing Δ satisfies $\Psi\Delta = \bar{E}_k' h^j + \bar{C}_k'(Y - \bar{D}_k U) - \bar{\epsilon}_k$. Substituting h^j from (4.39) into the above yields $\Psi\Delta = \Psi\bar{E}_k^{-1}\bar{\epsilon}_k$. As Ψ is full rank, this implies $\bar{E}_k\Delta = \bar{\epsilon_k}$. Then $J(\theta_k^d, \Delta) - \lambda F(\theta_k^d, \Delta) = J(\theta_k^d, \Delta) = \mathcal{E}(\theta_k^d)$. □

4.8.6 Proof of Proposition 4.1

When $\Sigma_w = 0$, $p_{\theta_k}(x_{1:T}|y_{1:T})$ is supported on the set

$$\mathcal{X}(\theta_k) = \{x_{1:T} : \mathrm{vec}(x_{1:T}) = \bar{N}U + \bar{H}\tilde{Z} \ \forall\ \xi_1 \in \mathbb{R}^{n_x}\}$$

where $\tilde{Z} = [\xi_1', 0]'$ and $G = 0$. Then,

$$Q^{\mathrm{s}}(\theta, \theta_k) = \int_{\mathcal{X}(\theta_k)} \log p_\theta(x_{1:T}, y_{1:T}) p_{\theta_k}(x_{1:T}|y_{1:T}) dx_{1:T}.$$

As $\Sigma_w = 0$, $p_\theta(x_{2:T}|x_1)$ is deterministic. When $A \neq A_k$ or $B \neq B_k$, $\log p_\theta(x_{2:T}|x_1) = 0$ for all $x_{1:T} \in \mathcal{X}(\theta_k)$, and so $\log p_\theta(x_{1:T}, y_{1:T})$ is undefined. As a consequence, $Q^{\mathrm{s}}(\theta, \theta_k)$ is undefined.

When $A = A_k$ and $B = B_k$, $p_\theta(x_{2:T}|x_1) = 1$ for all $x \in \mathcal{X}(\theta_k)$ and so $Q^{\mathrm{s}}(\theta, \theta_k)$ can be evaluated as usual. □

4.8.7 Proof of Proposition 4.2

As $G = 0$, $\Sigma_1 = 0$ the p.d.f. $p_{\theta_k}(x_1, w_{1:T}|y_{1:T})$ is trivially deterministic, evaluating to unity when $x_1 = \mu$ and $w_{1:T} \equiv 0$, and evaluating to zero otherwise. Therefore

$$Q^{\mathrm{d}}(\theta, \theta_k) = \log p_\theta(y_{1:T}, \mu, 0) = \log p_\theta(y_{1:T}|\mu).$$

The log likelihood can be decomposed as

$$\begin{aligned} L_\theta(y_{1:T}) &= \log \int p_\theta(y_{1:T}, x_1) dx_1 \\ &= \log \int p_\theta(y_{1:T}|x_1) p_\theta(x_1) dx_1 = \log p_\theta(y_{1:T}|\mu), \end{aligned}$$

where the final equality follows from the fact that $p_\theta(x_1)$ is a δ-function, at $x_1 = \mu$. □

4.8.8 Proof of Proposition 4.3

For a given θ, let $x^\theta_{1:T}$ denote the unique state sequence that is 'consistent' with the data, i.e. $x^\theta_{1:T} := \{x_{1:T} : y_t = Cx_t + Du_t, t = 1, \ldots, T\}$. There is also a corresponding unique disturbance sequence, denoted $w^\theta_{1:T} = \{w_{1:T} : x^\theta_{t+1} = Ax^\theta_t + Bu_t + Gw_t, t = 1, \ldots, T\}$.

As $\Sigma_v = 0$, the p.d.f. $p_{\theta_k}(x_1, w_{1:T}|y_{1:T})$ is a δ-function at $x_1 = x_1^{\theta_k}$ and $w_{1:T} = w^{\theta_k}_{1:T}$. The auxiliary function is then given by $Q^{\mathrm{d}}(\theta, \theta_k) = \log p_\theta(y_{1:T}, x_1^{\theta_k}, w^{\theta_k}_{1:T})$. We can decompose $p_\theta(y_{1:T}, x_1^{\theta_k}, w^{\theta_k}_{1:T})$ as in (4.14). As $\Sigma_v = 0$, the p.d.f. $p_\theta(y_{1:T}|x_1, w_{1:T})$ is also a δ-function at $x_1 = x_1^\theta$ and $w_{1:T} = w^\theta_{1:T}$. If $\theta \neq \theta_k$ then $x_1^\theta \neq x_1^{\theta_k}$ and $w^\theta_{1:T} \neq w^{\theta_k}_{1:T}$. In this case $p_\theta(y_{1:T}|x_1^{\theta_k}, w^{\theta_k}_{1:T}) = 0$ and so $Q^{\mathrm{d}}(\theta, \theta_k)$ is undefined.

When $\theta = \theta_k$, $p_\theta(y_{1:T}|x_1^{\theta_k}, w^{\theta_k}_{1:T}) = 1$ and $Q^{\mathrm{d}}(\theta, \theta_k)$ can be evaluated as usual. □

4.8.9 Proof of Proposition 4.4

For a given θ, let $x^\theta_{1:T}$ denote the unique state sequence that is 'consistent' with the data, i.e. $x^\theta_{1:T} := \{x_{1:T} : y_t = Cx_t + Du_t, t = 1, \ldots, T\}$. As $\Sigma_v = 0$, given $y_{1:T}$ both $p_{\theta_k}(x_{1:T}|y_{1:T})$ and $p_\theta(y_{1:T}|x_{1:T})$ are δ-functions at $x_{1:T} = x^\theta_{1:T}$. The auxiliary function is then given by

$$Q^{\mathrm{s}}(\theta, \theta_k) = \log p_\theta(y_{1:T}, x^{\theta_k}_{1:T}).$$

Let us now consider the two cases:

i. When $C \neq C_k$ or $D \neq D_k$, $x^\theta_{1:T} \neq x^{\theta_k}_{1:T}$ and so $p_\theta(y_{1:T}|x^{\theta_k}_{1:T}) = 0$. Therefore, $Q^{\mathrm{s}}(\theta, \theta_k)$ is undefined.

ii. When $C = C_k$ and $D = D_k$, $x_{1:T}^{\theta} = x_{1:T}^{\theta_k}$ and so

$$Q^{\mathrm{s}}(\theta, \theta_k) = \log p_\theta(y_{1:T}|x_{1:T}^{\theta}) p_\theta(x_{1:T}^{\theta}) = \log p_\theta(x_{1:T}^{\theta}).$$

The likelihood can be expressed as

$$\begin{aligned} L_\theta(y_{1:T}) &= \log \int p_\theta(y_{1:T}|x_{1:T}) p_\theta(x_{1:T}) dx_{1:T} \\ &= \log p_\theta(x_{1:T}^{\theta}), \end{aligned}$$

where the second inequality comes from the fact that $p_\theta(y_{1:T}|x_{1:T})$ is a δ-function. Therefore, $L_\theta(y_{1:T}) = Q^{\mathrm{s}}(\theta, \theta_k)$. □

Chapter 5

Identification of positive dynamical systems

The central motif of this thesis has been the scalable application of convex optimization to system identification, namely: minimization of convex quality-of-fit criteria over convex parametrizations of stable models. Hitherto, our main tool has been the linear matrix inequality (LMI), which can be optimized via semidefinite programming. Unfortunately, semidefinite programming suffers from poor scalability: memory requirements for large-scale problems are severe, unless structure (e.g., sparsity) can be exploited, c.f., Section 2.2.3.

In this chapter, we consider identification of *internally positive systems*, which have the property that nonnegative inputs lead to nonnegative internal states and outputs. Such system representations frequently arise in applications in which physical constraints imply that the quantities of interest are nonnegative. Internally positive systems enjoy substantially simpler stability and performance analysis, compared to general the LTI case. Specifically, a Lyapunov function that is linear, rather than quadratic, in both the state variable and the number of parameters is necessary and sufficient to verify stability. We exploit these simplified linear stability conditions to derive a polytopic parametrization of all stable positive systems. Furthermore, due to ease of decomposability compared to LMIs, these linear constraints are amenable to distributed optimization suitable for identification of large-scale dynamical systems. To access these benefits, we also derive convex quality-of-fit metrics based on Lagrangian relaxation, and demonstrate superior performance over existing approaches based on weighted equation error. We also derive convex upper bounds on simulation error that can be optimized with linear programming, by utilizing ℓ_1 dissipativity theory for positive systems.

Connection to other chapters

Scalability has been a recurrent theme throughout this thesis. For instance, Chapter 3 was concerned with scalability of identification with respect to the length of the training data. The specialized algorithms developed in Chapter 3 reduced computational complexity of Lagrangian relaxation of simulation error to linear growth with the number of data points, T, compared to the cubic growth exhibited by generic solvers. This chapter is concerned with scalability of identification with respect to system scale, i.e., the size of the state dimension n_x.

The main message of this chapter is that several problems in system identification, namely the construction of convex parametrizations of stable models and convex quality-of-fit criteria, are substantially simplified for positive systems. In particular, the linear stability and dissipativity theory for positive systems permits many of the ideas and techniques developed in the preceding chapters to be extended to large-scale systems.

Publications

Some of the material presented in this chapter also appears in:

> **J. Umenberger**, I.R. Manchester. Scalable identification of stable positive systems. In *Proceedings of the IEEE Conference for Decision and Control (CDC)*. 2016.

5.1 Introduction

Traffic flow through urban centers, antiretroviral treatment of infectious disease and the smart electricity grid are but a few examples of the diverse array of large-scale systems for which modeling and control is becoming increasingly important. In many of these applications, physical constraints imply that the quantities of interest - e.g., number of cars passing through a tunnel, concentrations of pathogens, or power through a transmission line - are nonnegative. In such cases, it is appropriate to model the situation as a *positive system*, in which the set of nonnegative internal states remains invariant under the dynamics.

Over the past decade, positive systems have received increased attention from the control community, largely due to the fact that many performance and stability results in linear system theory are simplified when the dynamics are positive. For example, static state and output feedback controllers were designed using *linear programming* in [49] and [188], respectively. Stability and dissipativity theory for positive systems based on *linear* storage functions and supply rates was developed in [82], and employed for robust stability analysis in [31]. Similarly, the work of [227] provided a bounded real lemma for positive systems based on a *diagonal* quadratic storage function, which enabled the design of *structured* H_∞

controllers. More recently, [190] has presented novel versions of many of the above results, with an emphasis on scalable controller synthesis and verification.

Model-based design and analysis depends of course on the availability accurate system models. While in some applications these come from first-principles, when physical models are either unknown or too complex, some form of data-driven modeling is necessary. So far, most of the research effort has focused on the so-called *positive realization problem*, i.e., determining the conditions for which there exists an internally positive realization of a system with nonnegative impulse response, c.f. [7, 13, 14]. In contrast, the identification problem has received little attention. Among the few published results are [15], which presents conditions for 'compartmentality'[1] of identified models, and [50], which considers identification of third order internally positive systems with Poisson output.

As discussed in Section 2.3, stability of the identified model is often a desirable property, c.f., also [131, §4]. To the best of our knowledge, the only work concerned with identification of stable positive systems is that of [94]. This work, which also appears in [83, §18], extends the approach presented in [113] to identification of positive systems, by deriving an LMI parametrization of stable positive systems that can be optimized over with semidefinite programming (SDP). Specifically, the usual dense positive definite solution to the Lyapunov equation for generic LTI systems is replaced with a diagonal solution; for positive systems this introduces no conservatism, c.f., Section 5.3 for details. The reduction in decision variables associated with this diagonal solution notwithstanding, such an approach is not really suitable for identification of large-scale systems that so often arise in networks with positive dynamics, due to the poor scalability of SDP.

Identification of large-scale systems (indeed large-scale optimization in general), typically requires decomposition of the problem into many sub-problems of lower complexity. Depending on the application, this is termed distributed, decentralized or parallel computation [20], c.f. Section 5.2.5 for a discussion of the terminology used in this chapter. Second order interior point methods for SDP employ barrier functions that make the decomposition of linear matrix inequalities problematic, unless some chordal sparsity properties can be exploited; c.f., [67, 139, 223, 244, 253] for recent progress on scalable SDP. In contrast, decomposition methods for element-wise inequality constraints, such as the stability conditions for positive systems based on Lyapunov functions, are far more mature. Early research on distributed solutions for such optimization problems includes dual decomposition [48, 57], with improvements in convergence properties obtained by the alternating direction method of multipliers [30, 69, 73]. More recently, approaches based on game theory have proven successful, especially when communication between processors is unreliable [120, 145].

The contributions of chapter are as follows. In Section 5.3 we present two new convex parametrizations of all stable positive systems: an LMI based parametrization that generalizes [94], and a polytopic parametrization that leverages the simplified stability conditions

[1] Compartmental systems are a special case of internally positive systems.

for positive dynamics. In Section 5.4 we present convex quality-of-fit criteria that are compatible with the parametrizations of Section 5.3. As an alternative to weighted equation error [94, 113], we propose Lagrangian relaxation of equation error; superior performance of the latter is demonstrated empirically in Section 5.5.1. We also present convex upper bounds for the ℓ_1 norm of simulation error (a.k.a. output error), based on linear dissipativity theory for positive systems; the utility of these bounds for identification of structured systems is demonstrated in Section 5.5.3. Finally, in Section 5.6, we exploit the decomposability of the polytopic parametrization of stable models to derive distributed algorithms for minimization of the criteria presented in Section 5.4.

5.2 Preliminaries

5.2.1 Notation

Specific notation used in this chapter is as follows. For real matrices and vectors $A, B \in \mathbb{R}^{m\times n}$, $A < (\leq) B$ denotes element-wise inequality, whereas $A \prec (\preceq) B$ means $B - A$ is positive definite (semidefinite). The transpose of A is denoted A'. For $A \in \mathbb{R}^{m\times n}$, $A(i,j)$ denotes the scalar entry in the ith row and j^{th} column. We define the sets $\mathbb{R}^n_{++} := \{a \in \mathbb{R}^n : a > 0\}$ and $\mathbb{R}^n_{+} := \{a \in \mathbb{R}^n : a \geq 0\}$. For $a \in \mathbb{R}^n$, $|a| \in \mathbb{R}^n_{+}$ denotes element-wise absolute value, and $|a|_\sigma := \left(\sum_{i=1}^n |a(i)|^\sigma\right)^{1/\sigma}$ denotes the σ-norm. We define $\mathbf{1} \in \mathbb{R}^n$ as the vector with all elements equal to 1; the dimension n can be inferred from the context. The spectral radius (largest magnitude of the eigenvalues) of a matrix A is denoted $r_\lambda(A)$.

5.2.2 Positive state space models

This chapter concerns the identification discrete time *positive* LTI models of the form

$$x_{t+1} = Ax_t + Bu_t, \tag{5.1a}$$

$$y_t = Cx_t + Du_t, \tag{5.1b}$$

where $u \in \mathbb{R}^{n_u}$, $y \in \mathbb{R}^{n_y}$ and $x \in \mathbb{R}^{n_x}$ denote the input, output and state, respectively. We can distinguish between two different notions of positivity, namely external and internal positivity.

Definition 5.1. A linear system (5.1) is called *externally positive* if and only if its output corresponding to a zero initial state is nonnegative for every nonnegative input.

Definition 5.2. A linear system (5.1) is called *internally* positive if and only if its state and output are nonnegative for every nonnegative input and every nonnegative initial state.

It is apparent that internal positivity implies external positivity; however, in general, it is NP-hard to test whether a system is externally positive.

Proposition 5.1 ([58])**.** A linear system (A, B, C, D) is externally positive if and only if $\forall t \geq 0 : Ce^{At}B \geq 0$ and $D \geq 0$.

By contrast, conditions for internal positivity are considerably more tractable. Internal positivity of (5.1) requires that the non-negative orthant $\mathbb{R}^{n_x}_{\geq 0}$ is invariant w.r.t. A. In [17] it is shown, that this is the case if and only if $A \geq 0$.

Proposition 5.2 ([58])**.** A discrete linear system (A, B, C, D) is internally positive if and only if A, B, C, D are element-wise nonnegative

In this chapter, we are interested in internally positive (henceforth, *positive*) systems only.

5.2.3 Stability of positive systems

Nonnegativity of the state variable x_t greatly simplifies Lyapunov stability analysis of positive systems:

Lemma 5.1 ([17, Lemma 6.2.1])**.** For $A \geq 0$ the following statements are equivalent:

1. The matrix A is Schur stable, i.e. $r_\lambda(A) < 1$.
2. There exists *diagonal* $P \succ 0$ such that $A'PA - P \prec 0$.
3. There exists $p \in \mathbb{R}^n_{++}$ such that $p'A < p'$.

The dynamical systems interpretation of this result is that $V(x) = p'x$ serves as a linear Lyapunov function for the system $x_{t+1} = Ax_t$. We denote the set of all Schur stable positive matrices by $\mathcal{S}_+$, i.e., $\mathcal{S}_+ := \{A \in \mathbb{R}^{n\times n} : A \geq 0,\ r_\lambda(A) < 1\}$. The set $\mathcal{S}_+$ is nonconvex, as the stability conditions $A'PA \prec P$ (and $p'A < p'$) are nonconvex in A and P (and p), jointly.

5.2.4 Problem data

We assume data of the form $\mathcal{Z}^T_{\mathrm{DT}} = \{\tilde{u}_t, \tilde{y}_t, \tilde{x}_t\}_{t=1}^T$ where $\tilde{u}$, $\tilde{y}$ and $\tilde{x}$ denote (possibly noisy) measurements of u, y and x, respectively. Notice that measurements (or at least estimates) of the state x are required. In this chapter, we have in mind the identification of networked systems such that $x = [x(1), \ldots, x(n_x)]$, where $x(i)$ denotes the measurable state at node i, e.g. transport networks where $x(i)$ denotes traffic density [210]. This is a rather restrictive assumption, necessitated by the fact that popular state estimation techniques, e.g. subspace methods, return state estimates subject to an arbitrary coordinate transformation [241, §2.2], which may not be consistent with a positive realization of the dynamics. Subspace methods for positive systems are an important subject for future research.

We note, in passing, that if one desires only external positivity of the identified model, then the arbitrary transformation introduced by subspace methods poses no difficulty. Our recent work [80] provides convex conditions for *cone invariance* of identified model, which is sufficient for external positivity. In particular, it is shown that when the true system (generating the problem data) is externally positive, enforcing cone invariance instead of internal positivity, as a means of ensuring input-output positivity, leads to much more accurate models.

5.2.5 Parallel and distributed identification

In Bertsekas and Tsitsikils' classic text [20], a distinction is made between *parallel* computing systems, characterized by many processors, working together in close proximity with reliable communication, and *distributed* systems, in which processors may be far apart and interprocessor communication is unreliable and possibly delayed [20, §1.1.2]. In this chapter we use the term *distributed* to refer to algorithms that decompose large optimization problems into many smaller problems, usually to reduce the memory requirements; refer to Section 5.6.1 for a thorough discussion of the scope and intended applications of the distributed algorithms presented in this chapter.

5.3 Convex parametrizations of stable models

The fundamental obstacle to optimization subject to stability constraints is the nonconvexity of the simultaneous search for the model parameters A, and a Lyapunov function V. The usual strategy for constructing a convex parametrization of all Schur stable matrices $\mathcal{S}$ is to adopt a Lyapunov function $V(x) = |x|_E^2$ and a change of variables, e.g. $F = AE$. Then the Lyapunov inequality $FE^{-1}F' \prec E$ is convex in E and F, and $A \in \mathcal{S}$ can be recovered as $A = FE^{-1}$, c.f., [113]. For internally positive systems, there is the additional challenge of ensuring nonnegativity of recovered solution. Fortunately, from Lemma 5.1, a diagonal E is necessary and sufficient, and thus there is no conservatism in using $E \in \mathbb{D}$ so that $F \geq 0$ implies $FE^{-1} \geq 0$. This is the approach presented in Haddad [94], which leads to the following convex parametrization of $\mathcal{S}_+$:

$$\Theta_l = \{E, F : M_l \succeq 0,\ F \geq 0\}, \tag{5.2}$$

where, for some arbitrarily small tolerance $\delta > 0$,

$$M_l := \begin{bmatrix} E - \delta I & F' \\ F & E \end{bmatrix}. \tag{5.3}$$

In this section we introduce two alternative convex parametrizations of $\mathcal{S}_+$: (i) an LMI representation that offers additional flexibility compared to Θ_l, and (ii) a polytopic parametriza-

tion that facilitates distributed optimization.

5.3.1 LMI parametrization with M-matrices

To optimize over Θ_l, it is necessary to *weight* the quality-of-fit metric by E in such a way that the product $F = AE$ appears as a decision variable. We will discuss this in greater detail in Section 5.4. For now, we emphasize that the same matrix E is used as a weight in the cost function and in the Lyapunov function $V(x) = |x|_E^2$. In what follows, we introduce a more flexible convex parametrization of $\mathcal{S}_+$. The idea is to use a slightly different change of variables, $F = EA$, and the Lyapunov function $V(x) = |Ex|_{P^{-1}}^2$, where E and P are not required to be diagonal. To ensure positivity of $A = E^{-1}F$, we require that E be an *M-matrix*:

Definition 5.3. An M-matrix is a matrix $H \in \mathbb{R}^{n\times n}$ with the properties: (i) $H(i,j) \leq 0$ for $i \neq j$, (ii) the real part of each eigenvalue of H is positive.

