GENERALIZED SYSTEM IDENTIFICATION WITH STABLE SPLINE KERNELS

ALEKSANDR Y. ARAVKIN*, JAMES V. BURKE†, AND GIANLUIGI PILLONETTO ‡

Abstract. Regularized least-squares approaches have been successfully applied to linear system identification. Recent approaches use quadratic penalty terms on the unknown impulse response defined by *stable spline kernels*, which control model space complexity by leveraging regularity and bounded-input bounded-output stability. This paper extends linear system identification to a wide class of nonsmooth stable spline estimators, where regularization functionals and data misfits can be selected from a rich set of piecewise linear-quadratic (PLQ) penalties. This class includes the 1-norm, Huber, and Vapnik, in addition to the least-squares penalty.

By representing penalties through their conjugates, the modeler can specify *any* piecewise linear-quadratic penalty for misfit and regularizer, as well as inequality constraints on the response. The interior-point solver we implement (IPsolve) is locally quadratically convergent, with $O(\min(m,n)^2(m+n))$ arithmetic operations per iteration, where n the number of unknown impulse response coefficients and m the number of observed output measurements. IPsolve is competitive with available alternatives for system identification. This is shown by a comparison with TFOCS, libSVM, and the FISTA algorithm. The code is open source[1].

The impact of the approach for system identification is illustrated with numerical experiments featuring robust formulations for contaminated data, relaxation systems, nonnegativity and unimodality constraints on the impulse response, and sparsity promoting regularization. Incorporating constraints yields particularly significant improvements.

Keywords: linear system identification; kernel-based regularization; Gaussian processes; bias-variance; model order selection; robust statistics; sparse optimization; interior point methods

1. Introduction. System identification formalizes the process of inferring models from observations and studying their properties. A *system* comprises multiple variable interactions to produce observable signals [45]. We focus on *linear time invariant* system (LTI) identification, i.e. systems where the response to a certain input signal does not depend on absolute time, and the output response to a linear combination of inputs is the linear combination of the responses to individual inputs. This class is computationally tractable, and has been successful in a wide range of applications including NMR spectroscopy, seismology, circuits, signal processing, control theory, biological processes, and many others. Many techniques have been developed to identify LTI systems, in state space and frequency domains (see eg.[45]). We focus on on identifying impulse responses from input-output data in the time domain.

Within the class of LTI systems, model selection and model space exploration is a key concept. Classic approaches build parametric models of different orders using autoregressive (moving average) models with exogenous inputs AR(MA)X, fit to data using Prediction Error Methods (PEM) [45, 63]. The "best" model is selected using complexity measures such as Akaike information criterion (AIC), Bayesian information criterion (BIC) or by cross validation (CV) techniques [2, 61, 35].

This approach has a number of limitations [55, 56]. For example, when the number of available output measurements is relatively small (so that asymptotic theory underlying AIC is not applicable), the selected models often have poor predictive capability on new data. In some cases, identifying a system is a highly ill-conditioned inverse problem [14], and principled regularization techniques are required for estimation. The classical approach cannot incorporate additional information about the system, including domain restrictions, monotonicity, and unimodality; this information can dramatically improve estimation. Finally, reliance on

*Applied Mathematics Department, University of Washington, Seattle, WA, (saravkin@uw.edu). Research was supported by the Washington Research Foundation Fund for Innovation in Data-Intensive Discovery.

†Mathematics Dept., University of Washington, Seattle, WA 98195 (burke@math.washington.edu). Research is supported in part by the National Science Foundation under grant number DMS-1514559.

‡Control and Dynamic Systems Department of Information Engineering at the University of Padova, Padova, Italy (giapi@dei.unipd.it)

July 27, 2018

[1]https://github.com/saravkin/IPsolve.