Equivalently, an M-matrix is a negative Hurwitz Metzler matrix. For our purposes, M-matrices have the following key properties:

Lemma 5.2 ([182, Theorem 2.1])**.** Suppose $(-H) \in \mathbb{M}^{n\times n}$. Then the following statements are equivalent: (i) H is an M-matrix, (ii) there exists diagonal $D \succ 0$ such that $HD+DH' \succ 0$, (iii) H is inverse-positive, i.e. H^{-1} exists and $H^{-1} \succ 0$.

Property (ii) implies that $E + E' \succ 0$ and $-E \in \mathbb{M}$ are sufficient for E to be an M-matrix, whereas property (iii) ensures positivity of $A = E^{-1}F$, when $F \geq 0$, without requiring E to be diagonal. We can now define the following convex parametrization of $\mathcal{S}_+$:

$$\Theta_m := \{E, F : \ -E \in \mathbb{M}^{n_x\times n_x}, F \geq 0, \ \exists P \in \mathbb{S}_{++}^{n_x}, \text{ s.t. } M_m \succeq 0\}, \tag{5.4}$$

where, for some arbitrarily small tolerance $\delta > 0$,

$$M_m := \begin{bmatrix} E + E' - P - \delta I & F' \\ F & P \end{bmatrix}. \tag{5.5}$$

Theorem 5.1. The set Θ_m is a convex parametrization of all (element-wise) nonnegative Schur matrices, i.e., $A \in \mathcal{S}_+$ iff there exists $\{P, E, F\} \in \Theta_m$ such that $A = E^{-1}F$.

Proof. Refer to Section 5.8.1. □

5.3.2 Polytopic parametrization

So far, we have used quadratic Lyapunov functions to derive convex parametrizations of $\mathcal{S}_+$ expressed in terms of LMIs. In this section, we exploit the stability conditions based on linear Lyapunov functions, c.f. Lemma 5.1, to develop a polytopic parametrization of $\mathcal{S}_+$. The idea is the same: we introduce the linear Lyapunov function $V(x) = \mathbf{1}'Ex$ with $E \in \mathbb{D}$, along with the change of variables $F = EA$. This leads to the following convex parametrization of $\mathcal{S}_+$:

$$\Theta_p := \{E, F : E \in \mathbb{D},\ F \geq 0,\ \mathbf{1}'F - \mathbf{1}'E \leq -\delta\mathbf{1}'\}. \tag{5.6}$$

Theorem 5.2. The set Θ_p is a convex (polytopic) parametrization of all (element-wise) nonnegative Schur matrices, i.e., $A \in \mathcal{S}_+$ iff there exists $\{E, F\} \in \Theta_p$ such that $A = E^{-1}F$.

Proof. Refer to Section 5.8.2. □

5.3.3 Discussion

This section has presented two new convex parametrizations of $\mathcal{S}_+$: Θ_m defined by the LMI (5.5), and the polytopic set Θ_p. A few comments are in order. First, it is apparent that Θ_m is a generalization of Θ_l, defined in (5.2) and introduced in [94]. For example, with the choices $E \in \mathbb{D}$ and $E = P$ in (5.5), Θ_m reduces to Θ_l. Other choices of E and P can be made to control complexity of the parametrization (i.e., the number of decision variables), e.g. $-E \in \mathbb{M}$ or $E \in \mathbb{D}$, $P \in \mathbb{S}$ or $P \in \mathbb{D}$. An advantage of choosing $E \in \mathbb{D}$ is that sparsity in F is preserved in $A = E^{-1}F$.

It is worth emphasizing that all of the convex parametrizations of $\mathcal{S}_+$ utilize an implicit representation of A. Consequently, many of the standard quality-of-fit criteria defined for (5.1) are no longer valid. Convex criteria compatible with the implicit representations is the subject of Section 5.4. Notice too, that we have two different implicit representations: specifically, E can either enter from the right ($F = AE$), or from the left ($F = EA$). The choice of representation depends on the quality-of-fit criterion being optimized; $E = AF$ is used for weighted equation error, c.f. Section 5.4.1, whereas $F = EA$ is used for Lagrangian relaxation, c.f. 5.4.2. For Θ_l and Θ_p it is trivial to reformulate the stability condition with either representation. For example, formulate the polytopic set Θ_p with $F = AE$, one simply replaces the inequality $\mathbf{1}'F - \mathbf{1}'E \leq -\delta\mathbf{1}'$ with $F\mathbf{1} - E\mathbf{1} \leq -\delta\mathbf{1}$ in (5.6).

5.4 Convex quality-of-fit criteria

In this section we present a variety of convex quality-of-fit metrics (i.e. cost functions to be minimized) that are compatible with the convex parametrizations of stable positive systems

developed in Section 5.3.

5.4.1 Weighted equation error

The 2-norm of equation error, a.k.a. the *least squares criterion*, is a popular quality-of-fit metric, used, e.g., in identification of autoregressive models [129, §4.2] and in subspace methods [241, §2.4]. For LTI models, equation error is given by

$$\sum_{t=1}^{T} \|\epsilon_t\|_2^2 + \sum_{t=1}^{T} \|\eta_t\|_2^2, \tag{5.7}$$

where $\epsilon_t = \tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t$ and $\eta_t = \tilde{y}_t - C\tilde{x}_t - D\tilde{u}_t$. The problem $\min_{C,D} \sum_t \|\eta_t\|_2^2$ s.t. $C \geq 0, D \geq 0$ is a convex quadratic program, so we will focus our attention on

$$\min_{A\geq 0, B\geq 0} \left\{ \mathcal{E}_2^{\mathrm{e}} := \sum_{t=1}^{T} \|\epsilon_t\|_2^2 \right\} \quad \text{s.t. } A \in \mathcal{S}_+. \tag{5.8}$$

The main challenge in solving (5.8) is nonconvexity of $\mathcal{S}_+$. For the convex parametrizations of $\mathcal{S}_+$ introduced in Section 5.3, equation error is no longer well defined, as we must work with the new decision variable $F = AE$, instead of A.

To circumvent this difficulty, the work of [113] proposed minimization of *weighted* equation error:

$$\mathcal{E}_2^{\mathrm{w}} := \|\tilde{x}_{2:T} - [A\ B] \begin{bmatrix} \tilde{x}_{1:T-1} \\ \tilde{u}_{1:T-1} \end{bmatrix} W\|_F^2. \tag{5.9}$$

By a judicious choice of weighting matrix, namely

$$W = \begin{bmatrix} \tilde{x}_{1:T-1} \\ \tilde{u}_{1:T-1} \end{bmatrix}^{\dagger} \begin{bmatrix} E & 0 \\ 0 & I \end{bmatrix}, \tag{5.10}$$

weighted equation error in (5.9) becomes

$$\mathcal{E}_2^{\mathrm{w}} = \|[\tilde{A}E\ \tilde{B}] - [F\ B]\|_F^2 = \|\tilde{A}E - F\|_F^2 + \|\tilde{B} - B\|_F^2, \tag{5.11}$$

where $\tilde{A}$ and $\tilde{B}$ are the unconstrained least squares minimizers of $\mathcal{E}_2^{\mathrm{e}}$. Notice that F has replaced A in (5.9), and thus $\mathcal{E}_2^{\mathrm{w}}$ is compatible with the change of variables used in the convex parametrizations of $\mathcal{S}_+$. This approach was extended to identification of stable positive systems in [94], where it was shown that

$$\max(0, \tilde{B}) = \arg\min_{B\geq 0} \|\tilde{B} - B\|_F^2. \tag{5.12}$$

Here, max applies element-wise. All that remains is to solve

$$\min_{E,F} \ \|\tilde{A}E - F\|_F^2, \quad \text{s.t. } (E,F) \in \Theta, \tag{5.13}$$

where Θ is one of the convex parametrizations of $\mathcal{S}_+$ presented in Section 5.3. For instance, choosing $\Theta = \Theta_m$ gives the most flexible parametrization, and leads to a semidefinite program. Choosing $\Theta = \Theta_l$ with $E \in \mathbb{D}$ gives the approach outlined in [94]. Finally, choosing the polytopic parametrization $\Theta = \Theta_p$ leads to a convex quadratic program, which is amenable to a distributed solution method c.f. Section 5.6.

5.4.2 Lagrangian relaxation of equation error

To optimize over a convex approximation of (5.8), Lagrangian relaxation (LR) may be used as an alternative to weighted equation error in (5.9). We begin with a quick recap of LR; c.f. [232, §2] for details in a system identification context. Consider a constrained optimization problem

$$\min_{\theta,\Delta} \ J(\theta,\Delta) \text{ s.t. } h(\theta,\Delta) = 0, \tag{5.14}$$

where $J(\theta,\Delta)$ and $h(\theta,\Delta)$, are convex and affine in θ, respectively. The Lagrangian relaxation of (5.14) takes the form

$$\bar{J}_\lambda(\theta) = \sup_{\Delta} \ J(\theta,\Delta) - 2\lambda(\Delta)'h(\theta,\Delta). \tag{5.15}$$

Here, $\lambda(\Delta)$ may be interpreted as a Lagrange multiplier. For arbitrary λ, the function $\bar{J}_\lambda(\theta)$ has two key properties:

1) It is convex in θ. Recall that J and h are convex and affine in θ, respectively. As such, $\bar{J}_\lambda(\theta)$ is the supremum of an infinite family of convex functions, and is therefore convex in θ; c.f., Section 3.2.3 of [28].

2) It is an upper bound for the original problem (5.14). Given θ, let Δ^* be any Δ such that $h(\theta,\Delta^*) = 0$. Then

$$J(\theta,\Delta^*) - 2\lambda(\Delta)'h(\theta,\Delta^*) = J(\theta,\Delta^*),$$

which implies that the supremum over all Δ can be no smaller than this feasible solution.

The original optimization problem (5.14) may then be approximated by the convex program $\min_\theta \ \bar{J}_\lambda(\theta)$.

We now present the LR of (5.8). We shall refer to this as Lagrangian relaxation of equation error (LREE). To optimize over a convex parametrization of $\mathcal{S}_+$, we work with an implicit representation of (5.1a),

$$Ex_{t+1} = Fx_t + Ku_t. \tag{5.16}$$

This implicit representation also improves the performance of the LR, as LR of equivalent constraints leads to non-equivalent bounds. Specifically, consider the constraint functions

$$h_1(\theta, x) = x_{t+1} - Ax_t - Bu_t, \tag{5.17}$$

$$h_2(\theta, x) = Ex_{t+1} - EAx_t - EBu_t. \tag{5.18}$$

Both h_1 and h_2 have the same feasible set, but the redundant parametrization in h_2 will lead to tighter bounds (i.e. bounds with lower numerical value) in general.

For convenience, introduce $\theta = \{E, F, K\}$ and define $\bar{\Theta} := \{\theta : K \geq 0, (E, F) \in \Theta\}$, where Θ is any one of the convex parametrizations of $\mathcal{S}_+$ developed in Section 5.3, e.g., Θ_m, Θ_l or Θ_p. To apply LR to (5.8), first consider the problem

$$\min_{A \geq 0, B \geq 0} \left\{ \|\epsilon_t\|_2^2 = \|\tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t\|_2^2 \right\} \quad \text{s.t. } A \in \mathcal{S}_+, \tag{5.19}$$

for ease of exposition. Problem (5.19) is equivalent to

$$\min_{\theta \in \bar{\Theta}, x_{t+1}} \|\tilde{x}_{t+1} - x_{t+1}\|_2^2 \quad \text{s.t. } Ex_{t+1} = F\tilde{x}_t + K\tilde{u}_t \tag{5.20}$$

as both (5.19) and (5.20) have the same objective and feasible set. Introducing $\Delta = \tilde{x}_{t+1} - x_{t+1}$ and $\bar{\epsilon}_t = E\tilde{x}_{t+1} - F\tilde{x}_t - K\tilde{u}_t$, the Lagrangian relaxation of (5.20) is given by

$$\bar{J}_\lambda(\theta, t) := \sup_{\Delta} \|\Delta\|_2^2 - 2\lambda_t(\Delta)'(E\Delta - \bar{\epsilon}_t) \tag{5.21}$$

for some multiplier $\lambda_t(\Delta)$. Convexity of $\bar{J}_\lambda(\theta, t)$ is guaranteed for all multipliers λ_t that are independent of θ. We propose the specific choice $\lambda_t(\Delta) = H\Delta_t = H(\tilde{x}_{t+1} - x_{t+1})$, where $H \in \mathbb{R}^{n_x \times n_x}$ is some user specified constant matrix. With this multiplier, (5.21) can be written explicitly as

$$\bar{J}_\lambda(\theta, t) = |H'\bar{\epsilon}_t|^2_{(H'E+E'H-I)^{-1}}. \tag{5.22}$$

As $\bar{J}_\lambda(\theta, t) \geq \|\epsilon_t\|_2^2$ it is clear that

$$\bar{J}_\lambda(\theta) := \sum_{t=1}^{T} \bar{J}_\lambda(\theta, t) \geq \sum_{t=1}^{T} \|\epsilon_t\|_2^2, \tag{5.23}$$

i.e., $\bar{J}_\lambda$ is an upper bound on equation error. In summary:

Theorem 5.3. The function $\bar{J}_\lambda(\theta)$, defined in (5.23), is a convex upper bound on $\sum_{t=1}^{T} \|\epsilon_t\|_2^2$ in (5.8). Furthermore $\min_\theta \ \bar{J}_\lambda(\theta)$ s.t. $\theta = (E, F, K) \in \bar{\Theta}$ can be solved as the following SDP,

$$\begin{aligned} \min_{R,\theta\in\bar{\Theta}} \quad & \operatorname{tr}(\Phi R) \\ \text{s.t.} \quad & \begin{bmatrix} R & E_K^{F\prime} H \\ H' E_K^F & H'E + E'H - I \end{bmatrix} \succeq 0, \end{aligned} \tag{5.24}$$

where $R \in \mathbb{S}^{2n_x+n_u}$ is a slack variable, $E_K^F = [E, -F, -K]$, and Φ denotes the empirical covariance matrix given by

$$\Phi := \sum_{t=1}^{T} \begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{x}_t \\ \tilde{u}_t \end{bmatrix} \begin{bmatrix} \tilde{x}_{t+1} \\ \tilde{x}_t \\ \tilde{u}_t \end{bmatrix}'. \tag{5.25}$$

System matrices $A \in \mathcal{S}_+$ and $B \geq 0$ can be recovered as $A = E^{-1}F$ and $B = E^{-1}K$.

Proof. Refer to Section 5.8.3. □

We conclude this section with two remarks. First, the LMI in (5.24) grows linearly with the size of the system; in particular, $R \in \mathbb{S}^{2n_x+n_u}$ is a dense, positive semidefinite matrix. Consequently, solving (5.24) directly may suffer from poor scalability. In Section 5.6.3 we show that minimization of $\bar{J}_\lambda$ can in fact be decomposed into many smaller subproblems; c.f. (5.42). Even without distributed computation, this decomposition can significantly reduce computation time for generic SDP solvers.

Second, it is of course possible to construct a Lagrangian relaxation of simulation error (LRSE), instead of equation error, as in Chapter 3. However, LR of simulation error cannot be decomposed into smaller subproblems in the same way as LR of equation error (LREE). Consequently, for large-scale systems the associated SDPs remain high dimensional, even with the specialized algorithms developed in Chapter 3. For this reason, we focus on LREE.

5.4.3 1-norm bounds on output error

The convex criteria considered so far can both be interpreted as weighted equation error. This is obvious for $\mathcal{E}_2^{\mathrm{w}}$ as defined in (5.11) with weighting matrix W as in (5.10). Furthermore, from (5.22), it is apparent that LR is also a procedure for constructing an alternative weighting of equation error, $\bar{J}_\lambda$. What is not clear, however, is the effect that these weighting matrices have on the quality of the identified model.

In this section, we provide convex conditions under which a simpler weighting matrix has a meaningful interpretation; specifically, the weight can be quantified as ℓ_1-gain from equation error to output error. To develop this result, we work with the 1-norm of equation error,

$$\mathcal{E}_1^{\mathrm{e}} := \sum\nolimits_{t=1}^{T} \|\tilde{x}_{t+1} - A\tilde{x}_t - B\tilde{u}_t\|_1 \tag{5.26}$$

and the 1-norm of *simulation error*, defined as

$$\mathcal{E}_1^{\mathrm{s}} := \sum\nolimits_{t=1}^{T} \|\tilde{y}_t - Cx_t - D\tilde{u}_t\|_1,$$

where x_t denotes the simulated state sequence satisfying $x_{t+1} = Ax_t + B\tilde{u}_t$ and $x_1 = \tilde{x}_1$. We use the 1-norm so as to leverage the *linear* dissipativity theory developed for positive systems. To study the relationship between $\mathcal{E}_1^{\mathrm{e}}$ and $\mathcal{E}_1^{\mathrm{s}}$, notice

$$\sum_{t=1}^{T} \|\eta_t + C\tilde{x}_t - Cx_t\|_1 \leq \sum_{t=1}^{T} \|\eta_t\|_1 + \sum_{t=1}^{T} \|C\tilde{x}_t - Cx_t\|_1. \tag{5.27}$$

As $\sum_t \|\eta_t\|_1$ is convex, we concentrate on $\sum_t \|C\tilde{x}_t - Cx_t\|_1$, and remark that the simulated states, $x_{1:T}$, and the estimated states, $\tilde{x}_{1:T}$, are in fact solutions to the same augmented dynamical system. Specifically, by considering the system

$$x_{t+1} = Ax_t + B\tilde{u}_t + v_t \tag{5.28}$$

with input v_t and initial condition $x_1 = \tilde{x}_1$, we observe:

i. for $v_t = 0$ the solution to (5.28) is x_t, i.e., the usual simulated state sequence.

ii. for $v_t = \epsilon_t$, the solution to (5.28) is $\tilde{x}_t$.

Recalling $\Delta_t = \tilde{x}_t - x_t$, it is clear that Δ_t satisfies the *incremental error dynamics* given by the system

$$\Delta_{t+1} = A\Delta_t + \epsilon_t, \tag{5.29a}$$

$$z = C\Delta_t. \tag{5.29b}$$

This analysis, much in the spirit of [150], relates the equation error, $\|\epsilon_t\|_1$, to the quantity of interest $\|C(\tilde{x}_t - x_t)\|_1$, via the ℓ_1-gain of the incremental error system in (5.29). Furthermore, observe that when the original system (5.1) is positive, i.e., $A \geq 0$, $C \geq 0$, so too are the incremental error dynamics in (5.29). The ℓ_1-gain of a positive system can be characterized as:

Lemma 5.3 ([31, Lemma 1])**.** Let (5.1) denote a positive system. The following statements are equivalent:

1. The matrix A is Schur and the ℓ_1-gain of $u \mapsto y$ is less than γ.

2. There exists $p \in \mathbb{R}_{++}^{n_x}$ such that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}' \begin{bmatrix} p \\ \mathbf{1} \end{bmatrix} < \begin{bmatrix} p \\ \gamma\mathbf{1} \end{bmatrix}. \tag{5.30}$$

Lemma 5.3 can be used to quantify the contribution of each 'channel' of the equation error ϵ to the simulation error:

Lemma 5.4. Given a stable, positive system of the form (5.1), the ℓ_1-gain from the ith input channel $\epsilon(i)$ to the output z of the incremental error system (5.29), is given by $p^*(i)$, where

$$p^* = \arg \min_{p \in \mathbb{R}^{n_x}_{++}} \sum_i p(i) \quad \text{s.t.} \quad p'A - p' + \mathbf{1}'C < 0. \tag{5.31}$$

Proof. Refer to Section 5.8.4. □

To leverage the result in Lemma 5.4 we propose the following convex parametrization of stable positive systems:

$$\begin{aligned} \bar{\Theta}_d := \{E, F, K, C, D : & E \in \mathbb{D}, F \geq 0, C \geq 0, D \geq 0, \\ & \mathbf{1}'F - \mathbf{1}'E + \mathbf{1}'C \leq -\delta \mathbf{1}'\}, \end{aligned} \tag{5.32}$$

and the following convex quality-of-fit metric:

$$\mathcal{E}_1^{\text{w}} := \sum_{t=1}^{T} \|E\tilde{x}_{t+1} - F\tilde{x}_t - K\tilde{u}_t\|_1 + \sum_{t=1}^{T} \|\eta_t\|_1, \tag{5.33}$$

i.e., the 1-norm of equation error, with ϵ_t weighted by E. For $\theta = (E, F, K, C, D)$ the problem $\min_{\theta \in \bar{\Theta}_d} \mathcal{E}_1^{\text{w}}$ is a convex linear program. Lemma 5.4 implies that E serves as both a stability certificate (i.e., the linear Lyapunov function, $V(x) = \mathbf{1}'Ex$) and a meaningful weighting of equation error. Specifically, E penalizes most heavily the 'channels' of ϵ_t that contribute most significantly to the simulation error, as $p^* = \text{diag}\,(E)$ encodes the ℓ_1-gain of $\epsilon \mapsto z$ in the incremental error system (5.29). In summary:

Theorem 5.4. For $\theta = (E, F, K, C, D) \in \bar{\Theta}_d$, with $\bar{\Theta}_d$ defined in (5.32), the function $\mathcal{E}_1^{\text{w}}(\theta)$, as defined in (5.33), is a convex upper bound on the 1-norm of output error, $\mathcal{E}_1^{\text{s}}$. The problem $\min_{\theta \in \bar{\Theta}_d} \mathcal{E}_1^{\text{w}}$ is a convex linear program, from which system matrices $A \in \mathcal{S}_+$ and $B \geq 0$ can be recovered as $A = E^{-1}F$ and $B = E^{-1}K$, resp., along with $C \geq 0$ and $D \geq 0$.

Proof. Refer to Section 5.8.5. □

Remark 5.1. The strict inequality in (5.31) has been replaced by a non-strict inequality in (5.32) to ensure that the constraints lead to a well posed optimization problem. The extent of the conservatism introduced by this approximation is characterized by δ, and can be made arbitrarily small.

5.5 Case studies

5.5.1 Comparison of convex parametrizations and quality-of-fit criteria

In this section we compare the different convex parametrizations of $\mathcal{S}_+$ outlined in Section 5.3, as well as the two main convex quality-of-fit criteria presented in Section 5.4, (i) *weighted equation error*, $\mathcal{E}_2^{\mathrm{w}}$, proposed in [94], c.f Section 5.4.1, and (ii) *Lagrangian relaxation of equation error*, $\bar{J}_\lambda$, developed in Section 5.4.2. In what follows, we will often discuss 'banded matrices'. A matrix A is said to have bandwidth n if $|j-i| > n$ implies $A(i,j) = 0$. Then each row and column will have $2n-1$ nonzero elements (except for the first/last n rows/columns).

We begin by empirically motivating the need for identification algorithms that ensure stability for large-scale systems. Table 5.1 records the prevalence of model instability during identification when using unconstrained least squares, as state dimension increases. The experimental procedure is as follows: (i) a system (A, B) of specified state dimension n_x is randomly generated using Matlab's `rand` function. A is banded, with bandwidth $= 5$ and spectral radius of 0.95, and $B = [I_{n_u}, \ldots, I_{n_u}, 0_{n_u \times m}]'$, with $n_u = \lfloor n_x/10 \rfloor$ and m specified for each experiment. (ii) the system is simulated for $T = 100$ time steps, and white noise $w \sim \mathcal{N}(0, \Sigma_w)$ is added to the simulated states to generate training data $\tilde{x}_{1:T}$. The covariance matrix Σ_w is diagonal, and each diagonal element is tuned to give SNR of 10dB, for each state. (iii) models are then fit with least squares, i.e. $(\tilde{A}, \tilde{B}) = \min_{A,B} \sum_t \|\epsilon_t\|_2^2$. The model is said to be unstable if $\tilde{A} \notin \mathcal{S}$. (iv) this process is repeated 1000 times for each n_x and m combination.

Table 5.1 – Percentage of models identified using least squares that were unstable (1000 experimental trials for each n_x and m combination). True models were randomly generated: A was banded (bandwidth $= 5$) with spectral radius of 0.95, and $B = [I_{n_u}, \ldots, I_{n_u}, 0_{n_u \times m}]'$, with $n_u = \lfloor n_x/10 \rfloor$. Training data was generated by simulating these models for $T = 100$ time steps, and corrupting the states with white noise (SNR= 10dB).

State dim., n_x	50	60	70	80	90	100	110	120	130	140	150
$m = 4n_u$	0	0.1	0.1	0.2	0.4	0.8	2.3	4.5	8.3	15.1	20.4
$m = 5n_u$	0.1	0.2	0.6	1.1	3.4	7.3	14.1	24.0	37.2	49.9	62.0
$m = 7n_u$	0.3	1.7	8.0	18.9	40.0	58.5	77.1	88.7	95.5	97.7	99.4
$m = 9n_u$	0.8	6.9	24.4	59.0	83.6	93.5	98.7	99.8	100	100	100

It is apparent that unstable models are identified by least squares with greater regularity for systems with larger state dimension, n_x. It is important to emphasize that, due to the banded structure of A, the ratio of the number of model parameters to the number of data points remains constant, regardless of n_x, i.e., each row of A contains 9 parameters, to be

fit from 100 scalar data points. Increased prevalence of model instability with n_x is not due simply to increased variance (associated with a larger parameter to data point ratio), but rather, the scaling of the network. In fact, there is a growing body of research on robustness of large-scale interconnected networks, c.f., [1, 90, 93, 265], as well as [200] for a study that specifically investigates how the robustness of a network changes with increased state dimension. Network topology has a significant influence on properties like robustness; indeed, the results in Table 5.1 demonstrate that model instability is particularly sensitive to m in the definition of B. For this reason, Table 5.1 should be interpreted as a motivating illustration, rather than a conclusive study.

Comparison of quality-of-fit criteria for large-scale systems

We now compare minimization of Lagrangian relaxation of equation error (henceforth, LREE), $\bar{J}_\lambda$, and weighted equation error (henceforth, EEW), $\mathcal{E}_2^{\mathrm{w}}$, for 'large-scale' systems, e.g., systems with hundreds or thousands of state variables. We begin with large-scale systems because model instability is more prevalent in this setting; c.f. Table 5.1. For systems with fewer states, model instability tends to arise only under 'extreme' circumstances, e.g., when the true systems are close to being unstable (i.e., spectral radius very close to unity), or when the training set contains very few data points (relative to the number of model parameters). In this section, we use the polytopic parametrization Θ_p of $\mathcal{S}_+$, as the ensuing optimization problems scale better (i.e., can be solved more quickly for high-dimensional systems compared to the LMI parametrizations of $\mathcal{S}_+$); c.f., Section 5.5.2 for a comparison of computation time. The LMI parametrizations of $\mathcal{S}_+$ are examined in the sequel.

The results in this section are generated using the following experimental procedure: (i) a stable positive system (of specified state dimension n_x and $n_u = \lfloor n_x/10 \rfloor$) is randomly generated using Matlab's `rand` function. A is banded, with bandwidth $= 5$ and spectral radius of 0.95; B is also sparse. We shall refer to this system as the 'true model'. (ii) the true model is simulated for T time steps, excited by $\tilde{u}_t = |w_t|$, $w_t \sim \mathcal{N}(0, I)$. White noise (SNR 10dB) is added to the simulated states to form the state estimates, i.e. $\tilde{x}_t = x_t + v_t$, $v_t \sim \mathcal{N}(0, \Sigma_v)$ (iii) a model is fit by ordinary, unconstrained least squares, i.e., minimization of (5.7). If this model is *not* unstable, i.e., if $\tilde{A} \in \mathcal{S}$, we return to (i) and generate a new dataset; (iv) if the least squares solution $\tilde{A}$ is unstable, we proceed with minimization of $\mathcal{E}_2^{\mathrm{w}}(\theta)$ and $\bar{J}_\lambda(\theta)$, for $\theta \in \Theta_p$. (v) this process is repeated 100 times. We use $C = [\mathbf{1}_{50}' \; 0]$ as the output mapping for both the true and identified models.

The results of this experimental procedure for varying state dimension n_x are depicted in Figures 5.1 and 5.2. Normalized simulation error, defined as $\sum_{t=1}^{T} \|y_t - \tilde{y}_t\|_2^2 / \sum_{t=1}^{T} \|\tilde{y}_t\|_2^2$ where y_t is the simulated output of the identified model and $\tilde{y}_t$ is validation data from the true model, is used to quantify model fit. It is apparent that LREE significantly outperforms EEW (on average) for each value of n_x.

Figure 5.3 presents the results of the same experimental procedure, for fixed state dimen-

sion ($n_x = 500$) but varying training dataset length, T. Once again, LREE significantly outperforms EEW. Furthermore, performance of both methods improves with increasing T, as expected.

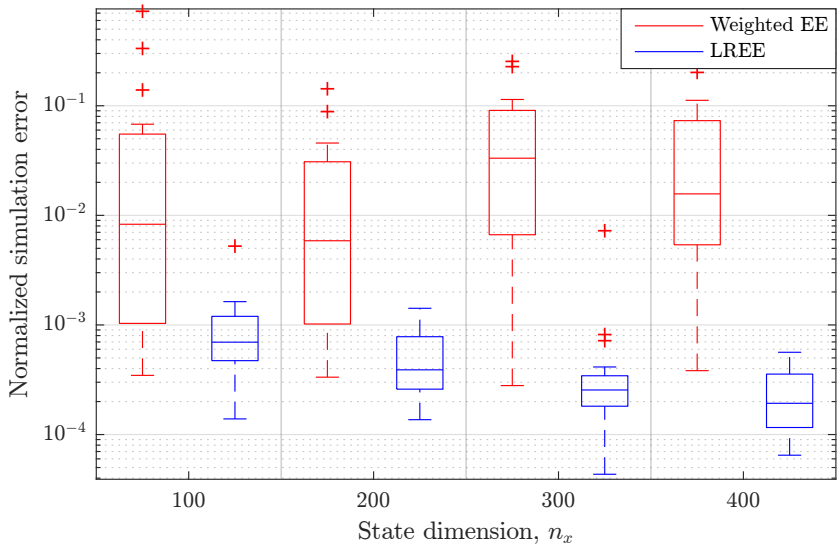


Figure 5.1 – Normalized simulation error vs. state dimension of the true system, n_x. Two methods are compared: (i) *Weighted EE*: $\min_{\theta\in\Theta_p} \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta_p} \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. The polytopic parametrization of $\mathcal{S}_+$, Θ_p, is used for both. Models have banded A (bandwidth of 5). $T = 100$ data points were used for training; the SNR was 10dB. The spectral radius of the true models was 0.9.

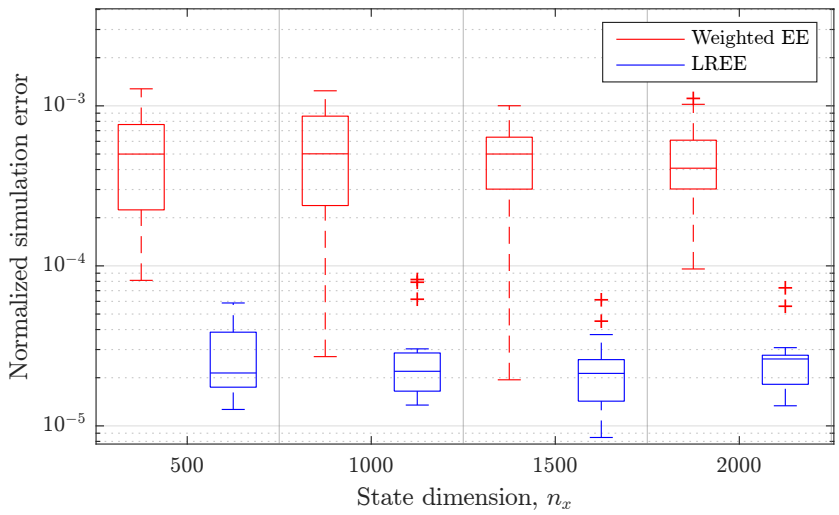


Figure 5.2 – Normalized simulation error vs. state dimension of the true system, n_x. Two methods are compared: (i) *Weighted EE*: $\min_{\theta\in\Theta_p} \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta_p} \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. The polytopic parametrization of $\mathcal{S}_+$, Θ_p, is used for both. Models have banded A (bandwidth of 5). $T = 200$ data points were used for training; the SNR was 10dB. The spectral radius of the true models was 0.9.

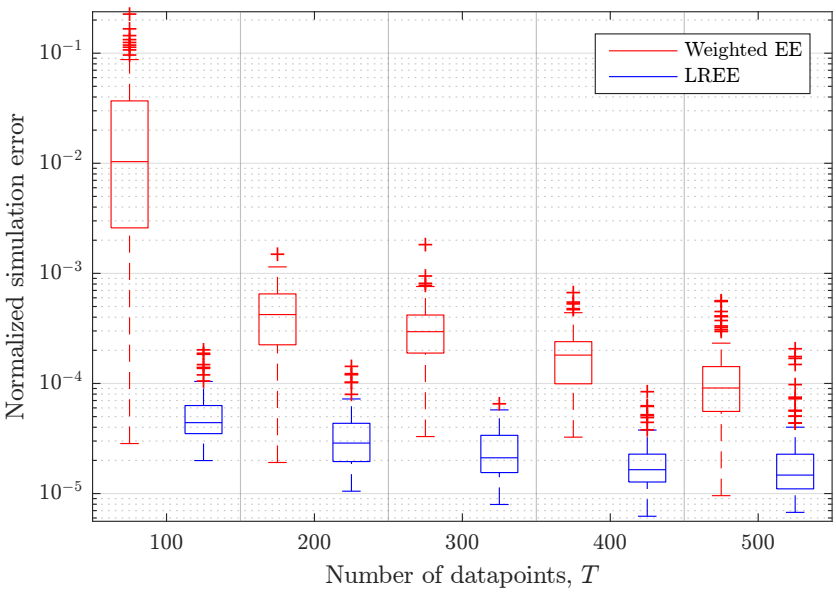


Figure 5.3 – Normalized simulation error vs. length of training data set, T. Two methods are compared: (i) *Weighted EE*: $\min_{\theta\in\Theta_p} \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta_p} \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. The polytopic parametrization of $\mathcal{S}_+$, Θ_p, is used for both. Models have banded A (bandwidth of 5) with $n_x = 500$. The SNR was 10dB, and the spectral radius of the true models was 0.9.

Influence of least squares solution

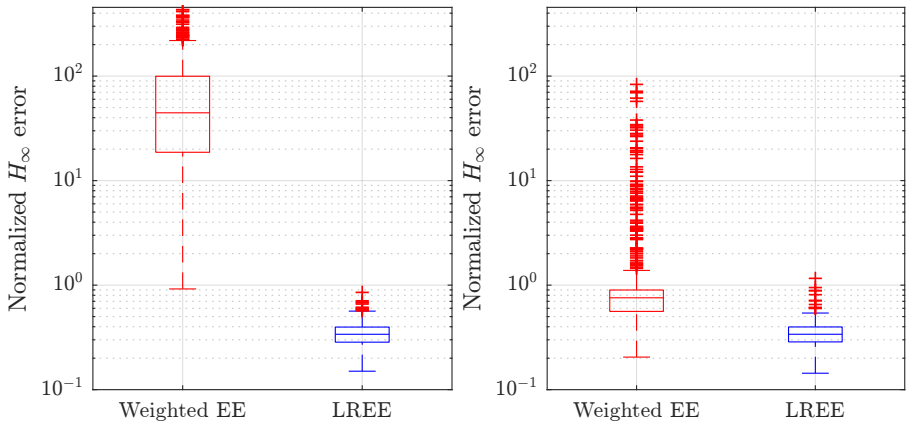
The results in Figures 5.1, 5.2, and 5.3 suggest that LREE offers improved performance over weighted EE. In this section, we attempt to gain deeper insight into the differences between the two methods. The same experimental procedure as the previous section is repeated, but with the following differences: (a) the dimension of the true models is reduced to $n_x = 5$. Furthermore, A and B are both dense (not banded), and $n_u = 1$. (b) to compensate for the reduced state dimension, the spectral radius of the true A is increased to 0.98. (c) we use the LMI parametrization of $\mathcal{S}_+$, Θ_l as in (5.2); however, the conclusions we draw are valid for the parametrizations Θ_m and Θ_p, as the empirical results are qualitatively similar; c.f., e.g., Figure 5.5. (d) we use $C = \mathbf{1}'$ as the output mapping for both the true and identified models. (e) the experimental procedure is repeated 1000 times.

The results are presented in Figure 5.4. We use normalized H_∞ error as a measure of model fit; the results are qualitatively similar for H_2 error, c.f. Figure 5.5. Figures 5.4(a) and (b) plot normalized H_∞ error for cases in which the least squares solution was nonnegative (i.e., $\tilde{A} \geq 0$) and *not* nonnegative (i.e., $\tilde{A} \ngeq 0$), respectively. The performance of EEW in the first case is very bad (e.g., the 'best' models have normalized H_∞ error of unity, which is no better than no model at all). Performance of EEW in the second case is somewhat improved; although LREE is clearly superior in both cases. To understand this behavior,

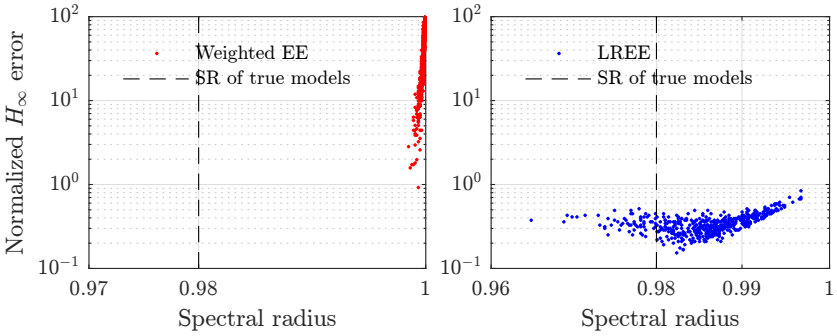
consider once more minimization of weighted equation error as in (5.13),

$$\min_{E,F} \|\tilde{A}E - F\|_F^2, \quad \text{s.t. } (E,F) \in \Theta_l. \tag{5.34}$$

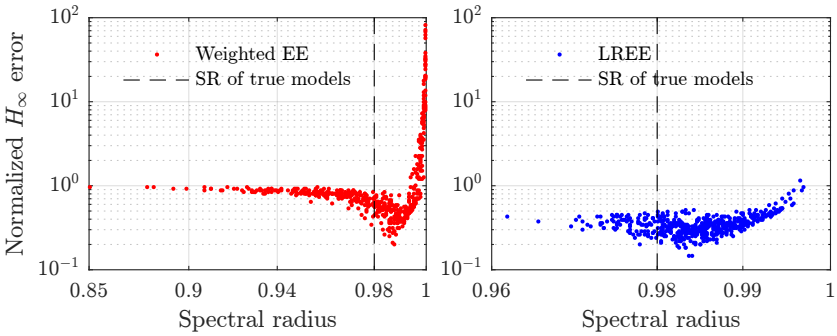
This can be interpreted as a projection of the least squares solution $\tilde{A}$ onto the parametrization of $\mathcal{S}_+$ given by Θ_l. When $\tilde{A} \geq 0$ but not stable, this projection moves the poles of $A = E^{-1}F$ 'just' inside the unit circle. This leads to a model with spectral radius that is far too close to unity, leading to extremely large error. This is exactly what we observe in Figure 5.4(c). When $\tilde{A} \ngeq 0$ but unstable, the projection also has to correct the negative elements in $\tilde{A}$, and the poles often end up a little further inside the unit circle, which *may* improve performance; c.f. Figure 5.4(d). In summary, the performance of EEW is highly sensitive to the least squares solution $\tilde{A}$, as might be expected. Conversely, LREE appears largely insensitive to $\tilde{A}$, as there is essentially no difference between Figure 5.4(c) and (d). As stated above, these observations tend to hold true all three parametrizations of $\mathcal{S}_+$, i.e., Θ_m, Θ_l, and Θ_p.



(a) Least squares nonnegative, $\tilde{A} \geq 0$. **(b)** Least squares not nonnegative, $\tilde{A} \ngeq 0$.



(c) Least squares nonnegative, $\tilde{A} \geq 0$. Note the different x-axis scale.



(d) Least squares not nonnegative, $\tilde{A} \ngeq 0$. Note the different x-axis scale.

Figure 5.4 – Sensitivity of two methods to the unconstrained least squares solution, $\tilde{A}$: (i) *Weighted EE*: $\min_{\theta \in \Theta_l} \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta \in \Theta_l} \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. 'Least squares not nonnegative' means that at least one entry of $\tilde{A}$ is strictly negative. Weighted EE is highly sensitive to $\tilde{A}$: the spectral radius (SR) is systematically overestimated when $\tilde{A} \geq 0$. LREE appears to be insensitive to $\tilde{A}$; compare (c) and (d). The SRs of the true models were 0.98, with $n_x = 5$. $T = 100$ data points were used for training; the SNR was 10dB.

Table 5.2 – Comparison of Lagrangian relaxation of equation error (LREE) with two parametrizations of $\mathcal{S}_+$, Θ_m defined in (5.5) and Θ_l defined in (5.2). For each metric (the bound $\bar{J}_\lambda$, H_∞ error and H_2 error) the number recorded is the ratio of the median value of LREE with Θ_m to median value of LREE with Θ_l, i.e., numbers less than unity indicate better performance with Θ_m.

Spectral radius	0.97	0.98	0.99	0.999	0.999	0.999	0.9999
SNR (dB)	5	10	10	10	22	30	30
$\bar{J}_\lambda$	0.997	0.994	0.993	0.989	0.994	0.995	0.995
H_∞ error	1.000	1.009	1.004	0.999	1.097	0.575	1.051
H_2 error	1.028	1.001	0.998	0.997	1.126	0.615	1.070

Effect of spectral radius

We continue our comparisons of convex parametrizations and quality-of-fit metrics with an investigation of the effect that spectral radius (SR) of the true model has on performance. To generate some baseline results, we begin with identification of 10th order systems ($n_x = 10$, $n_u = 2$) with SR equal to 0.97. Figures 5.5 and 5.6 record the normalized H_2 and H_∞ error, respectively, for the two different criteria, $\mathcal{E}_2^{\text{w}}$ and $\bar{J}_\lambda$, and three different parametrizations of $\mathcal{S}_+$, Θ_m, Θ_l and Θ_p. From these figures we make the following observations. First, LREE significantly outperforms WEE for all parametrizations of $\mathcal{S}_+$. Second, it is apparent that for in this experiment choice of parametrization has little influence on performance; in particular, the difference between the two LMI parametrizations Θ_m and Θ_l is negligible; c.f. also Table 5.2. Henceforth, we shall consider Θ_l only. Finally, we can gain further insight into these results by plotting error as a function of the SR of the identified models, as in Figure 5.6(b) and (c), for Θ_l and Θ_p, respectively. The plot for Θ_m is omitted due to similarity with Θ_l. From Figure 5.6(b), we observe that EEW tends to underestimate the SR; furthermore, these overestimates are correlated with larger error. Conversely, models from LREE tend to have SR closer to the true value. The situation is slightly different for Θ_p. From Figure 5.6(c), it is apparent that EEW continues to underestimate the SR; however, it is also clear that LREE tends to overestimate the SR, and that these overestimates are correlated with larger error. This is a trend that we will continue to observe in the sequel.