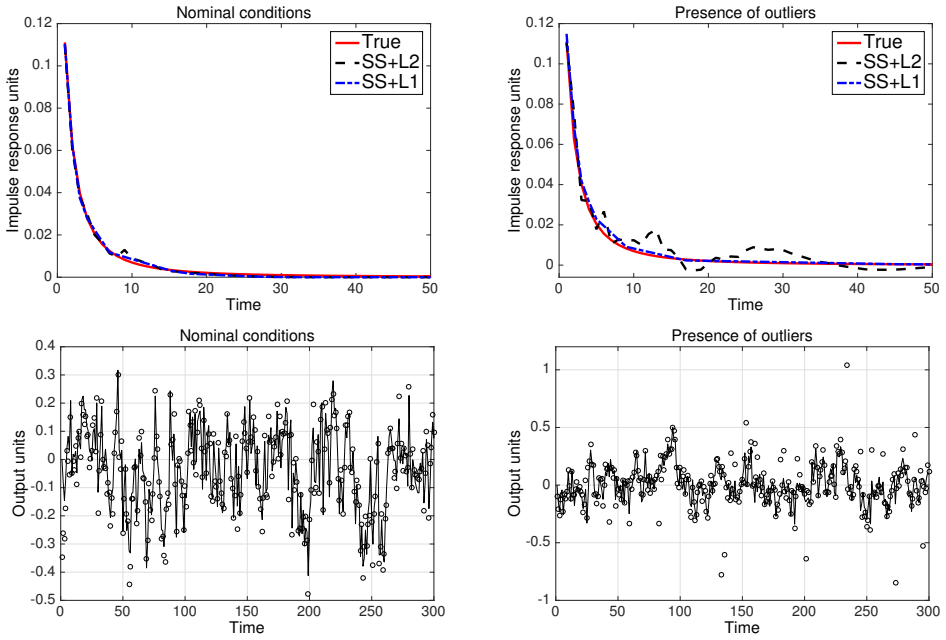


Fig. 1.1: *Top:* true impulse response (red solid) and estimates obtained under nominal conditions (left) and in presence of outliers (right) using the stable spline estimator (Section 2.1) with L_2 loss (black dash) and L_1 loss (blue dash-dot). *Bottom:* linearly interpolated noiseless outputs (solid line), and noisy measurements $(\circ)$ under nominal conditions (left) and in presence of outliers (right). The L_1 estimator performs much better in the presence of outliers.

least-squares leaves the system identification process vulnerable to model mis-specification, including outliers in the observations [36, 28, 4, 26].

Illustrative example. Some of the drawbacks to the standard approach are illustrated by the example in Fig. 1.1 (implementation details are given in Section 5). The impulse response (top panels, solid red line) is estimated from noisy outputs obtained using white noise as system input. Consider two different situations. In the first case, the data set of 1000 samples is corrupted by white stationary Gaussian noise (bottom left panel). The estimate obtained using the stable spline estimator (discussed in Section 2.1) using the L_2 loss (top left, black dashdot line) is close to the true signal. In the second case, the data set is corrupted by a few outliers (bottom right panel). Now, the impulse response profile (top right panel, black dashdot line) reveals the vulnerability of the L_2 loss to deviations in the noise model.

Contributions. We build a modeling framework for LTI system identification, together with an open source solver (IPsolve)[2] to fit all of the models of interest. The modeler can choose from a range of convex *piecewise linear-quadratic (PLQ)* penalties to use as misfit and regularizer. A particularly effective regularization strategy uses the *stable spline kernel* approach to control model space complexity. The modeler can also incorporate constraints on the signal response, significantly narrowing the search when additional information is available. We now briefly describe each component of the proposed framework.

Piecewise linear-quadratic penalties (PLQ). The limitations of L_2 motivate outlier-robust

[2]https://github.com/saravkin/IPsolve

losses, including the L_1-norm, Huber [36], Vapnik [65, 58] and the hinge loss [25, 60]. All of these penalties fall into the PLQ class, and appear in Figures 3.1a-3.1h. Just as the L_2 penalty corresponds to maximum likelihood estimation with Gaussian errors, other penalties can be viewed as negative log likelihoods for non Gaussian noise [29]. The L_1 loss corresponds to assuming the noise follows a Laplacian distribution [4], and analogous likelihood interpretations have been developed for Vapnik and Huber penalties [9].

Stable-spline kernels. Recent approaches cast system identification as a kernel learning problem, formulated in a Hilbert space [55, 54, 57]. Ill-posedness and ill-conditioning are studied within a Gaussian regression framework [59]. The unknown impulse response is modeled as a Gaussian process whose covariance encodes available prior knowledge, and estimators proposed in [55, 53] model covariances using *stable spline kernels*, which include information on regularity and exponential stability of the impulse response. Stable spline estimators have significant advantages over the classical approach, especially in terms of the quality of the model complexity selection [57].