Next, we examine the performance of each parametrization for increasing spectral radius of the true model. In Figure 5.7, we record performance (H_∞ error) of the LMI parametrization Θ_l and make the following observations. First, LREE performs significantly better than EEW, for all values of true spectral radius (although there are a few outliers at SR = 0.999). Second, although performance of both methods degrades (H_∞ error gets larger) with increasing true SR, degradation of EEW is much more pronounced; for SR = 0.999 the normalized H_∞ error is approximately unity. For relatively small SR, e.g. 0.98 as in Figure 5.7(b), we observe EEW systematically underestimating the SR, as in Figure 5.6(b). However, for larger SR, e.g., 0.999 as in Figure 5.7(c), the situation is different. EEW still

underestimates the SR, but now there is no correlation between H_∞ error and SR; all models are equally bad, regardless of SR. For LREE, there is a very strong relationship between estimated SR and H_∞ error. In most trials, large H_∞ error is due to underestimation of the SR; however, there are a few cases in which the SR is overestimated. Indeed, it is these cases that correspond to particularly bad models, i.e., the outliers in Figure 5.7(a).

In Figure 5.8 we record the performance of the polytopic parametrization Θ_p for varying true SR. From Figure 5.8(a), we see that LREE performs better than EEW for SR of 0.98 and 0.99, but considerably worse for larger SR of 0.999 and 0.9999. Let us first discuss the performance for SR of 0.98 and 0.99. From 5.8(b), we see, once more, that error in EEW is correlated with underestimating the SR; compare with 5.7(b). Performance of LREE is better, but not quite as good as LREE with Θ_l. Comparing 5.8(b) with 5.7(b) it is apparent that LREE with Θ_p is more likely to overestimate the SR, and these overestimates correlate with greater H_∞ error. Let us now turn to SR of 0.999 and 0.9999. The first thing to emphasize is that performance of EEW is very bad (normalized H_∞ error of unity), even though, on average, it is better than LREE. Figure 5.8(c) sheds some light onto the behavior of LREE. Here, the propensity for LREE with Θ_p to overestimate the SR leads to extremely poor performance. For trials in which the SR was estimated accurately, performance is quite acceptable.

In Figure 5.9, we repeat the experiment for SR of 0.999 and 0.999, but with a lower SNR of 10dB. The additional noise removes the propensity for LREE with Θ_p to overestimate the SR, c.f. Figure 5.9(b), and models with very large H_∞ are no longer identified. It is worth pointing out that these difficulties arise when the true system has very large SR, i.e., the system almost contains a pure integrator. In such a setting, some additional preprocessing of the signals (e.g. differentiation) may be appropriate.

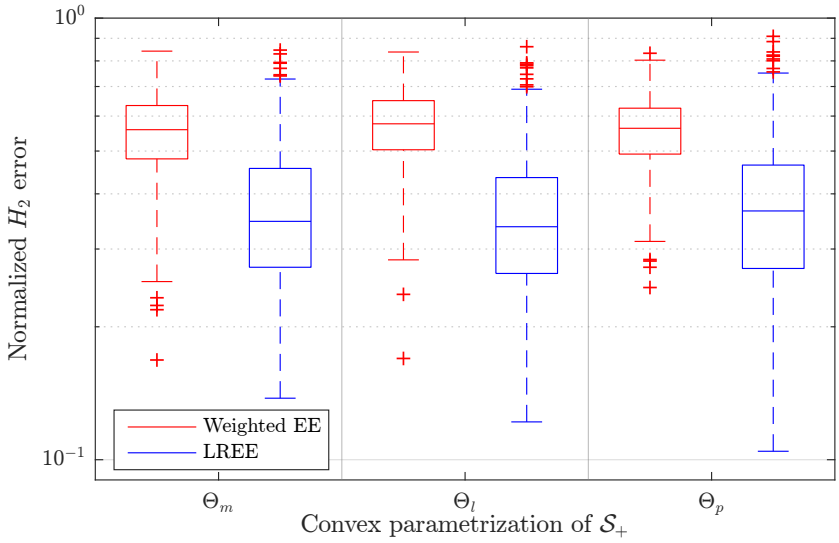
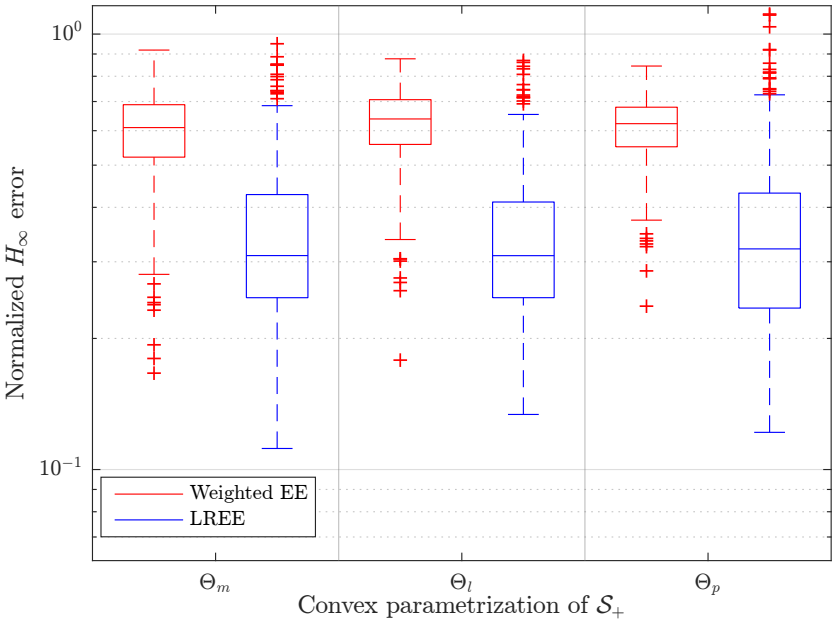
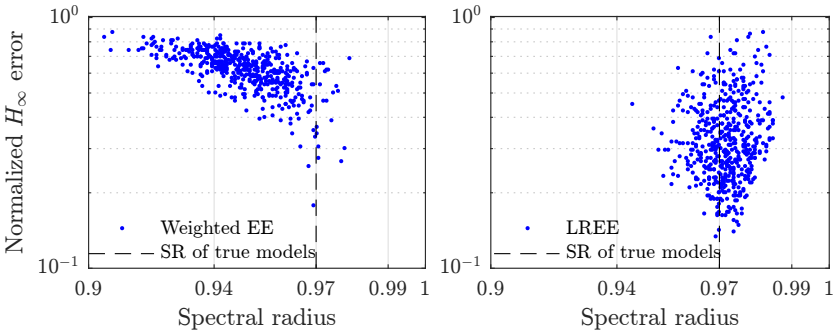


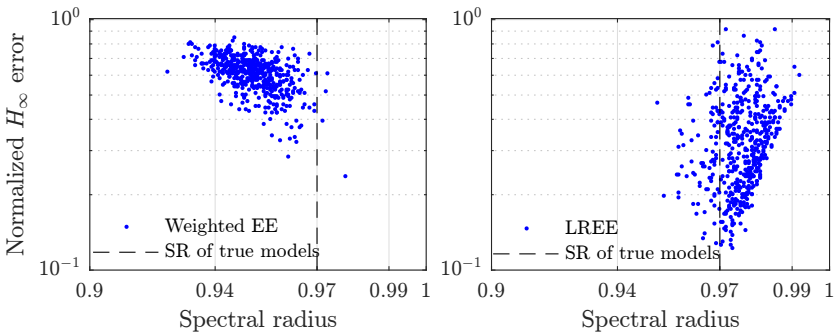
Figure 5.5 – Normalized H_2 error for two methods: (i) *Weighted EE*: $\min_{\theta\in\Theta}\ \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta}\ \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. All three parametrizations of $\mathcal{S}_+$ are used: Θ_m (LMI, c.f. (5.4)), Θ_l (LMI, c.f. (5.2)), and Θ_p (polytopic, c.f. (5.6)). True models are 10th order ($n_x = 10$, $n_u = 2$) with SR of 0.97. $T = 100$ data points were used for training (SNR of 5dB). 500 trials are depicted for each method/parametrization.



(a) True spectral radius: 0.97. SNR: 5dB.

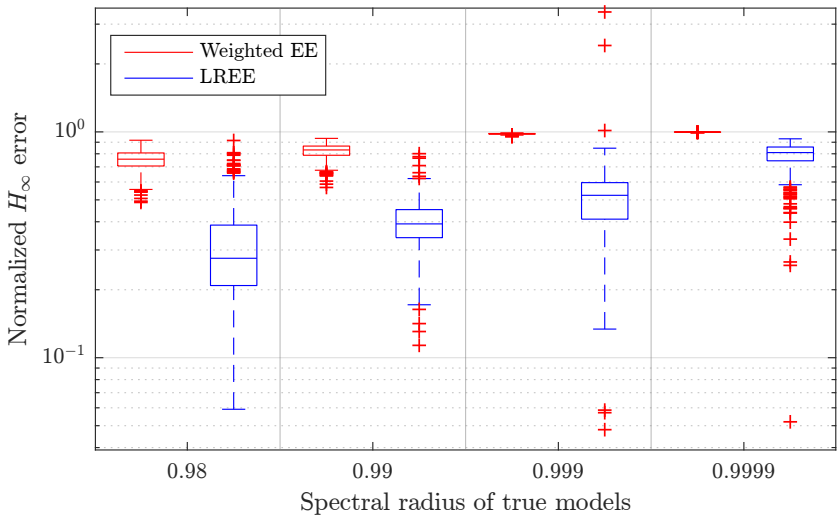


(b) Parametrization: Θ_l (LMI), c.f., (5.2).

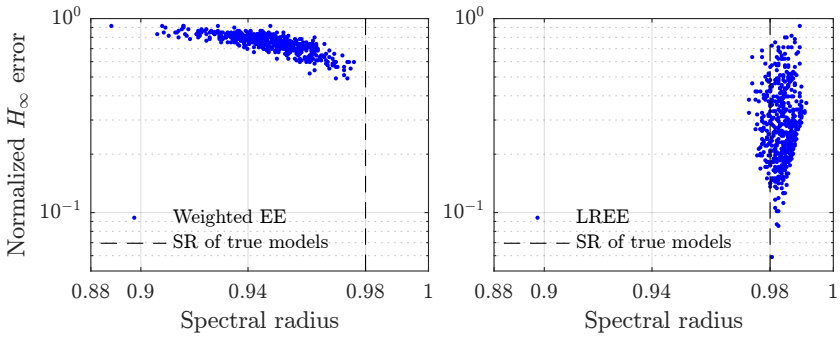


(c) Parametrization: Θ_p (polytopic), c.f., (5.6).

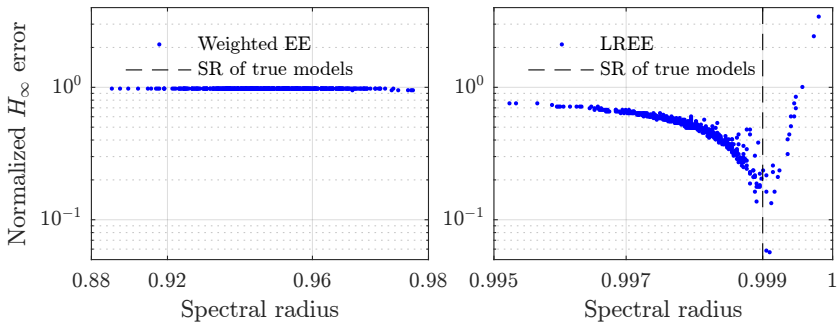
Figure 5.6 – Normalized H_∞ error for two methods: (i) *Weighted EE*: $\min_{\theta\in\Theta}\ \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta}\ \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. All three parametrizations of $\mathcal{S}_+$ are used: Θ_m (LMI, c.f. (5.4)), Θ_l (LMI, c.f. (5.2)), and Θ_p (polytopic, c.f. (5.6)). The experimental setup is identical to that described in Figure 5.5. True models are 10th order ($n_x = 10$, $n_u = 2$) with SR of 0.97. $T = 100$ data points were used for training (SNR of 5dB). 500 trials are depicted for each method/parametrization.



(a) The SNR used for SR of 0.98, 0.99, 0.999 and 0.9999 were 10dB, 10dB, 22dB and 30dB, respectively.

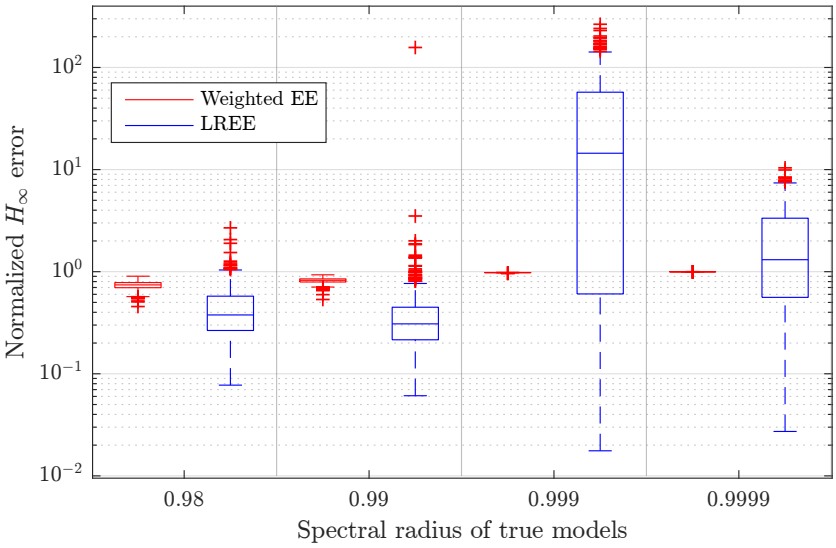


(b) True spectral radius: 0.98.

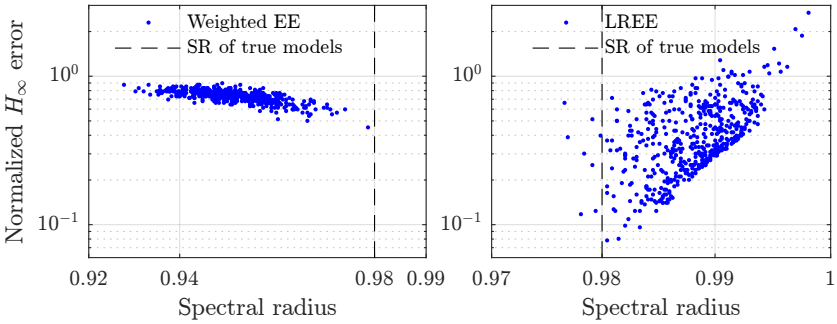


(c) True spectral radius: 0.999. Note the different x-axis scale.

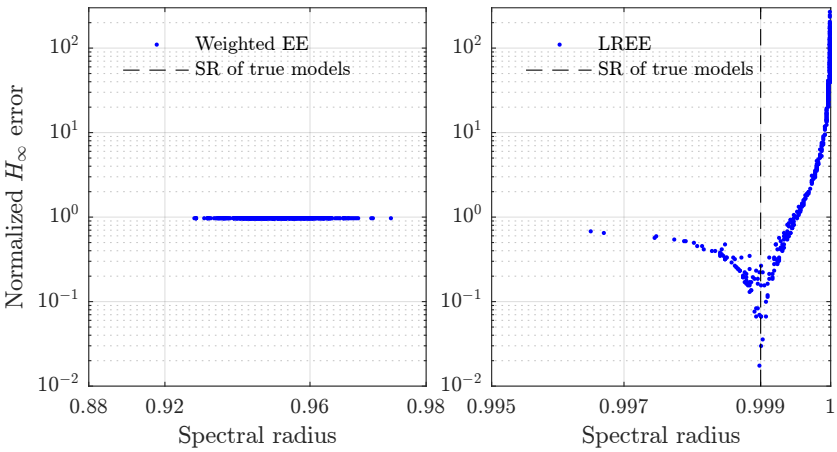
Figure 5.7 – Normalized H_∞ error for two methods: (i) *Weighted EE*: $\min_{\theta\in\Theta_l} \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta_l} \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. The LMI parametrization Θ_l of $\mathcal{S}_+$ is used, c.f. (5.2)). True models are 10th order ($n_x = 10$, $n_u = 2$), and $T = 100$ data points were used for training. 500 trials are depicted for each SR.



(a) The SNR used for SR of 0.98, 0.99, 0.999 and 0.9999 were 10dB, 10dB, 22dB and 30dB, respectively.

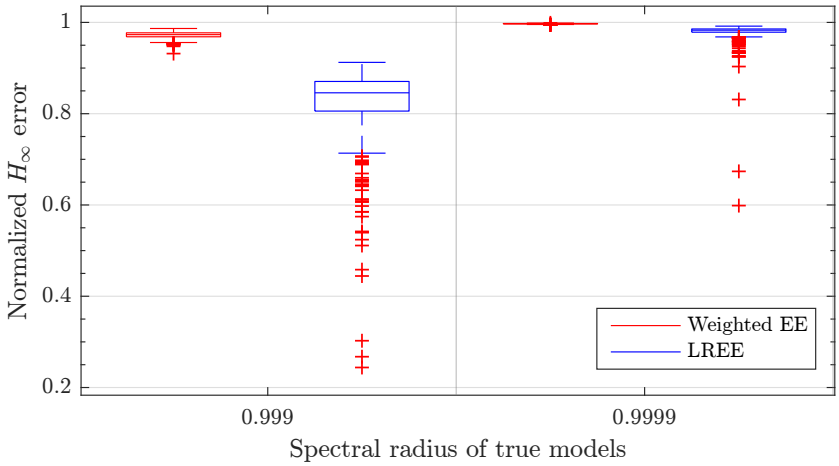


(b) True spectral radius: 0.98. Note the different x-axis scale.

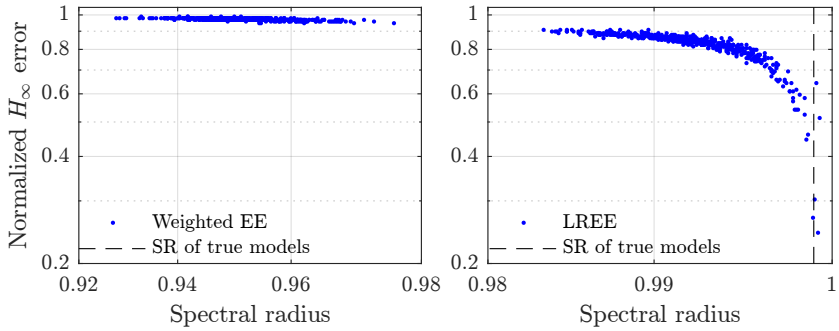


(c) True spectral radius: 0.999. Note the different x-axis scale.

Figure 5.8 – Normalized H_∞ error for two methods: (i) *Weighted EE*: $\min_{\theta\in\Theta_p} \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta_p} \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. The polytopic parametrization Θ_p of $\mathcal{S}_+$ is used, c.f. (5.6)). True models are 10th order ($n_x = 10$, $n_u = 2$), and $T = 100$ data points were used for training. 500 trials are depicted for each SR.



(a) The SNR for both cases was 10dB.



(b) True spectral radius: 0.999. Note the different x-axis scale.

Figure 5.9 – Normalized H_∞ error for two methods: (i) *Weighted EE*: $\min_{\theta\in\Theta_p}\ \mathcal{E}_2^{\mathrm{w}}(\theta)$, c.f. Section 5.4.1, (ii) *LREE*: $\min_{\theta\in\Theta_p}\ \bar{J}_\lambda(\theta)$, c.f. Section 5.4.2. The polytopic parametrization Θ_p of $\mathcal{S}_+$ is used, c.f. (5.6)). True models are 10th order ($n_x = 10$, $n_u = 2$), and $T = 100$ data points were used for training. 500 trials are depicted for each SR.

5.5.2 Scalability

In this section we illustrate the difference in scalability between the LMI parametrization (Θ_l) of $\mathcal{S}_+$ introduced in [94], c.f. (5.2), and the polytopic parametrization (Θ_p) introduced in Section 5.3.2, c.f., (5.6). Specifically, we compare the following four optimization problems,

i. Weighted EE (polytopic, Θ_p), i.e., the QP $\min_\theta\ \mathcal{E}_2^{\mathrm{w}}(\theta)$ s.t. $\theta \in \Theta_p$,

ii. Weighted EE (LMI, Θ_l), i.e., the SDP $\min_\theta\ \mathcal{E}_2^{\mathrm{w}}(\theta)$ s.t. $\theta \in \Theta_l$,

iii. LREE (polytopic, Θ_p), i.e., the SDP $\min_\theta\ \bar{J}_\lambda(\theta)$ s.t. $\theta \in \Theta_p$,

iv. LREE (LMI, Θ_l), i.e., the SDP $\min_\theta\ \bar{J}_\lambda(\theta)$ s.t. $\theta \in \Theta_l$.

Each experimental trial consists of the following steps: (i) a stable positive system with state dimension n_x is randomly generated using Matlab's `rand` function; $A \in \mathbb{R}^{n_x \times n_x}$ has a banded structure with bandwidth equal to 5. (ii) the system is simulated for $T = 10^4$ time steps; $\tilde{x}_{1:T}$ is obtained by adding white noise to the simulated states at SNR equal to 20dB. (iii) Each of the four methods listed above are run on a desktop computer (Intel i7, 3.40CHz, 8GB RAM). Programs using the LMI parametrization are formulated with Yalmip [133], whereas programs using the polytopic parametrization are formulated with SPOT [233]. All programs are solved with Mosek v7.0.0.119. (iv) this process is repeated 5 times for each n_x.

The results are presented in Figure 5.10, from which we make the following observations. Foremost, it is clear that the methods using the polytopic parametrization have superior scalability compared to the LMI parametrization. Specifically, Weighted EE (Θ_p) and LREE (Θ_p) each have complexity that grows approx. linearly with state dimension; the approaches using Θ_l have complexity that grows approx. cubicly. Furthermore, the methods using Θ_l begin to exhaust available memory between $n_x = 1100$ and $n_x = 2000$. The methods using Θ_p can handle $n_x > 10^4$ before memory becomes a limitation. Finally, we note that no explicit attempts to exploit the sparsity of the system were made; use of solvers and parsers designed to exploit sparsity could improve performance (i.e. reduce computationally complexity), especially for the SDPs associated with the LMI parametrization, c.f., e.g., [6].

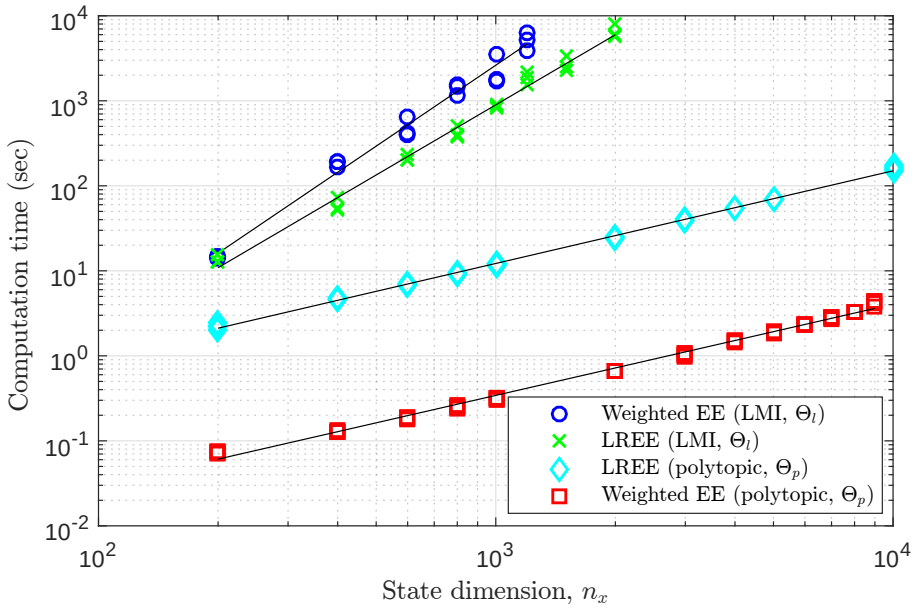


Figure 5.10 – Computation time as a function of system size, for four identification strategies; c.f. Section 5.5.2 for details. For each value of n_x, 5 trials were conducted. The slopes of the lines of best fit are: 1.08 (EEW, Θ_p), 3.16 (EEW, Θ_l), 1.09 (LREE, Θ_p), and 2.74 (LREE, Θ_l).

5.5.3 Identification of structured systems

In this section, we illustrate the utility of the bound on simulation error developed in Section 5.4.3 for identification of structured systems. Incorporating a priori structural information is recognized as a central problem in the industrial application of system identification algorithms [131, §6]; e.g., it may be known that two identical components are connected in series, and we would like our model to respect this structural property. To enforce model stability under structural constraints of the form $A(i,j) = A(k,l)$, with a parametrization $F = EA$ for diagonal E, we require $F(i,j)/E(i,i) = F(k,l)/E(k,k)$, which is not jointly convex in F and E.

To circumvent this nonconvexity, one may consider a two-step approach, solving first (5.35a) and then (5.35b):

$$\{E_d, \star\} := \arg\min_{\theta \in \bar{\Theta}_d} \ \mathcal{E}_1^{\mathrm{w}}(\theta) \tag{5.35a}$$

$$\begin{aligned} \Sigma_d := \{A_d, B_d, C_d, D_d\} = \arg \min_{\{A,B,C,D\} \geq 0} \ & \mathcal{E}_1^{\mathrm{w}}(\theta) \\ \text{s.t.} \ \ & \mathbf{1}' E_d A - \mathbf{1}' E_d + \mathbf{1}' C \leq -\delta \mathbf{1}', \\ & A(i,j) = A(k,l), \ F = E_d A, \ K = E_d B, \end{aligned} \tag{5.35b}$$

for all i, j, k, l for which we wish to enforce such constraints, based on *a priori* structural knowledge. Here, $\theta = \{E, F, K, C, D\}$ with $E \in \mathbb{D}$, and $\bar{\Theta}_d$ is defined in (5.32). The idea is that (5.35a) furnishes us with an approximation of the ℓ_1 gain from equation error to simulation error, $\mathrm{diag}\,(E_d)$. Then, in (5.35b), E is fixed as $E = E_d$ such that the structural constraints $A(i,j) = A(k,l)$ are convex in A.

To demonstrate the advantages of enforcing $\theta \in \bar{\Theta}_d$ over the regular polytopic stability condition $\theta \in \Theta_p$, c.f. (5.6), we compare the approach in (5.35) to the following similar two-step procedure:

$$\{E_s, \star\} := \arg\min_{\theta \in \Theta_p} \ \mathcal{E}_1^{\mathrm{w}}(\theta) \tag{5.36a}$$

$$\begin{aligned} \Sigma_s := \{A_s, B_s, C_s, D_s\} = \arg \min_{\{A,B,C,D\} \geq 0} \ & \mathcal{E}_1^{\mathrm{w}}(\theta) \\ \text{s.t.} \ \ & \mathbf{1}' E_s A - \mathbf{1}' E_s \leq -\delta \mathbf{1}', \\ & A(i,j) = A(k,l), \ F = E_s A, \ K = E_s B, \end{aligned} \tag{5.36b}$$

Notice that in (5.35) we enforce the dissipation inequality $p'A - p' + 1'C \leq -\delta 1'$, whereas in (5.36) we simply enforce the stability condition $p'A - p' < 0$. The difference between these two conditions is illustrated by identifying a model Σ with structure of the form:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, B = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix}, C = \begin{bmatrix} C_1 & 0 \end{bmatrix}$$

where $A_{11} = A_{22}$ and $B_1 = B_2$. The interpretation is that Σ represents two identical subsystems, that are coupled by A_{12} and A_{21}, however only the first subsystem contributes directly to the measured plant output.

We generate a dataset from a model $\tilde{\Sigma}$ with parameters:

$$\begin{aligned} A_{11} &= [0.2, 0.7; 0.5, 0.4], \;\; A_{22} = [0.5, 0.2; 0.4, 0.7], \\ A_{12} &= [0, 0.001; 0, 0.02], \;\; A_{21} = [0.01, 0.05; 0.03, 0.01], \\ B_1 &= [0.1; 0], \;\; B_2 = [0.2, 0], \;\; C_1 = [1, 0]. \end{aligned}$$

To form the training dataset $\tilde{y}$ and $\tilde{x}$ were taken to be the true simulated quantities, corrupted by additive Gaussian noise. Notice that $\tilde{\Sigma}$ is not in the model class, as there is some mismatch between the two subsystems that are assumed to be identical.

The normalized (simulation) error of each identified model, defined as $\mathcal{E}_1^{\mathrm{s}} / \sum_t |\tilde{y}_t|$, for 1000 experimental trials is plotted in Fig. 5.11, from which it is evident that (5.35) outperforms (5.36). Greater insight can be gained by studying Fig. 5.12, which depicts the equation error of Σ_d (from (5.35)) and Σ_s (from (5.36)), for a typical experimental trial. The ℓ_1 gain from ϵ to $\mathcal{E}_1^{\mathrm{s}}$ for the true model $\tilde{\Sigma}$ is given by $[4.66, 5.44, 0.09, 0.11]$, for each state, respectively; i.e. equation error in x_1 and x_2 contribute most significantly to simulation error. In this trial, $\mathrm{diag}\,(E_d) = [4.11, 4.51, 0.25, 0.30]$ and $\mathrm{diag}\,(E_s) = [1.03, 1.34, 1.11, 1.09] \times 10^{-4}$. As E functions as a weight on equation error, (5.35) will prioritize minimization of equation error in x_1, x_2, at the expense of poor fit in x_3, x_4; c.f. Fig. 5.12. This accounts for the superior performance of Σ_d over Σ_s, in Fig. 5.11.

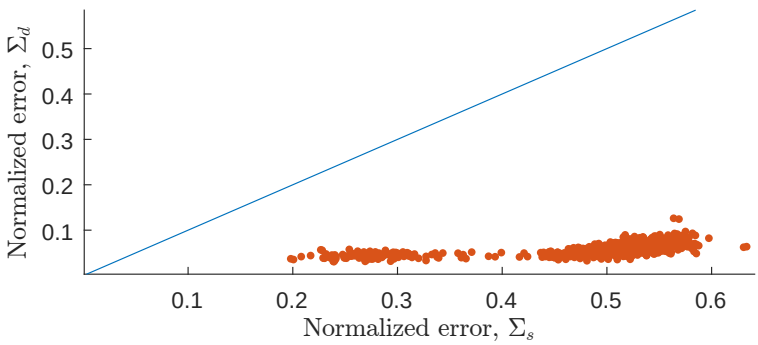


Figure 5.11 – Normalized simulation error for models Σ_d, identified by (5.35), and Σ_s, identified by (5.36). 1000 experimental trials were conducted.

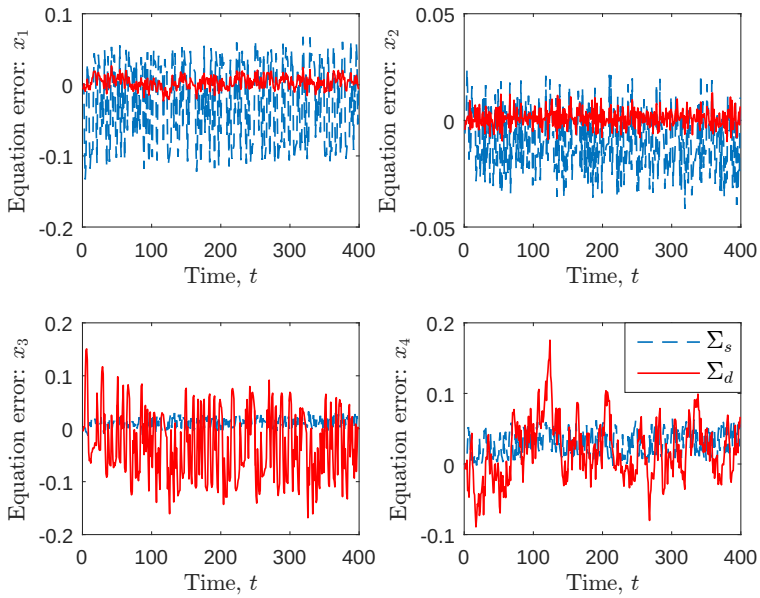


Figure 5.12 – Equation error for Σ_d from (5.35), and Σ_s from (5.36). The approach of (5.35) detects that equation error in x_1 and x_2 contributes most significantly to simulation error, and returns a model that achieves good fit in these states, at the expense of poor fit in x_3 and x_4, due to structural constraints.

5.6 Algorithms for distributed identification

In this section we exploit the polytopic parametrization of stable positive systems, Θ_p, introduced in Section 5.3.2 and present scalable distributed algorithms for minimization of the quality-of-fit criteria derived in Section 5.4.

5.6.1 Problem scope and set-up

Distributed optimization is a rich subject that finds application in a variety of fields, each with their own goals, assumptions and terminology. Before proceeding, let us clarify the scope of the problem that we consider in this section. The main challenge that we address is scalability of the search for a model subject to stability constraints. Consider minimization of equation error (in the state), $\mathcal{E}_2^{\mathrm{e}}$, as defined in (5.8). Equation error can be trivially decomposed as

$$\mathcal{E}_2^{\mathrm{e}} = \sum_{i=1}^{n_x} \left\{ \sum_{t=1}^{T} |\tilde{x}_{t+1}(i) - A(i,:)\tilde{x}_t - B(i,:)\tilde{u}_t|^2 \right\}, \tag{5.37}$$

i.e., the sum of n_x independent cost functions, each of which depend only on the decision variables associated with the ith row of A and B. In the absence of any stability constraints, minimization of (5.37) can be carried out in a distributed way, in the sense that $A(i,:)$ and $B(i,:)$ can be fit independently of $A(j,:)$ and $B(j,:)$, $i \neq j$. Model stability constraints

introduce coupling between rows. For convex parametrizations of stable models based on LMIs, e.g. Θ_m and Θ_l, decoupling these constraints is nontrivial. When A is sparse, as is often the case for networks of dynamical systems, these LMIs may inherit a sparsity pattern characterized by a chordal graph. Chordal sparsity permits a decomposition of the problem for which more efficient solution methods are applicable, c.f., e.g., [139, 244]. In general, identification of large-scale systems with non-sparse A, subject to LMI stability constraints, leads to large, dense SDP that are computationally intractable.

For positive systems, the situation is fundamentally different. In the polytopic parametrization of stable models Θ_p, each column of the linear stability condition $\mathbf{1}'F - \mathbf{1}'E < 0$ in (5.6) can be checked independently; i.e., stability is verified by $i = 1, \ldots, n_x$ linear inequalities, each of which depends only on the ith column of F. In other words, the linear stability condition decomposes into simpler independent conditions, even when A (and, therefore, F) is non-sparse.

With this in mind, the purpose of this section is to make explicit the ways in which the polytopic parametrization Θ_p of $\mathcal{S}_+$ permits distributed optimization of the convex quality-of-fit criteria outlined in Section 5.4. By distributed , we mean decomposition of the problem into minimization of the sum of many 'local' cost functions, each of which depends only on a subset of the decision variables, for the purpose of improved scalability (namely: methods that require less memory). The problems we consider are not always completely separable; i.e., the stability constraints may couple the variables associated with each local cost function. In our context, this coupling is due to the fact that equation error permits a 'row-based' decomposition, as in (5.37), whereas the stability condition (5.6) allows for a 'column-based' decomposition, when formulated with $F = EA$.

As discussed, for positive systems decomposability of the stability constraints does not depend on sparsity. Nevertheless, sparsity of A (and B) is common when modeling large-scale networks of dynamical systems; e.g., when each subsystems is only directly connected to a few neighboring subsystems within the network. When a sparse system representation is desired, we shall assume that the sparsity patterns (i.e., locations of the nonzero elements) of A and B are known *a priori*. This is a strong assumption, not valid in all applications; however, there are many settings in which the topology of the network is explicitly known, e.g., the interconnection of roads. Furthermore, the problem of inferring the topological structure of networks of dynamical systems from measured data has received increased attention as of late, c.f., e.g., [146, 205]. To characterize the sparsity of A, we define the 'row neighbors' of the ith row, and 'column neighbors' of the ith column by $n_i^{ar} := \{j : A(i,j) \neq 0\}$, and $n_i^{ac} := \{j : A(j,i) \neq 0\}$, respectively. The row and column neighbors of B are given by $n_i^{br} := \{j : B(i,j) \neq 0\}$, and $n_i^{bc} := \{j : B(j,i) \neq 0\}$, respectively.

Hitherto, we have discussed separability of the identification problem in terms of the decision variables. It is worth emphasizing that each 'local' cost function in (5.37) depends on the entire state and input sequence, $\tilde{x}_{1:T}$ and $\tilde{u}_{1:T}$, respectively. When the system is sparse, the ith local cost function depends on the states $\tilde{x}_{1:T}(n_i^{ar})$ and inputs $\tilde{u}_{1:T}(n_i^{br})$ associated

with its row neighbors. If each local cost function is assigned its own processor, then it is apparent that communication of problem data $\tilde{x}$ and $\tilde{u}$ between processors is required. In Section 5.6.2 and 5.6.3, we assume that this communication is reliable (i.e. problem data is shared perfectly). In Section 5.6.4, we consider game-theoretic approaches for distributed optimization, useful in settings where communication of decision variables is unreliable (e.g., packet loss).

5.6.2 Distributed minimization of weighted equation error

Weighted equation error, as in (5.11), can be decomposed as

$$\mathcal{E}_2^{\mathrm{w}} = \sum_{i=1}^{n_x}\sum_{j=1}^{n_x}\left\{|\tilde{A}(i,j)E(i,i) - F(i,j)|^2 + |\tilde{B}(i,j) - B(i,j)|^2\right\}, \tag{5.38}$$

i.e., the sum-of-squares of the elements of the matrices $\tilde{A}E - F$ and $\tilde{B} - B$. For $F = AE$, the stability condition $F\mathbf{1} - E\mathbf{1} \leq -\delta\mathbf{1}$ can be checked 'row-by-row'. Therefore, by taking 'row-based' decomposition of the system in which $\theta = \{E, F\}$, $\theta_i = \{E(i,i), F(i, n_i^{ar})\}$ and

$$\mathcal{E}_i^{\mathrm{w}}(\theta_i) := \sum_{j\in n_i^{ar}} |\tilde{A}(i,j)E(i,i) - F(i,j)|^2$$

it is apparent that $\min_{\theta\in\Theta_p} \|\tilde{A}E - F\|_F^2$ is equivalent to

$$\min_{\theta\geq 0} \sum_{i=1}^{n_x} \mathcal{E}_i^{\mathrm{w}}(\theta_i), \quad \text{s.t. } F\mathbf{1} - E\mathbf{1} \leq -\delta\mathbf{1}. \tag{5.39}$$

Problem (5.39) is completely separable, and may be solved as the independent collection of subproblems

$$\min_{\theta_i} \mathcal{E}_i^{\mathrm{w}}(\theta_i), \quad \text{s.t. } F(i, n_i^{ar})\mathbf{1} - E(i,i) \leq -\delta, \quad i = 1, \ldots, n_x, \tag{5.40}$$

each of which depends only on θ_i. Each problem in (5.40) is a linearly constrained quadratic program, for which efficient solvers are available.

5.6.3 Distributed minimization of Lagrangian relaxation of equation error

Lagrangian relaxation of equation error, i.e. $\bar{J}_\lambda$ as defined in (5.23), can also be decomposed into 'local' cost functions, in the style of (5.37). To see this, let us write (5.21) as

$$\bar{J}_\lambda(\theta, t) = \sup_{\Delta} \ \Delta'\Xi\Delta + \Delta'\xi_t, \tag{5.41}$$

where $\theta = (E, F, K)$, $\Xi = I - H'E - E'H$ and $\xi_t = H'\bar{\epsilon}_t$. When $E \in \mathbb{D}$, as in the case when using the polytopic set Θ_p, Ξ is also diagonal provided $H \in \mathbb{D}$. Then,

$$\begin{aligned}\bar{J}_\lambda(\theta, t) =& \sup_{\Delta} \sum\nolimits_{i=1}^{n_x} \Delta(i)'\Xi(i,i)\Delta(i) + \Delta(i)'\xi_t(i), \\ =& \sum\nolimits_{i=1}^{n_x} \sup_{\Delta} \{\Delta(i)'\Xi(i,i)\Delta(i) + \Delta(i)'\xi_t(i)\}, \\ :=& \sum\nolimits_{i=1}^{n_x} \bar{J}_i(\theta_i, t),\end{aligned} \tag{5.42}$$

for $\theta_i = \{E(i,i), F(i, n_i^{ar}), K(i, n_i^{br})\}$ By defining $\bar{J}_i(\theta_i) := \sum_{t=1}^{T} \bar{J}_i(\theta_i, t)$, it is apparent that

$$\bar{J}_\lambda(\theta) = \sum\nolimits_{t=1}^{T} \bar{J}_\lambda(\theta, t) = \sum\nolimits_{i=1}^{n_x} \bar{J}_i(\theta_i). \tag{5.43}$$

With this definition of $\bar{J}_i$, $\min_\theta \bar{J}_\lambda(\theta)$ is equivalent to

$$\min_{\theta \geq 0} \sum_{i=1}^{n_x} \bar{J}_i(\theta_i), \quad \text{s.t. } \mathbf{1}'F - \mathbf{1}'E \leq -\delta \mathbf{1}'. \tag{5.44}$$

The stability constraint, which can be enforced 'column-by-column', introduces coupling between θ_i and θ_j. This means that (5.44) cannot be decomposed into independent subproblems, as in (5.40). Fortunately, (5.44) is well suited to the alternating direction method of multipliers (ADMM); c.f. Section 2.2.3 for a brief introduction to the method. For convex functions f and g, ADMM solves problems of the form

$$\min_{\theta, z} \; f(\theta) + g(z), \quad \text{s.t. } A\theta + Bz = c, \tag{5.45}$$

by alternately minimizing the augmented Lagrangian,

$$L_\rho = f(\theta) + g(z) + \mu'(A\theta + Bz - c) + \frac{\rho}{2}\|A\theta + Bz - c\|_2^2, \tag{5.46}$$

w.r.t. θ and z, i.e., at the kth iteration, θ^k, z^k and μ^k are updated according to:

$$\begin{aligned}\theta^{k+1} &= \arg\min_{\theta} \; L_\rho(\theta, z^k, \mu^k), \\ z^{k+1} &= \arg\min_{z} \; L_\rho(\theta^{k+1}, z, \mu^k), \\ \mu^{k+1} &= \mu^k + \rho(A\theta^{k+1} + Bz^{k+1} - c).\end{aligned}$$

Here μ is a Lagrange multiplier, and $\rho > 0$ is a user-specified (scalar) *penalty parameter*. To solve (5.44) with ADMM, we introduce duplicates of the decision variables, i.e. $z = \{E^z, F^z\}$, with $z_i = \{E^z(i,i), F^z(i, n^{ar})\}$ for $i = 1, \ldots, n_x$. Then (5.44) is equivalent to

$$\min_{\theta \geq 0, z \geq 0} \sum_{i=1}^{n_x} \bar{J}_i(\theta_i) + \sum_{j=1}^{n_x} \mathcal{I}_- \left(\mathbf{1}'F^z(n_j^{ac}, j) - E^z(j,j) + \delta\right), \quad \text{s.t. } \theta = z, \tag{5.47}$$

where $\mathcal{I}_-$ is the indicator function for nonpositive reals,

$$\mathcal{I}_-(x) = \begin{cases} 0 & x \leq 0 \\ \infty & x > 0 \end{cases}. \tag{5.48}$$