General regularizers. Though quadratic penalization of stable spline coefficients has proved to be effective, other choices can yield dramatic improvements. For example, if the impulse response is expected to have many zero entries, the inclusion of a sparsity promoting prior can significantly improve the quality of the estimator, e.g. the Laplace prior, or L_1 loss, used in the LASSO [64]. This leads to a weighted combination of norms in the spirit of the elastic net procedure [71]. Our framework includes these priors, allowing any PLQ penalty to be used as a regularizer or data misfit, see Figures 3.1a-3.1h.

Incorporating constraints. Additional information can be incorporated using inequality constraints on the impulse response coefficients. *Nonnegativity* is often a consequence of physical considerations. Relaxed systems with *monotonic responses* are frequently encountered in reciprocal electrical networks and mechanical systems with negligible inertial phenomena [66] see e.g. the impulse response Fig. 1.1. A third example is bolus-tracking magnetic resonance imaging (MRI) [70], where quantification of cerebral hemodynamics requires estimation of impulse responses known to be *positive and unimodal*. In all of these examples, there is a wealth of prior knowledge about the signal, and incorporating this knowledge into system identification can yield significant improvements (see Figures 5.1 and 5.3).

IPsolve. Convex optimization has become a standard tool in many applications, and general convex solvers such as the MATLAB package CVX [31] and TFOCS [11] are important tools for prototyping and testing new ideas. However, these general tools are not competitive with solvers that exploit problem-specific structure. IPsolve strikes a good balance between generality and problem structure, and outperforms TFOCS in the system identification context. IPsolve can solve any PLQ problem, but is less general than TFOCS, which can in principle solve any convex problem. On the other hand, IPsolve is much more general than solvers targeted to specific problem classes (e.g. L_1 regularization of smooth problems, or convex quadratic solvers). Nevertheless, the numerical experiments show that it is still competitive with targeted solvers for system identification. In particular, for system identification problems that can be formulated as support vector regression (SVR), we compare IPsolve to libSVM [16], a state of the art solver for this SVR. Similarly, for sparse system identification, IPsolve is competitive with fast iterative soft thresholding (FISTA) [10], an optimal first-order method designed for smooth problems with simple regularizers. In general, first-order methods are faster than interior point methods for large-scale composite problems [6]. However, both system identification problems and stable spline kernels can be ill-conditioned, and interior-point methods are well-suited to ill-conditioned problems, which explains the result.

We extend prior general work in PLQ modeling and optimization [7, 9] by incorporating inequality constraints, and develop convergence guarantees for the entire framework in The-

orem 4.3. We then apply the constrained PLQ framework to the linear system identification scenario, and compare resulting estimators with classical PEM and stable spline approaches in a range of numerical studies, featuring contaminated data, and the inclusion of additional information about the impulse response, e.g. unimodality or complete monotonicity.

Road map. In Section 2, we review the classical approach to linear system identification, and provide a brief introduction to the stable spline estimation technique. In Section 3, we formulate the general class of nonsmooth stable spline estimators, using PLQ penalties as misfits and regularizers, and incorporate inequality constraints on the impulse response. We develop a general IP method for the class in Section 4, along with specific results for the system identification setting. In Section 5, we compare against TFOCS, libSVM and FISTA to illustrate scaling and efficiency of IPsolve, and also test the performance of new estimators using several Monte Carlo studies, including estimators for the example discussed in the introduction. While most of the experiments focus on the regime $n << m$, in subsection 5.5 we develop a sparse and stable estimator for the high-dimensional setting ($n >> m$) that arises in identifying multiple input single output (MISO) systems. This approach can be used to simultaneously identify connectivity and estimate models in dynamic networks[18].

2. PEM and stable spline approaches to linear system identification. Consider the following linear time-invariant discrete-time system

$$z(t) = G(q)u(t) + e(t) \tag{2.1}$$

where $z \in \mathbb{R}^m$ is the output , q is the shift operator $qu(t) = u(t+1)$, $G(q)$ is the linear operator associated with the true system, assumed stable, u is the input, and e is white noise of variance σ^2. Our problem is to estimate the system impulse response assuming that the system input is known for m measurements of z at instants $t = 1, \ldots, m$. We then measure the quality of an estimator $\widehat{G}$ by means of the fit measure

$$\mathscr{F}(G, \hat{G}) = 100\left(1 - \|G - \widehat{G}\|_2 / \|G\|_2\right) \tag{2.2}$$

where, given a linear system $S(q)$, $\|S\|_2$ is the L_2-norm of its impulse response.