Then, at the kth iteration, update θ, z and μ by solving:

$$\theta_i^{k+1} = \arg\min_{\theta_i \geq 0} \; \bar{J}_i(\theta_i) - \mu_i'\theta_i + \frac{\rho}{2}\|\theta_i - z_i^k\|_2^2, \tag{5.49a}$$

$$z_{c_i}^{k+1} = \arg\min_{z_{c_i} \geq 0} \; \mu_{c_i}' z_{c_i} + \frac{\rho}{2}\|\theta_{c_i}^{k+1} - z_{c_i}\|_2^2, \tag{5.49b}$$

$$\text{s.t.} \quad \mathbf{1}'F^z(n_i^{ac}, i) - E^z(i,i) \leq -\delta\mathbf{1}'$$

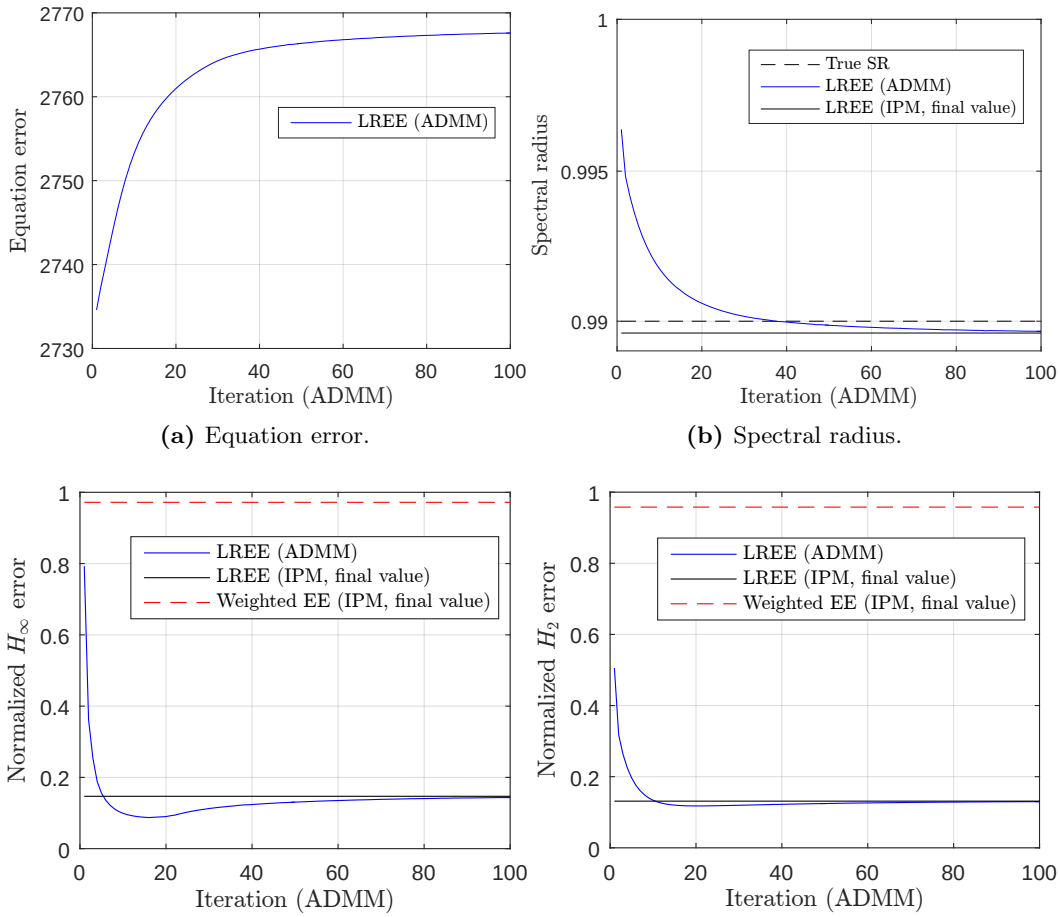
$$\mu_i^{k+1} = \mu_i^k - \rho\left(\theta_i^{k+1} - z_i^{k+1}\right), \tag{5.49c}$$

for $i = 1, \ldots, n_x$. Here $\theta_{c_i} = \{E(i,i), F(n_i^{ac}, i)\}$ denotes a partition of the model parameters by columns, rather than by rows, as in θ_i; z_{c_i} and μ_{c_i} are defined analogously. A couple of comments on this procedure are in order. The update policies (5.49a) and (5.49b) are a low-dimensional SDP and linearly constrained QP, respectively, to which standard solvers are applicable. From a distributed optimization perspective, suppose the ith processor is responsible for storing and updating θ_i, z_i and μ_i. The update (5.49a) can be performed with 'local decision variables' (i.e. z_i and μ_i); however, the update (5.49b) requires processor i to communicate with its column neighbors, so as to form the quantities θ_{c_i}, z_{c_i} and μ_{c_i}.

Numerical illustration

We conclude this section with an illustration of the solution of $\min_{\theta \in \Theta_p} \bar{J}_\lambda(\theta)$ via ADMM. The true system was a randomly generated 5th order positive system ($n_x = 5$, $n_u = 1$): A had spectral radius equal to 0.99; both A and B were dense. We set $C = \mathbf{1}'$. This system was simulated for $T = 100$ time steps, and the simulated states were corrupted by white noise (10dB) to generate training data. ADMM was initialized with $\theta = z = 0$ (i.e. all parameters set to zero), $\mu = 0$ and $\rho = 1$.

From Figure 5.13 we see that ADMM converges to a solution with the same quality-of-fit (measured by H_∞ and H_2 error) as that from an interior point method (IPM) within 100 iterations. We also observe that Lagrangian relaxation combined with model stability constraints has something of a 'regularizing effect' on the solution; i.e., the global minimizer of $\min_{\theta \in \Theta_p} \bar{J}_\lambda(\theta)$ is different to the (stable) minimizer of equation error. This can be seen from Figure 5.13(a) and (b). In fact, the model from LREE has better 'generalizability' (i.e. lower H_∞ and H_2 error) compared to the stable minimizer of equation error; c.f. Figure 5.13(c) and (d). The regularizing effect of LR and model stability constraints was also observed in Chapter 3, c.f., Section 3.5.2. Application of ADMM to the identification of very high-dimensional systems (e.g. hundreds of thousands of state variables) is a subject for further research.



(a) Equation error. **(b)** Spectral radius.

(c) Normalized H_∞ error. **(d)** Normalized H_2 error.

Figure 5.13 – Illustration of ADMM for the solution of $\min_{\theta \in \Theta_p} \bar{J}_\lambda(\theta)$. ADMM converges to a solution with the same quality-of-fit (measured by H_∞ and H_2 error) as that from an interior point method (IPM). Notice that only the final value of the solutions from IPMs are plotted. Lagrangian relaxation of equation error (LREE) demonstrates superior performance compared to weighted equation error.

5.6.4 Game-theoretic approaches

The distributed formulation of Section 5.6.3 gave no consideration to robustness against imperfect communication between processors. In this section, we address this by adopting a recently developed approach to distributed optimization, in which the solution is obtained from the Nash equilibrium of a state-based potential game [120, 145]. Such an approach is known to be robust to delays in communication and heterogeneous clock rates [145]. In what follows, we show how the game-theoretic approach of [121] can be used to solve $\min_{\theta \in \Theta_p} \bar{J}_\lambda(\theta)$ in a distributed fashion.

The method of [121] applies to optimization problems of the form

$$\min_{\theta_i \geq 0} \sum_{i \in N} \phi_i(\theta_i), \tag{5.50a}$$

$$\text{s.t.} \sum_{i \in N} Q_i^k \theta_i + q^k \leq 0, \; k = 1, \ldots, m. \tag{5.50b}$$

Here $N = \{1, \ldots, n\}$ denotes a set of n agents. Each agent i has its own set of local decision variables, θ_i, and local cost function $\phi_i(\theta_i)$, assumed to be convex and differentiable. The agents' decisions are constrained by m linear constraints $\{\sum_{i \in N} Q_i^k \theta_i + q^k \leq 0\}_{k=1}^m$. Furthermore, it is assumed that each agent i can only communicate with its neighboring agents $j \in N_i$.

The optimization problem $\min_{\theta \in \Theta_p} \bar{J}_\lambda(\theta)$ can be put into the form (5.50) by setting $\theta_i = \{E(i,i), F(i, n_i^{ar}), K(i, n_i^{br})\}$ (as in ADMM), $\phi_i(\theta_i) = \bar{J}_i(\theta_i)$, and choosing $\{Q_i^k, q_i^k\}$ so as to encode the stability constraint (5.6), i.e.,

$$\sum_{i \in N} Q_i^k \theta_i + q^k \leq 0 \iff \mathbf{1}' F(:, k) - E(k,k) \leq -\delta, \quad k = 1, \ldots, n_x.$$

As we have discussed, the major obstacle to be overcome in a distributed solution to problems of the form (5.50) are the constraints (5.50b) introduce coupling between decision variables θ_i and θ_j, $i \neq j$. The method proposed in [121] decouples these constraints by auxiliary variables $e_i = \{e_i^k\}_{k=1}^m$ for $i \in N$. Here e_i^k denotes the ith agent's estimate of the kth constraint, i.e. $e_i^k \approx \sum_{i \in N} Q_i^k \theta_i + q^k$. A state-based potential game equivalent to (5.50) can be defined by introducing a state $\zeta_i = (\theta_i, e_i)$ for each agent $i \in N$, as well as a state *action* $\hat{\zeta}_i = (\hat{\theta}_i, \hat{e}_i)$. Here, $\hat{e}_i = \{\hat{e}_i^k\}_{k=1}^m$ with $\hat{e}_i^k = \{\hat{e}_{i \to j}^k\}_{j \in N_i}$. The term $\hat{e}_{i \to j}^k$ denotes the change in the estimation of the kth constraint that agent i communicates to agent $j \in N_i$. Agent states then evolve according to the dynamics $(\tilde{\theta}, \tilde{e}) = f(\zeta, \hat{\zeta})$ given explicitly by

$$\tilde{\theta}_i = \theta_i + \hat{\theta}_i \tag{5.51a}$$

$$\tilde{e}_i^k = e_i^k + Q_i^k \hat{v}_i + \sum_{j \in N_i} \hat{e}_{j \to i}^k - \sum_{j \in N_i} \hat{e}_{i \to j}^k. \tag{5.51b}$$

The introduction of estimation variables e_i enables the constraints in (5.50b) to be decoupled. Specifically, each agent is assigned its own cost function, which depends only on the states (and actions) of neighboring agents (c.f. [121, §III-B] for details):

$$\bar{\phi}_i(\zeta, \hat{\zeta}) = \phi_i(\theta_i) + \beta \sum_{j \in N_i} \sum_{k=1m} \left(\max(0, \tilde{e}_j^k) \right)^2. \tag{5.52}$$

Here $\beta > 0$ denotes a trade-off parameter, which balances the ith agent's local cost function with the penalty on inconsistencies between estimation terms. To solve (5.50), each agent $i \in N$ minimizes its own individual cost function $\bar{\phi}_i$. Almost any optimization policy can be

employed, gradient play is a popular choice (c.f., e.g., [120, §4]). At each iteration, the ith agent's state is updated in accordance with (5.51), with expressions for $\hat{\theta}_i$ and $\hat{e}^k_{i\to j}$ given in [121, §4]. Gradient play is particularly well suited to $\min_{\theta\in\Theta_p} \bar{J}_\lambda(\theta)$, given the simplified expressions available for computing the gradient of the Lagrangian relaxation, c.f., 3.3.4.

Finally, it is clear that the gradient play policy outlined above requires agent i to communicate with neighboring agents $j \in N_i$. However, unlike the ADMM algorithm in §5.6.3, it is not clear how this neighbor set should be specified. In fact, any specification of the neighbor sets $\{N_i\}_{i\in N}$ that gives rise to an *connected*, *undirected* communication graph is sufficient; c.f. [121, Theorem 1]. The effect of different choices of $\{N_i\}_{i\in N}$ on convergence properties, as well as application of this method to the identification of large-scale systems, is the subject of ongoing research.

5.7 Conclusions

The main message of this chapter is that several problems in system identification are substantially simplified for positive systems, by leveraging stability and performance conditions based on element-wise linear inequalities. In particular, we have introduced a polytopic parametrization of all stable positive systems. Minimization of existing quality-of-fit criteria, such as weighted equation error [94, 113], subject to model stability conditions then takes the form of a quadratic program. In contrast, parametrizations of general stable LTI systems necessarily involve LMIs, and the analogous identification tasks result in semidefinite programs.

This chapter has also introduced a new convex quality-of-fit criterion, Lagrangian relaxation of equation error (LREE), as an alternative to weighted equation error. Although this construction involves a LMI, minimization of LREE has been shown, empirically, to produce significantly more accurate models compared to minimization of weighted equation error. Furthermore, unlike Lagrangian relaxation of simulation error, c.f. Chapter 3, LREE for high dimensional systems with sparse structure is readily decomposed into many smaller LMIs. When combined with the polytopic parametrization of stable models, this decomposition permits minimization of LREE by distributed methods such as ADMM and game-theoretic approaches.

5.8 Proofs

5.8.1 Proof of Theorem 5.1

First, let us establish that for $\theta = \{E, F\} \in \Theta_m$, $A := E^{-1}F \geq 0$. Notice that positive semidefiniteness of (5.5) implies that $E + E' \succeq P + \delta I \succ 0$. This implies that E is an

M-matrix, as $-E \in \mathbb{M}^{n_x \times n_x}$, c.f. Definition 5.3. Therefore, $E^{-1} \geq 0$, which ensures that $E^{-1}F \geq 0$ for $F \geq 0$.

To prove that Θ_m contains *only* stable models, note that Theorem 5.1 is in fact a special case of Lemma 4.2 (c.f., also, Lemma 4 and Corollary 5 in [141, Section 3.2]) with $-E \in \mathbb{M}^{n_x \times n_x}$ and $F \geq 0$. Therefore $\theta = \{E, F\} \in \Theta_m$ implies $E^{-1}F \in \mathcal{S}_+$.

To prove that Θ_m contains *every* stable positive system, note that Θ_m is a generalization of Θ_m, which is a parametrization of all stable positive systems, c.f. (5.2) and [94]. Specifically, Θ_m is equivalent to Θ_l when $E = P \in \mathbb{D}$. Therefore, every $A \in \mathcal{S}_+$ has a representation in Θ_l, and also in Θ_m. □

5.8.2 Proof of Theorem 5.2

We first establish that for $\theta = \{E, F\} \in \Theta_p$, $A := E^{-1}F \geq 0$. This is trivial, because $E \in \mathbb{D}$ and (5.6) implies $E \geq \delta I$.

Let us now prove sufficiency, i.e. $\theta \in \Theta_p \implies A := E^{-1}F \in \mathcal{S}_+$. For clarity, let $E = \operatorname{diag}(e)$. As $F = FA$, (5.6) implies

$$\mathbf{1}'F - \mathbf{1}'E = \mathbf{1}'EA - \mathbf{1}'E = e'A - e' \leq -\delta\mathbf{1}' < 0 \tag{5.53}$$

which, from Lemma 5.1, implies $A \in \mathcal{S}_+$.

Let us now prove necessity, i.e., $A \in \mathcal{S}_+ \implies \exists\{E, F\} \in \Theta_p$. If $A \in \mathcal{S}_+$ then Lemma 5.1 implies existence of $e \in \mathbb{R}^{n_x}_{++}$ such that $e'A - e' < 0$. For any $\delta > 0$, we can scale e such that $e'A - e' \leq -\delta\mathbf{1}'$. With this scaled e, let $E = \operatorname{diag}(e)$ and $F = EA$. Then (5.53) implies $\{E, F\} \in \Theta_p$. □

5.8.3 Proof of Theorem 5.3

By construction $\bar{J}_\lambda(\theta)$ is a convex upper bound on $\sum_{t=1}^{T} \|\epsilon_t\|_2^2$; c.f., (5.23). In this section we derive the LMI in (5.24). First, notice that

$$\bar{\epsilon}_t = \underbrace{[E, -F, -K]}_{E_K^F} \underbrace{\left[\tilde{x}'_{t+1}, \tilde{x}'_t, \tilde{u}'_t\right]'}_{\phi_t}. \tag{5.54}$$

Then, recalling the explicit definition of $\bar{J}_\lambda(\theta, t)$ from (5.22),

$$\begin{aligned}
\bar{J}_\lambda(\theta) := \sum_{t=1}^{T} \bar{J}_\lambda(\theta, t) &= \sum_{t=1}^{T} |H'\bar{\epsilon}_t|^2_{(H'E+E'H-I)^{-1}} \\
&= \sum_{t=1}^{T} \operatorname{tr}\left(\phi_t\phi_t' E_K^{F'} H(H'E + E'H - I)^{-1} H' E_K^F\right) \\
&= \operatorname{tr}\left(\sum_{t=1}^{T} \phi_t\phi_t' E_K^{F'} H(H'E + E'H - I)^{-1} H' E_K^F\right) \\
&= \operatorname{tr}\left(\Phi E_K^{F'} H(H'E + E'H - I)^{-1} H' E_K^F\right) \\
&= \operatorname{tr}(\Phi R),
\end{aligned}$$

where Φ is defined in (5.25) and R is a slack variable such that

$$R \succeq E_K^{F'} H(H'E + E'H - I)^{-1} H' E_K^F. \tag{5.55}$$

By the Schur complement, (5.55) is equivalent to the LMI in (5.24). □

5.8.4 Proof of Lemma 5.4

By an application of Lemma 5.3, with $D = 0$ and $B = I_{n_x}(:, j)$, the ℓ_1-gain of $\epsilon(j) \mapsto z$ is equal to $q(j)$, given by

$$\min_{q \in \mathbb{R}^{n_x}_{++}} \; q(j) \quad \text{s.t.} \quad C'\mathbf{1} + (A' - I)q < 0. \tag{5.56}$$

We must show that $p(j) = q(j)$. Suppose $p(j) < q(j)$. As p is in the feasible set of (5.56), this implies that q is not the optimal solution of (5.56), and thus $p(j) \geq q(j)$. Suppose, that $p(j) > q(j)$. The jth row of the constraint in (5.31) is $p(j)(1 - A(j, j)) - \sum_{i \neq j}^{n_x} p(i)A(j, i) > (C'\mathbf{1})(j)$ which also holds for q. As A is Schur stable, $A(j, j) < 1$, $q^{(j)} < p^(j)$, implies $q(i) < p(i)$ for $i \neq j$. This implies that p is not the optimal solution of (5.31), and thus $p(j) = q(j)$.

5.8.5 Proof of Theorem 5.4

To prove the claim that $\mathcal{E}_1^{\text{w}}$ is an upper bound for the 1-norm of output error, $\mathcal{E}_1^{\text{s}}$, we need only consider the E that satisfies (5.32) with minimal $\sum_i E(i, i)$, as $\mathcal{E}_1^{\text{w}}$ will only be larger for any other choice of E. For convenience, define e such that $E = \operatorname{diag}(e)$. From (5.27), to prove $\mathcal{E}_1^{\text{w}} \geq \mathcal{E}_1^{\text{s}}$ it is sufficient to show that $\sum_t \|E\epsilon_t\|_1 \geq \sum_t \|z_t\|_1$, where ϵ and z are the input and output of (5.29). Let $z_{t,j}$ denote the output of (5.29) at time t, in response to the jth input channel $\epsilon(j)$ alone, i.e. when $\epsilon(i) = 0$ for $i \neq j$. As (5.29) is linear, by

superposition we have $z_t = \sum_{j=1}^{n_x} z_{t,j}$. Therefore,

$$\sum_{t=1}^{T} \|z_t\|_1 = \sum_{t=1}^{T} \|\sum_{j=1}^{n_x} z_{t,j}\|_1 \leq \sum_{t=1}^{T}\sum_{j=1}^{n_x} |z_{t,j}| \leq \sum_{t=1}^{T}\sum_{j=1}^{n_x} e(j)|\epsilon_t(j)| = \sum_{t=1}^{T} \|E\epsilon_t\|_1,$$

holds for all T. Here the second inequality holds because $e(j)$ is an upper bound for the ℓ_1-gain from $\epsilon \mapsto z$, by Lemma 5.4. $\square$

Chapter 6

Conclusion

The central theme of this thesis has been the scalable application of convex optimization to problems arising in data-driven modeling of dynamical systems, specifically, model instability and nonconvexity of quality-of-fit criteria, such as simulation error. To address these challenges, we have developed new convex parametrizations of stable models and convex quality-of-fit criteria, as well as efficient algorithms to optimize the latter over the former.

Specifically, Chapter 3 presented specialized interior point algorithms for Lagrangian relaxation of simulation error. Hitherto, Lagrangian relaxation had been used to generate convex approximations to simulation error (a.k.a. output error) and guarantee model stability, however, the large-dimension of the resulting semdefinite programs (SDPs) limited practical utility. The custom algorithms developed in this thesis reduce computational complexity to linear growth in the length of the training dataset, down from the cubic growth exhibited by generic SDP solvers. This new algorithm enabled empirical comparisons to established methods, such as Nonlinear ARX, in which superior generalization to new data was demonstrated.

Chapter 4 introduced model stability constraints into the maximum likelihood identification of dynamical systems, thereby extended some of the ideas from Chapter 3 to a stochastic setting. Lagrangian relaxation was combined with the expectation maximization (EM) algorithm to generate tight lower bounds to the likelihood that can be optimized over a convex parametrization of all stable linear systems, via semidefinite programing. Two formulations of EM were proposed: one uses states as latent variables, the other uses disturbances. It was shown, via both theoretical and empirical analysis, that bounds based on latent states perform better when the effect of disturbances in more significant than measurement noise; the converse is true for the latent disturbances formulation when the situation is reversed.

Finally, Chapter 5 considered the special case of identification of internally positive systems. Such systems have received increased attention from the control community in the past decade, especially in the context of distributed analysis and design for large scale networks.

The main message of Chapter 5 is that many of the convex constructions derived for generic systems are greatly simplified when the dynamics are positive. Convex parametrizations of stable models defined by polytopic sets, and quality-of-fit criteria that can be minimized by linear and quadratic programing were proposed; analogous constructions for generic systems require linear matrix inequalities (LMIs) and lead to SDPs. These simplified stability and performance conditions permit identification of large-scale positive systems via distributed optimization.

In this concluding chapter, some open problems and directions for future research are discussed.

6.1 Open problems and directions for future work

EM for identification of stable nonlinear systems

It would be useful to extend the methods presented in Chapter 4 to identification of nonlinear dynamical systems; in fact, our early work [237] on EM with latent disturbances outlined a framework for such problems. The major challenges associated with the nonlinear setting are as follows. First, it was shown in Chapter 4 that Lagrangian relaxation (LR) can be used to develop tight bounds on the likelihood that are compatible with a convex parametrization of stable models. For linear systems, LR leads to semidefinite programs. For nonlinear systems, LR leads to convex constructions that are not necessarily computationally tractable; e.g., LR of simulation error for nonlinear systems requires computing the supremum of a nonlinear, nonconvex function. When the system nonlinearities are given by polynomials, sum-of-squares programing can be used to approximate the supremum; however, the resulting SDPs are very large, c.f. Section 2.4.2. One solution is to consider LR of *linearized* quantities, e.g., linearized simulation error as in Chapter 3. Working with linearized quantities improves tractability; however, it is not yet clear whether the resulting relaxations can be designed to give tight bounds on the likelihood.

Second, the latent disturbances formulation of EM requires samples from the joint smoothing distribution of *disturbances*, $p_\theta(x_1, w_{1:T}|y_{1:T})$. In the linear-Gaussian setting, (the mean and variance of) this distribution can be computed in closed form; c.f. Section 4.3.2. In the nonlinear non-Gaussian setting, disturbance smoothing is still an active area of research; c.f. [157] for recent work on the *disturbance filtering* problem. When the model structure is such that knowledge of states implies knowledge of disturbances, e.g. $x_{t+1} = a(x_t, u_t) + w_t$, then samples $\bar{w}_t \sim p_\theta(w_t|y_{1:T})$ can be computed as $\bar{w}_t = \bar{x}_t - a(\bar{x}_t, u_t)$, where $\bar{x}_t \sim p_\theta(x_t|y_{1:T})$ are generated by, e.g., sequential Monte Carlo methods [204].

Subspace algorithms for positive systems

As in Chapter 3, the convex quality-of-fit criteria presented in Chapter 5 also depend on approximate state sequences for their construction. For generic LTI systems, these state estimates can be generated by subspace algorithms, c.f., Section 2.1.3. However, for the internally positive systems studied in Chapter 5, subspace algorithms do not, in general, return state estimates in a basis consistent with a positive realization of the dynamics. The fundamental problem is that internal positivity is not a property that is preserved under arbitrary similarity transformations of the dynamics.

One way to modify the standard subspace algorithms for application to positive systems could involve replacing the singular value decomposition (SVD) in (2.34) with a *nonnegative matrix factorization* (NMF), i.e. $\mathcal{O}_i = NM$, where $N = \Gamma_i \geq 0$ (the nonnegative extended observability matrix) and $M = X_f \geq 0$ (the nonnegative estimated state sequence). Computing such a factorization is nonconvex, so in one sense we have simply traded one difficult problem for another. Nevertheless, over the past two decades or so NMF has received considerable attention in the machine learning community, especially for unsupervised learning, as it has been shown to provide useful decompositions of multivariate data, c.f., e.g., [117]. It would be interesting to apply NMF algorithms from the machine learning community [18, 118] to subspace identification of positive systems.

Connections to machine learning

As discussed in Chapter 2, the task of training a neural network in machine learning is an example of data-driven modeling of a dynamical system. In particular, the equivalence between recurrent neural networks (RNNs), popular in applications that process sequential data such as speech and text, and nonlinear state space models has been well established [173]. Training of RNNs has, at least conceptually, much in common with the simulation error minimization problem. Despite this, there has been relatively little interaction between the system identification and machine learning community. Neural networks have certainly been utilized in system identification, but mostly as a functional form for the transition dynamics, e.g., in nonlinear ARX [2, 40, 41]. However, until recently, lessons learned from training neural networks have not been translated to a system identification setting.

Machine learning has enjoyed considerable success in recent years, especially with advances in deep learning [116]. One of the most remarkable aspects of this success, is the effectiveness of first order methods (such as stochastic gradient descent) for minimization of quality-of-fit criteria (i.e., training error) for neural networks. The efficacy of first order methods for system identification has been explored in the recent work [85], which proved that gradient descent, under very mild assumptions on the true model generating the training data, converges to a global minimizer of output error for LTI state space models. It has long been known that, asymptotically, every stationary point of the output error criterion is also a global minimum [213], however, [85] appears to be the first work to provide polynomial

bounds on the convergence. The result is clearly an important stepping stone towards better understanding the training of nonlinear dynamical systems constituting RNNs, and it will be interesting to see what impact these findings have on the system identification community.

On the other hand, it will be interesting to see whether developments from system identification can be translated to machine learning applications. For instance, a crucial ingredient in the success of RNNs for speech and text processing has been the development of model structures capable of learning long-range dependencies in the training data. Such model structures include long-short term memory (LSTM) networks [91], Neural Turning Machines [78], and memory networks [254]. It may be possible that representations of dynamical systems developed in the identification community (e.g. parametrizations of stable models) end up being useful for refining the structure of neural networks in certain applications.

Bibliography

[1] D. Acemoglu, V. M. Carvalho, A. Ozdaglar, and A. Tahbaz-Salehi. The network origins of aggregate fluctuations. *Econometrica*, 80(5):1977–2016, 2012.

[2] H. Adeli and X. Jiang. Dynamic fuzzy wavelet neural network model for structural system identification. *Journal of Structural Engineering*, 132(1):102–111, 2006.

[3] H. Akaike. On the use of a linear model for the identification of feedback systems. *Annals of the Institute of Statistical Mathematics*, 20(1):425–439, 1968.

[4] H. Akaike. Maximum likelihood identification of gaussian autoregressive moving average models. *Biometrika*, pages 255–265, 1973.

[5] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.

[6] M. S. Andersen, J. Dahl, and L. Vandenberghe. Implementation of nonsymmetric interior-point methods for linear optimization over sparse matrix cones. *Mathematical Programming Computation*, 2(3):167–201, 2010.

[7] B. D. Anderson, M. Deistler, L. Farina, and L. Benvenuti. Nonnegative realization of a linear system with nonnegative impulse response. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 43(2):134–142, 1996.

[8] C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

[9] D. Angeli. A Lyapunov approach to incremental stability properties. *IEEE Transactions on Automatic Control*, 47(3):410–421, 2002.

[10] K. J. Aström. *Lectures on the identification problem the least squares method.* Lund Institute of Technology, Division of Automatic Control, 1968.

[11] D. Bauer and L. Ljung. Some facts about the choice of the weighting matrices in Larimore type of subspace algorithms. *Automatica*, 38(5):763–773, 2002.

[12] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[13] L. Benvenuti and L. Farina. An example of how positivity may force realizations of large dimension. *Systems & Control Letters*, 36(4):261–266, 1999.

[14] L. Benvenuti and L. Farina. A tutorial on the positive realization problem. *IEEE Transactions on Automatic Control*, 49(5):651–664, 2004.

[15] L. Benvenuti, A. De Santis, and L. Farina. On model consistency in compartmental systems identification. *Automatica*, 38(11):1969–1976, 2002.

[16] A. Berman and R. Plemmons. *Nonnegative Matrices in the Mathematical Sciences.* Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

[17] A. Berman and R. J. Plemmons. *Nonnegative matrices in the mathematical sciences.* SIAM, 1979.

[18] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis*, 52(1):155–173, 2007.

[19] D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods.* Academic Press, 1982.

[20] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and distributed computation: numerical methods*, volume 23. Prentice Hall Englewood Cliffs, NJ, 1989.

[21] B. Besselink, N. van de Wouw, and H. Nijmeijer. Model reduction for nonlinear systems with incremental gain or passivity properties. *Automatica*, 49(4):861–872, 2013.

[22] J. T. Betts. *Practical methods for optimal control and estimation using nonlinear programming.* SIAM, 2010.

[23] S. Billings and S. Fakhouri. Identification of systems containing linear dynamic and static nonlinear elements. *Automatica*, 18(1):15–26, 1982.

[24] S. A. Billings. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains.* John Wiley & Sons, 2013.

[25] B. Bond, Z. Mahmood, Y. Li, R. Sredojevic, A. Megretski, V. Stojanovi, Y. Avniel, and L. Daniel. Compact modeling of nonlinear analog circuits using system identification via semidefinite programming and incremental stability certification. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(8):1149–1162, 2010.

[26] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT*, pages 177–186. Springer, 2010.

[27] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time series analysis: forecasting and control.* John Wiley & Sons, 2015.

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

[29] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory.* SIAM, 1994.

[30] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[31] C. Briat. Robust stability and stabilization of uncertain linear positive systems via integral linear constraints: L_1- and L_∞-gains characterization. *International Journal of Robust and Nonlinear Control*, 23(17):1932–1954, 2013.

[32] S. Burer. Semidefinite programming in the space of partial positive semidefinite matrices. *SIAM Journal on Optimization*, 14(1):139–172, 2003.

[33] V. Cerone, D. Piga, and D. Regruto. Set-membership error-in-variables identification through convex relaxation techniques. *IEEE Transactions on Automatic Control*, 57(2):517–522, 2012.

[34] V. Cerone, D. Piga, and D. Regruto. Computational load reduction in bounded error identification of hammerstein systems. *IEEE Transactions on Automatic Control*, 58(5):1317–1322, 2013.

[35] F. Ch. Hilbert's 17th problem and best dual bounds in quadratic minimization. *Cybernetics and Systems Analysis*, 34(5), 1998.

[36] A. Charnes and W. W. Cooper. Management models and industrial applications of linear programming. *Management Science*, 4(1):38–91, 1957.

[37] J. Chen and G. Gu. *Control-oriented system identification: an H-infinity approach*, volume 19. Wiley-Interscience, 2000.

[38] J. Chen, C. N. Nett, and M. K. Fan. Worst-case system identification in h∞: validation of apriori information, essentially optimal algorithms, and error bounds. In *Proc. of American Control Conference (ACC)*, pages 251–257. IEEE, 1992.

[39] S. Chen and S. Billings. Representations of non-linear systems: the NARMAX model. *International Journal of Control*, 49(3):1013–1032, 1989.

[40] S. Chen and S. Billings. Neural networks for nonlinear dynamic system modelling and identification. *International Journal of Control*, 56(2):319–346, 1992.

[41] S. Chen, S. Billings, and P. Grant. Non-linear system identification using neural networks. *International Journal of Control*, 51(6):1191–1214, 1990.

[42] T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and gaussian processes - revisited. *Automatica*, 48(8):1525–1535, 2012.

[43] A. Chiuso and G. Picci. Consistency analysis of some closed-loop subspace identification methods. *Automatica*, 41(3):377–391, 2005.

[44] M.-D. Choi, T. Y. Lam, and B. Reznick. Sums of squares of real polynomials. In *Proceedings of Symposia in Pure mathematics*, volume 58, pages 103–126. American Mathematical Society, 1995.

[45] C. T. Chou and M. Verhaegen. Subspace algorithms for the identification of multivariable dynamic errors-in-variables models. *Automatica*, 33(10):1857–1869, 1997.

[46] E. Dall'Anese, H. Zhu, and G. B. Giannakis. Distributed optimal power flow for smart microgrids. *IEEE Transactions on Smart Grid*, 4(3):1464–1475, 2013.

[47] G. Dantzig. *Linear programming and extensions.* Princeton University Press, 2016.

[48] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960.

[49] P. De Leenheer and D. Aeyels. Stabilization of positive linear systems. *Systems & Control Letters*, 44(4):259–271, 2001.

[50] A. De Santis and L. Farina. Identification of positive linear systems with poisson output transformation. *Automatica*, 38(5):861–868, 2002.

[51] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (methodological)*, pages 1–38, 1977.

[52] F. J. Doyle III, R. K. Pearson, and B. A. Ogunnaike. *Identification and control using Volterra models.* Springer Science & Business Media, 2012.

[53] J. Durbin and S. J. Koopman. *Time series analysis by state space methods.* Number 38. Oxford University Press, 2012.

[54] J. Eckstein. *Splitting methods for monotone operators with applications to parallel optimization.* PhD thesis, Massachusetts Institute of Technology, 1989.

[55] J. Eckstein and M. Fukushima. Some reformulations and applications of the alternating direction method of multipliers. In *Large Scale Optimization*, pages 115–134. Springer, 1994.

[56] B. Efron and T. Hastie. *Computer Age Statistical Inference*, volume 5. Cambridge University Press, 2016.

[57] H. Everett III. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11(3):399–417, 1963.

[58] L. Farina and S. Rinaldi. *Positive Linear Systems: Theory and Applications.* John Wiley & Sons, 2011.

[59] W. Favoreel, B. De Moor, and P. Van Overschee. Subspace state space system identification for industrial processes. *Journal of process control*, 10(2):149–155, 2000.

[60] A. V. Fiacco and G. P. McCormick. *Nonlinear programming: sequential unconstrained minimization techniques*. SIAM, 1990.

[61] M. L. Fisher. The Lagrangian relaxation method for solving integer programming problems. *Management Science*, 27(1):1–18, 1981.

[62] R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 222:309–368, 1922.

[63] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *The Computer Journal*, 7(2):149–154, 1964.

[64] R. Fourer and S. Mehrotra. Performance of an augmented system approach for solving least-squares problems in an interior-point method for linear programming. *COAL Newsletter*, 19:26–31, 1991.

[65] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen. Bayesian inference and learning in gaussian process state-space models with particle mcmc. In *Advances in Neural Information Processing Systems*, pages 3156–3164, 2013.

[66] R. Frigola-Alcalde. *Bayesian time series learning with Gaussian processes*. PhD thesis, University of Cambridge, 2015.

[67] M. Fukuda, M. Kojima, K. Murota, and K. Nakata. Exploiting sparsity in semidefinite programming via matrix completion I: General framework. *SIAM Journal on Optimization*, 11(3):647–674, 2001.

[68] M. Fukushima. Application of the alternating direction method of multipliers to separable convex programming problems. *Computational Optimization and Applications*, 1(1):93–111, 1992.

[69] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976.

[70] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, November 1984.

[71] S. Gibson and B. Ninness. Robust maximum-likelihood estimation of multivariable dynamic systems. *Automatica*, 41(10):1667–1682, 2005.

[72] P. E. Gill, W. Murray, M. A. Saunders, J. A. Tomlin, and M. H. Wright. On projected Newton barrier methods for linear programming and an equivalence to Karmarkars projective method. *Mathematical Programming*, 36(2):183–209, 1986.

[73] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 9(2):41–76, 1975.

[74] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014.

[75] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[76] G. C. Goodwin. *Encyclopedia of Systems and Control*, chapter Experiment design for system identification. Pergamon Press, Oxford, 1987.

[77] G. C. Goodwin and R. L. Payne. *Dynamic system identification: experiment design and data analysis*. Academic press, 1977.

[78] A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[79] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. Positive definite completions of partial hermitian matrices. *Linear Algebra and its Applications*, 58:109–124, 1984.

[80] C. Grussler, J. Umenberger, and I. R. Manchester. Identification of externally positive systems. In *Proceedings of the 56th Conference on Decision and Control (CDC)*. IEEE, 2017.

[81] T. Gustafsson. Subspace-based system identification: weighting and pre-filtering of instruments. *Automatica*, 38(3):433–443, 2002.

[82] W. M. Haddad and V. Chellaboina. Stability and dissipativity theory for nonnegative dynamical systems: a unified analysis framework for biological and physiological systems. *Nonlinear Analysis: Real World Applications*, 6(1):35–65, 2005.

[83] W. M. Haddad, V. Chellaboina, and Q. Hui. *Nonnegative and compartmental dynamical systems*. Princeton University Press, 2010.

[84] A. Hansson, R. Wallin, and L. Vandenberghe. Comparison of two structure-exploiting optimization algorithms for integral quadratic constraints. In *Proc. of IFAC Symp. on Robust Control Design*, Milan, Italy, 2003.

[85] M. Hardt, T. Ma, and B. Recht. Gradient descent learns linear dynamical systems. *arXiv preprint arXiv:1609.05191*, 2016.

[86] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer Series in Statistics, 2 edition, 2009.

[87] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[88] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2): 342–361, 1996.

[89] A. J. Helmicki, C. A. Jacobson, and C. N. Nett. Control oriented system identification: a worst-case/deterministic approach in h/sub infinity. *IEEE Transactions on Automatic Control*, 36(10):1163–1176, 1991.

[90] I. Herman, D. Martinec, Z. Hurák, and M. Šebek. Nonzero bound on fiedler eigenvalue causes exponential growth of h-infinity norm of vehicular platoon. *IEEE Transactions on Automatic Control*, 60(8):2248–2253, 2015.

[91] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997.

[92] T. Hsia. On least squares algorithms for system parameter identification. *IEEE Transactions on Automatic Control*, 21(1):104–108, 1976.

[93] Q. Huang, Y. Yuan, J. Goncalves, and M. A. Dahleh. H2 norm based network volatility measures. In *Proc. American Control Conference (ACC)*, pages 3310–3315. IEEE, 2014.

[94] Q. Hui and W. M. Haddad. Subspace identification of stable nonnegative and compartmental dynamical systems via constrained optimization. In *American Control Conference, 2006*, pages 6–pp. IEEE, 2006.

[95] D. R. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–37, 2004.

[96] E. Jan and N. A. Kotov. Successful differentiation of mouse neural stem cells on layer-by-layer assembled single-walled carbon nanotube composite. *Nano Letters*, 7 (5):1123–1128, 2007.

[97] M. Jansson and B. Wahlberg. A linear regression approach to state-space subspace system identification. *Signal Processing*, 52(2):103–129, 1996.

[98] M. Jansson and B. Wahlberg. On consistency of subspace methods for system identification. *Automatica*, 34(12):1507–1519, 1998.

[99] T. A. Johansen. On tikhonov regularization, bias and variance in nonlinear system identification. *Automatica*, 33(3):441–446, 1997.

[100] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear estimation*, volume 1. Prentice Hall Upper Saddle River, NJ, 2000.

[101] N. Kantas, A. Doucet, S. S. Singh, and J. M. Maciejowski. An overview of sequential monte carlo methods for parameter estimation in general state-space models. *IFAC Proceedings*, 42(10):774–785, 2009.

[102] C.-Y. Kao and A. Megretski. Fast algorithms for solving iqc feasibility and optimization problems. In *Proc. of the American Control Conference (ACC)*, volume 4, pages 3019–3024. IEEE, 2001.

[103] C.-Y. Kao and A. Megretski. On the new barrier function and specialized algorithms for a class of semidefinite programs. *SIAM Journal on Control and Optimization*, 46(2):468–495, 2007.

[104] C.-Y. Kao, A. Megretski, and U. Jönsson. Specialized Fast Algorithms for IQC Feasibility and Optimization Problems. *Automatica*, 40(2):239–252, 2004.

[105] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the 16th annual ACM symposium on Theory of Computing*, pages 302–311. ACM, 1984.

[106] H. K. Khalil. *Noninear Systems*. Prentice-Hall, New Jersey, 1996.

[107] S. Kim and M. Kojima. Exploiting sparsity in SDP relaxation of polynomial optimization problems. In *Handbook on Semidefinite, Conic and Polynomial Optimization*, pages 499–531. Springer, 2012.

[108] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, 2007.

[109] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith. Dynamic systems identification with gaussian processes. *Mathematical and Computer Modelling of Dynamical Systems*, 11(4):411–424, 2005.

[110] M. Kojima, S. Shindoh, and S. Hara. Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM Journal on Optimization*, 7(1):86–125, 1997.

[111] K. Kristinsson and G. A. Dumont. System identification and control using genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(5):1033–1046, 1992.

[112] A. Krogh and J. A. Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[113] S. L. Lacy and D. S. Bernstein. Subspace identification with guaranteed stability using constrained optimization. *IEEE Transactions on Automatic Control*, 48(7): 1259–1263, 2003.

[114] W. E. Larimore. System identification, reduced-order filtering and modeling via canonical variate analysis. In *Proceedings of the American Control Conference (ACC)*, pages 445–451, San Francisco, USA, 1983.

[115] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[116] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[117] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[118] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[119] C. Lemaréchal. Lagrangian relaxation. In *Computational Combinatorial Optimization*, pages 112–156. Springer, 2001.

[120] N. Li and J. R. Marden. Designing games for distributed optimization. *Selected Topics in Signal Processing, IEEE Journal of*, 7(2):230–242, 2013.

[121] N. Li and J. R. Marden. Decoupling coupled constraints through utility design. *Automatic Control, IEEE Transactions on*, 59(8):2289–2294, 2014.

[122] F. Lindsten and T. B. Schön. Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends in Machine Learning*, 6(1):1–143, 2013.