The classic approach to system identification represents a parametrized model space $\mathscr{M}$ for linear systems by a transfer function G from input to output, parameterized by x:

$$z(t) = G(q,x)u(t) + e(t). \tag{2.3}$$

For instance, a standard *black box* description assumes G is a rational function of the shift operator q,

$$G(q,x) = B(q)/C(q) \tag{2.4}$$

where $B(q)$ and $C(q)$ are polynomials in q^{-1} whose unknown coefficients are the components of $x \in \mathbb{R}^n$. Different model structures can be associated with different degrees of $B(q)$ and $C(q)$. For each model structure, the state x can be estimated by PEM [45], i.e.

$$\hat{x} = \operatorname*{argmin}_x V(x), \quad V(x) = \sum_{t=1}^{m} \big(z(t) - G(q,x)u(t)\big)^2. \tag{2.5}$$

where the quadratic loss V is a standard choice. In real applications, a suitable model structure (dimension of x) is typically unknown and needs to be inferred from data. This step is crucial, as it balances bias and variance, and popular approaches include cross validation [35], Akaike's criterion [2], and its small-sample version, corrected Akaike's criterion (AICc) [37].

2.1. The stable spline estimator. A drawback to rational transfer functions is that they require solving (2.5), a nonconvex and potentially high-dimensional problem, for each postulated model order. A common alternative is the finite impulse response (FIR) model obtained setting $C(q) = 1$ in (2.4), which makes (2.5) a linear least-squares problem in the polynomial coefficients x for B:

$$V(x) = \sum_{t=1}^{m} \big(z(t) - B_x(q)u(t)\big)^2 = \sum_{t=1}^{m} (z_t - \langle \phi(u_t, q), x\rangle)^2, \tag{2.6}$$

where $\phi(u_t, q)$ is a vector determined by the input and shift operators. However, a high-order FIR, often necessary to capture system dynamics, can suffer from high variance, so regularization is crucial.

We quickly review the regularized approaches described in [55, 17]. First rewrite the measurement model (2.6) using matrix-vector notation:

$$z = \Phi x + E, \tag{2.7}$$

where $z \in \mathbb{R}^m$ is a vector comprising the m output measurements, E is the noise, $x \in \mathbb{R}^n$ is the (column) vector of impulse response coefficients, and Φ is a matrix with rows $\phi_t = \phi(u_t, q)$ determined by input values and shift operator. For instance, assuming an input delay of one sample, we have

$$z = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{pmatrix}, \quad \Phi = \begin{pmatrix} u_0 & u_{-1} & \dots & u_{-n+1} \\ u_1 & u_0 & \dots & u_{-n+2} \\ \vdots & \vdots & \vdots & \vdots \\ u_{m-1} & u_{m-2} & \dots & u_{-n+m} \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

In contrast to classical approaches to system identification, the nth-order FIR approach does not need to balance bias and variance, but only needs to be of sufficiently large order to capture the system dynamics. The model complexity is controlled via stable spline kernels. A stable spline estimator for impulse response solves

$$\hat{x} = \arg\min_x \|z - \Phi x\|^2 + \gamma x^T Q^{-1} x\,, \tag{2.8}$$

where the positive scalar γ is the *regularization parameter*, while $Q \in \mathbb{R}^{n\times n}$ is a regularization matrix defined by the class of the stable spline kernels [53]. The choice of γ is important: ideally, it must be tuned so that a small bias is introduced in the estimation process in order to significantly reduce the variance (in comparison with a baseline such as least squares).

Problem (2.8) is always well-posed because of the strongly convex term $\gamma x^T Q^{-1} x$, which also controls model order. When using the discrete-time version of the first-order stable spline kernel (also called TC kernel in [17]), the (i, j) entry of Q is specified to be

$$Q_{ij} := \alpha^{\max(i,j)}, \quad \text{for some } 0 \le \alpha < 1\,. \tag{2.9}$$

Smoother impulse response estimates can be obtained by using the second-order stable spline kernel. In this case, the entries of Q are given by

$$Q_{ij} := \Big[\alpha^{(i+j)}\alpha^{\max(i,j)}/2 - \alpha^{3\max(i,j)}/6\Big], \quad \text{for some } 0 \le \alpha < 1\,. \tag{2.10}$$

The kernel Q may be ill-conditioned (the condition number grows to infinity with small α and large n). Stability properties of first-order stable-spline kernels, as well as formulas for

their inverses and Cholesky factors are discussed in detail in [15]. While ill-conditioning of the kernel can partially controlled by hyper-parameter selection, general system identification problems may be ill-conditioned, and the implications are discussed in Section 5.