[123] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

[124] G. P. Liu. *Nonlinear identification and control: a neural network approach*. Springer Science & Business Media, 2012.

[125] Z. Liu and L. Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1235–1256, 2009.

[126] L. Ljung. On consistency for prediction error identification methods. *Report TFRT*, 3072, 1974.

[127] L. Ljung. On the consistency of prediction error identification methods. *Mathematics in Science and Engineering*, 126:121–164, 1976.

[128] L. Ljung. Convergence analysis of parametric identification methods. *IEEE Transactions on Automatic Control*, 23(5):770–783, 1978.

[129] L. Ljung. *System Identification: Theory for the User*. Prentice Hall, 2 edition, January 1999.

[130] L. Ljung. Prediction error estimation methods. *Circuits, Systems and Signal Processing*, 21(1):11–21, 2002.

[131] L. Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1): 1–12, April 2010.

[132] L. Ljung and T. Söderström. *Theory and practice of recursive identification*, volume 5. JSTOR, 1983.

[133] J. Lofberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pages 284–289. IEEE, 2005.

[134] J. Lofberg. Pre- and post-processing sum-of-squares programs in practice. *IEEE Transactions on Automatic Control*, 54(5):1007–1011, May 2009.

[135] W. Lohmiller and J.-J. E. Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998.

[136] D. G. Luenberger. *Introduction to linear and nonlinear programming*, volume 28. Addison-Wesley Reading, MA, 1973.

[137] I. J. Lustig, R. E. Marsten, and D. F. Shanno. Interior point methods for linear programming: Computational state of the art. *ORSA Journal on Computing*, 6(1): 1–14, 1994.

[138] J. M. Maciejowski. Guaranteed stability with subspace methods. *Systems & Control Letters*, 26(2):153–156, 1995.

[139] R. Madani, A. Kalbat, and J. Lavaei. ADMM for sparse semidefinite programming with applications to optimal power flow problem. In *Proc. Conference on Decision and Control (CDC)*, pages 5932–5939. IEEE, 2015.

[140] J. Mallet-Paret and H. L. Smith. The poincaré-bendixson theorem for monotone cyclic feedback systems. *Journal of Dynamics and Differential Equations*, 2(4): 367–421, 1990.

[141] I. Manchester, M. M. Tobenkin, and A. Megretski. Stable nonlinear system identification: Convexity, model class, and consistency. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, Brussels, Belgium, 2012.

[142] I. R. Manchester. Input design for system identification via convex relaxation. In *Proc. Decision and Control (CDC)*, pages 2041–2046. IEEE, 2010.

[143] I. R. Manchester and J.-J. E. Slotine. Transverse contraction criteria for existence, stability, and robustness of a limit cycle. *Systems & Control Letters*, 63:32–38, 2014.

[144] I. R. Manchester, M. M. Tobenkin, and J. Wang. Identification of nonlinear systems with stable oscillations. In *Proc. Decision and Control and European Control Conference (CDC-ECC)*, pages 5792–5797. IEEE, 2011.

[145] J. R. Marden. State based potential games. *Automatica*, 48(12):3075–3088, 2012.

[146] D. Materassi and G. Innocenti. Topological identification in networks of dynamical systems. *IEEE Transactions on Automatic Control*, 55(8):1860–1871, 2010.

[147] MathWorks. Identifying Nonlinear ARX and Hammerstein-Wiener Models Using Measured Data. Available at: https://mathworks.com/help/ident/examples. 2016.

[148] N. Matni. Optimal zero-queue congestion control using admm. In *Proc. IEEE Amer. Control Conf.(ACC), Seattle, WA, USA*, 2017.

[149] A. Megretski. H-infinity model reduction with guaranteed suboptimality bound. In *Proc. American Control Conference (ACC)*, pages 6–pp. IEEE, 2006.

[150] A. Megretski. Convex optimization in robust identification of nonlinear feedback. In *Proceedings of the 47th IEEE Conference on Decision and Control (CDC)*, pages 1370–1374, Cancun, Mexico, 2008.

[151] R. Mestrom, R. Fey, J. Van Beek, K. Phan, and H. Nijmeijer. Modelling the dynamics of a MEMS resonator: simulations and experiments. *Sensors and Actuators A: Physical*, 142(1):306–315, 2008.

[152] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[153] M. Milanese and R. Tempo. Optimal algorithms theory for robust estimation and prediction. *IEEE Transactions on Automatic Control*, 30(8):730–738, 1985.

[154] D. N. Miller and R. A. De Callafon. Subspace identification with eigenvalue constraints. *Automatica*, 49(8):2468–2473, 2013.

[155] M. Moonen and J. Ramos. A subspace algorithm for balanced state space system identification. *IEEE Transactions on Automatic Control*, 38(11):1727–1729, 1993.

[156] J. J. Moré. The Levenberg-Marquardt algorithm: implementation and theory. In *Numerical Analysis*, pages 105–116. Springer, 1978.

[157] L. M. Murray, E. M. Jones, and J. Parslow. On disturbance state-space models and the particle marginal metropolis-hastings sampler. *SIAM/ASA Journal on Uncertainty Quantification*, 1(1):494–521, 2013.

[158] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota. Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results. *Mathematical Programming*, 95(2):303–327, 2003.

[159] O. Nelles. *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer Science & Business Media, 2013.

[160] G. L. Nemhauser and L. A. Wolsey. Integer programming and combinatorial optimization. *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, 20:8–12, 1988.

[161] A. Nemirovskii and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. John Wiley & Sons, 1983.

[162] Y. Nesterov, A. Nemirovski, and Y. Ye. *Interior-point polynomial algorithms in convex programming*, volume 13 of *Studies in Applied and Numerical Mathematics*. SIAM, 1994.

[163] Y. E. Nesterov and M. J. Todd. Primal-dual interior-point methods for self-scaled cones. *SIAM Journal on Optimization*, 8(2):324–364, 1998.

[164] B. Ninness and S. Henriksen. Bayesian system identification via Markov chain Monte Carlo techniques. *Automatica*, 46(1):40–51, 2010.

[165] L. Noakes. The takens embedding theorem. *International Journal of Bifurcation and Chaos*, 1(04):867–872, 1991.

[166] J. M. Ortega and W. C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*, volume 30. SIAM, 1970.

[167] J. Otsuka and T. Masuda. The influence of nonlinear spring behavior of rolling elements on ultraprecision positioning control systems. *Nanotechnology*, 9(2):85, 1998.

[168] A. Packard, U. Topcu, P. Seiler, and G. Balas. Help on SOS. *IEEE Control Systems*, 30(4):18–23, 2010.

[169] J. Paduart, L. Lauwers, J. Swevers, K. Smolders, J. Schoukens, and R. Pintelon. Identification of nonlinear systems using polynomial nonlinear state space models. *Automatica*, 46(4):647–656, 2010.

[170] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975.

[171] P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, May 2000.

[172] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2):293–320, 2003.

[173] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[174] F. Permenter and P. Parrilo. Partial facial reduction: simplified, equivalent SDPs via approximations of the PSD cone. *arXiv preprint arXiv:1408.4685*, 2014.

[175] F. Permenter and P. A. Parrilo. Basis selection for sos programs via facial reduction and polyhedral approximations. In *Proc. of IEEE Conference on Decision and Control*, pages 6615–6620. IEEE, 2014.

[176] V. Peterka. Bayesian system identification. *Automatica*, 17(1):41–53, 1981.

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

[178] G. Pillonetto, M. H. Quang, and A. Chiuso. A new kernel-based approach for nonlinear system identification. *IEEE Transactions on Automatic Control*, 56(12): 2825–2840, 2011.

[179] G. Pillonetto, F. Dinuzzo, T. Chen, G. De Nicolao, and L. Ljung. Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, 50(3):657–682, 2014.

[180] H. Poincaré. Mémoire sur les courbes définies par une équation différentielle (i). *Journal de mathématiques pures et appliquées*, 7:375–422, 1881.

[181] I. Pólik and T. Terlaky. A survey of the S-lemma. *SIAM Review*, 49(3):371–418, 2007.

[182] G. Poole and T. Boullion. A Survey on M-Matrices. *SIAM Review*, 16(4):419–427, 1974.

[183] V.-M. Popov. Absolute stability of nonlinear systems of automatic control. *Automation and Remote Control*, 22(8):857–875, 1962.

[184] V. Powers and T. Wörmann. An algorithm for sums of squares of real polynomials. *Journal of Pure and Applied Algebra*, 127(1):99–104, 1998.

[185] S. J. Qin. An overview of subspace identification. *Computers & Chemical Engineering*, 30(10):1502–1513, 2006.

[186] S. J. Qin and L. Ljung. Closed-loop subspace identification with innovation estimation. *IFAC Proceedings*, 36(16):861–866, 2003.

[187] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*, volume 37. Springer Science & Business Media, 2010.

[188] M. A. Rami. Solvability of static output-feedback stabilization for LTI positive systems. *Systems & control letters*, 60(9):704–708, 2011.

[189] A. Rantzer. On the Kalman-Yakubovich-Popov lemma. *Systems & Control Letters*, 28(1):7–10, 1996.

[190] A. Rantzer. Distributed control of positive systems. *arXiv preprint arXiv:1203.0047*, 2012.

[191] A. Rantzer. On the Kalman-Yakubovich-Popov lemma for positive systems. In *Proc. 51st Decision and Control (CDC)*, pages 7482–7484. IEEE, 2012.

[192] A. Rantzer. Scalable control of positive systems. *European Journal of Control*, 24: 72–80, 2015.

[193] C. E. Rasmussen and C. K. Williams. *Gaussian processes for machine learning*, volume 1. MIT Press Cambridge, 2006.

[194] H. E. Rauch, C. Striebel, and F. Tung. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965.

[195] P. A. Regalia and P. Stoica. Stability of multivariable least-squares models. *IEEE Signal Processing Letters*, 2(10):195–196, 1995.

[196] R. T. Rockafellar. Lagrange multipliers and optimality. *SIAM review*, 35(2): 183–238, 1993.

[197] C. R. Rojas, J. S. Welsh, G. C. Goodwin, and A. Feuer. Robust optimal experiment design for system identification. *Automatica*, 43(6):993–1008, 2007.

[198] M. Safanov. *Stability and Robustness of Multivariable Feedback Systems.* The MIT Press, 1980.

[199] P. Salamon, P. Sibani, and R. Frost. *Facts, conjectures, and improvements for simulated annealing.* SIAM, 2002.

[200] T. Sarkar, M. Roozbehani, and M. A. Dahleh. Robustness scaling in large networks. In *Proc. American Control Conference (ACC)*, pages 197–202. IEEE, 2016.

[201] C. W. Scherer. Lmi relaxations in robust control. *European Journal of Control*, 12 (1):3–29, 2006.

[202] M. Schetzen. *The Volterra and Wiener theories of nonlinear systems.* Wiley & Sons, 1980.

[203] T. B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

[204] T. B. Schön, F. Lindsten, J. Dahlin, J. Wågberg, C. A. Naesseth, A. Svensson, and L. Dai. Sequential Monte Carlo methods for system identification. In *Proceedings of the 17th IFAC Symposium on System Identificatin (SYSID)* , volume 48, pages 775–786, Beijing, China, 2015. Elsevier.

[205] F. Sepehr and D. Materassi. Inferring the structure of polytree networks of dynamic systems with hidden nodes. In *Proc. Decision and Control (CDC)*, pages 4618–4623. IEEE, 2016.

[206] N. Z. Shor. Class of global minimum bounds of polynomial functions. *Cybernetics and Systems Analysis*, 23(6):731–734, 1987.

[207] R. H. Shumway and D. S. Stoffer. An approach to time series smoothing and forecasting using the em algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[208] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. k. Hjalmarsson, and A. Juditsky. Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, 31(12):1691–1724, 1995.

[209] S. Skogestad and I. Postlethwaite. *Multivariable feedback control: analysis and design*, volume 2. Wiley New York, 2007.

[210] B. L. Smith, B. M. Williams, and R. K. Oswald. Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10(4):303–321, 2002.

[211] T. Söderström. On the uniqueness of maximum likelihood identification. *Automatica*, 11(2):193–197, 1975.

[212] T. Söderström. Errors-in-variables methods in system identification. *Automatica*, 43 (6):939–958, 2007.

[213] T. Söderström and P. Stoica. Some properties of the output error method. *Automatica*, 18(1):93–99, 1982.

[214] V. Solo. An EM algorithm for singular state space models. In *Proceedings of the 42nd IEEE Conference on Decision and Control (CDC)*, pages 3457–3460, Maui Maui, USA, 2003.

[215] V. Solo. An EM algorithm for singular state space models II. In *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC)*, pages 3611–3612, The Bahamas, 2004.

[216] E. Sontag. Input to state stability: Basic concepts and results. *Nonlinear and Optimal Control Theory*, pages 163–220, 2008.

[217] E. D. Sontag. Smooth stabilization implies coprime factorization. *IEEE Transactions on Automatic Control*, 34(4):435–443, 1989.

[218] K. C. Sou, A. Megretski, and L. Daniel. A quasi-convex optimization approach to parameterized model order reduction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(3):456–469, 2008.

[219] G. Srijuntongsiri and S. A. Vavasis. A fully sparse implementation of a primal-dual interior-point potential reduction method for semidefinite programming. *arXiv preprint cs/0412009*, 2004.

[220] J. Stark. Delay embeddings for forced systems. I. Deterministic forcing. *Journal of Nonlinear Science*, 9(3):255–332, 1999.

[221] J. Stark, D. S. Broomhead, M. Davies, and J. Huke. Delay embeddings for forced systems. II. Stochastic forcing. *Journal of Nonlinear Science*, 13(6):519–577, 2003.

[222] B. Sturmfels. Polynomial equations and convex polytopes. *The American Mathematical Monthly*, 105(10):907–922, 1998.

[223] Y. Sun, M. S. Andersen, and L. Vandenberghe. Decomposition in conic optimization with partially separable structure. *SIAM Journal on Optimization*, 24(2):873–897, 2014.

[224] A. Svensson and T. B. Schön. A flexible state-space model for learning nonlinear dynamical systems. *Automatica*, 80:189 – 199, 2017.

[225] M. Sznaier. Computational complexity analysis of set membership identification of hammerstein and wiener systems. *Automatica*, 45(3):701–705, 2009.

[226] F. Takens et al. Detecting strange attractors in turbulence. *Lecture Notes in Mathematics*, 898(1):366–381, 1981.

[227] T. Tanaka and C. Langbort. The bounded real lemma for internally positive systems and h-infinity structured static state feedback. *IEEE Transactions on Automatic Control*, 56(9):2218–2223, 2011.

[228] G. Taylor, R. Burmeister, Z. Xu, B. Singh, A. Patel, and T. Goldstein. Training neural networks without gradients: A scalable admm approach. In *International Conference on Machine Learning*, pages 2722–2731, 2016.

[229] M. M. Tobenkin. *Robustness Analysis for Identification and Control of Nonlinear Systems*. PhD thesis, Massachusetts Institute of Technology, 2014.

[230] M. M. Tobenkin, I. R. Manchester, J. Wang, A. Megretski, and R. Tedrake. Convex optimization in identification of stable non-linear state space models. In *Proceedings of the 49th IEEE Conference on Decision and Control, CDC*, pages 7232–7237, Atlanta, USA, 2010.

[231] M. M. Tobenkin, I. R. Manchester, and A. Megretski. Stable nonlinear identification from noisy repeated experiments via convex optimization. In *Proc. of the American Control Conference (ACC)*, pages 3936–3941. IEEE, 2013.

[232] M. M. Tobenkin, I. R. Manchester, and A. Megretski. Convex parameterizations and fidelity bounds for nonlinear identification and reduced-order modelling. *IEEE Transactions on Automatic Control*, pages 3679 – 3686, 2017.

[233] M. M. Tobenkin, F. Permenter, and A. Megretski. SPOTless: polynomial and conic optimization toolbox. Available at: https://github.com/spot-toolbox/spotless . 2017.

[234] M. J. Todd, K.-C. Toh, and R. H. Tütüncü. On the Nesterov-Todd Direction in Semidefinite Programming. *SIAM Journal on Optimization*, 8(3):769–796, 1998.

[235] J. Umenberger and I. R. Manchester. Scalable identification of stable positive systems. In *Proceedings of the 55th Conference on Decision and Control (CDC)*, pages 4630–4635. IEEE, 2016.

[236] J. Umenberger and I. R. Manchester. Specialized algorithm for identification of stable linear systems using lagrangian relaxation. In *Proceedings of the American Control Conference (ACC)*, pages 930–935, Boston, USA, 2016.

[237] J. Umenberger, J. Wågberg, I. R. Manchester, and T. B. Schön. On identification via EM with latent disturbances and Lagrangian relaxation. In *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Beijing, China, 2015.

[238] T. Van Gestel, J. A. Suykens, P. Van Dooren, and B. De Moor. Identification of stable models in subspace identification by using regularization. *IEEE Transactions on Automatic Control*, 46(9):1416–1420, 2001.

[239] P. Van Overschee and B. De Moor. N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1): 75–93, 1994.

[240] P. Van Overschee and B. De Moor. A unifying theorem for three subspace system identification algorithms. *Automatica*, 31(12):1853–1864, 1995.

[241] P. Van Overschee and B. De Moor. *Subspace identification for linear systems: Theory, Implementation, Applications*. Springer Science & Business Media, 2012.

[242] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM review*, 38(1): 49–95, 1996.

[243] L. Vandenberghe, V. R. Balakrishnan, R. Wallin, A. Hansson, and T. Roh. Interior-point algorithms for semidefinite programming problems derived from the lemma. In *Positive Polynomials in Control*, pages 195–238. Springer.

[244] L. Vandenberghe, M. S. Andersen, et al. Chordal graphs and semidefinite optimization. *Foundations and Trends in Optimization*, 1(4):241–433, 2015.

[245] S. A. Vavasis. *Nonlinear optimization: complexity issues*. Oxford University Press, Inc., 1991.

[246] M. Verhaegen. Application of a subspace model identification technique to identify lti systems operating in closed-loop. *Automatica*, 29(4):1027–1040, 1993.

[247] M. Verhaegen and P. Dewilde. Subspace model identification part 1. The output-error state-space model identification class of algorithms. *International Journal of Control*, 56(5):1187–1210, 1992.

[248] H. Waki, S. Kim, M. Kojima, and M. Muramatsu. Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity. *SIAM Journal on Optimization*, 17(1):218–242, 2006.

[249] R. Wallin and A. Hansson. KYPD: a solver for semidefinite programs derived from the Kalman-Yakubovich-Popov lemma. In *Proc. of Int. Symp. of Computer Aided Control Systems Design*, pages 1–6, 2004.

[250] R. Wallin, C.-Y. Kao, and A. Hansson. A cutting plane method for solving KYP-SDPs. *Automatica*, 44(2):418–429, 2008.

[251] E. Walter, L. Pronzato, and J. Norton. *Identification of parametric models from experimental data*, volume 1. Springer Berlin, 1997.

[252] R. W. Wedderburn. Quasi-likelihood functions, generalized linear models, and the gauss-newton method. *Biometrika*, 61(3):439–447, 1974.

[253] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2 (3):203–230, 2010.

[254] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[255] J. Willems and G. Blankenship. Frequency domain stability criteria for stochastic systems. *IEEE Transactions on Automatic Control*, 16(4):292–299, 1971.

[256] J. C. Willems. *Analysis of feedback systems.* The MIT Press, 1971.

[257] J. C. Willems. Dissipative dynamical systems part ii: Linear systems with quadratic supply rates. *Archive for Rational Mechanics and Analysis*, 45(5):352–393, 1972.

[258] A. Wills, T. B. Schön, and B. Ninness. Estimating state-space models in innovations form using the expectation maximisation algorithm. In *Proceedings of the 49th IEEE Conference on Decision and Control (CDC)*, pages 5524–5529, Atlanta, USA, 2010.

[259] A. Wills, T. B. Schön, F. Lindsten, and B. Ninness. Estimation of linear systems using a gibbs sampler. *IFAC Proceedings Volumes*, 45(16):203–208, 2012.

[260] F. W. Wilson. The structure of the level surfaces of a lyapunov function. *Journal of Differential Equations*, 3(3):323–329, 1967.

[261] M. H. Wright. Direct search methods: Once scorned, now respectable. *Pitman Research Notes in Mathematics Series*, pages 191–208, 1996.

[262] S. J. Wright and J. Nocedal. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.

[263] V. Yakubovich. S-procedure in nonlinear control theory. *Vestnik Leningrad University*, 1:62–77, 1971.

[264] Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

[265] G. F. Young, L. Scardovi, and N. E. Leonard. Robustness of noisy consensus dynamics with directed communication. In *Proc. American Control Conference (ACC)*, pages 6312–6317. IEEE, 2010.

[266] G. Zames. On the input-output stability of time-varying nonlinear feedback systems, Part I: Conditions derived using concepts of loop gain, conicity, and positivity. *IEEE Transactions on Automatic Control*, 11(2):228–238, 1966.

[267] G. Zames. On the input-output stability of time-varying nonlinear feedback systems–Part II: Conditions involving circles in the frequency plane and sector nonlinearities. *IEEE Transactions on Automatic Control*, 11(3):465–476, 1966.

[268] R. Zhang and J. Kwok. Asynchronous distributed admm for consensus optimization. In *International Conference on Machine Learning*, pages 1701–1709, 2014.

[269] K. Zhou and J. C. Doyle. *Essentials of robust control*, volume 104. Prentice hall Upper Saddle River, NJ, 1998.

[270] P. Zhou and B. W. Ang. Linear programming models for measuring economy-wide energy efficiency performance. *Energy Policy*, 36(8):2911–2916, 2008.

[271] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603, 2010.