In (2.9) and (2.10), α is a kernel hyperparameter related to the dominant pole of the system (i.e. it establishes how fast the impulse response decays to zero) and is typically unknown. The estimator (2.8),where Q is given by (2.9) or (2.10), depends on α and γ, which need to be determined from data using marginal likelihood maximization [48, 46, 13, 7, 6]. Learning hyperparameters is analogous to model order selection in the classical PEM framework. Once the two parameters are found, the impulse response estimate for the least-squares formulation can be obtained by solving a linear system of equations given by the first-order optimality conditions for the problem (2.8).

3. New formulations of the stable spline estimator. We now introduce the class of piecewise linear quadratic (PLQ) functions and penalties. We first develop a representation calculus for estimators of interest using simple PLQ building blocks, and then show how to formulate general estimation problems as minimizers of a single PLQ objective over a polyhedral set. All estimators are obtained using the algorithm developed in Section 4.

3.1. From quadratic to PLQ penalties. It is useful to rewrite the estimator (2.8) as:

$$y = L^{-1}x, \quad Q = LL^T \tag{3.1}$$

where L is invertible thanks to the positive definite property of stable spline kernels (we caution that the variable y introduced here is *not* the same object as the function $y(t)$ defined in (2.1)). In the new variable y the estimation problem (2.8) translates into the problem

$$\min_y \|z - \Phi Ly\|^2 + \gamma\|y\|^2 . \tag{3.2}$$

This estimator uses quadratic functions for both the misfit penalty and the regularizer. The goal of the remainder of the paper is to show how to generalizing (3.2) for adaptation to a variety of data scenarios and to illustrate the computational efficacy of these adaptations.

Let Y denote the feasible polyhedral constraint region for y. Then an explicit representation for Y can be written as

$$Y = \{y : A^T y \leq a\} , \quad A \in \mathbb{R}^{n\times p}, \quad a \in \mathbb{R}^p. \tag{3.3}$$

This allows us to represent prior knowledge about the signal, including domain information (e.g. lower and upper limits), as well as monotonicity or unimodality properties.
We consider generalizations of (3.2) that use any PLQ penalty:

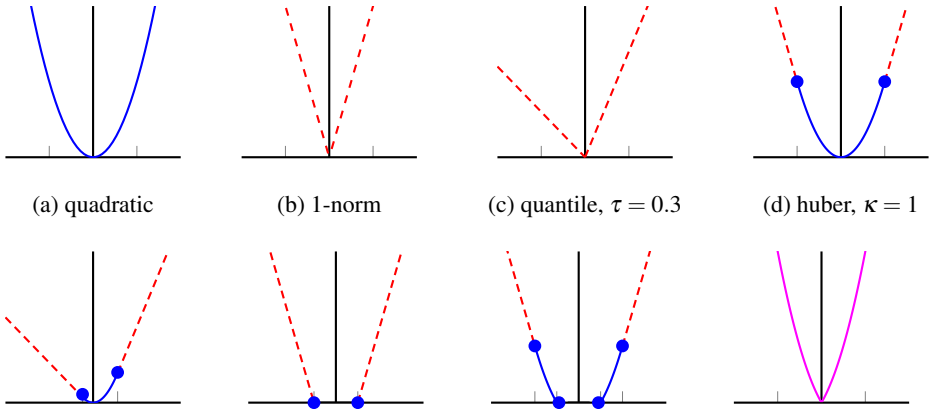
$$\min_{y\in Y} V(z - \Phi Ly) + \gamma W(y), \tag{3.4}$$

where V and W are piecewise linear quadratic functions introduced below, and Y is as in (3.3). Nine important examples of these penalties appear in Figures 3.1a-3.1h.

DEFINITION 3.1 (PLQ functions and penalties). *A piecewise linear quadratic (PLQ) function is any function $\rho(c,C,b,B,M;\cdot) : \mathbb{R}^n \to \overline{\mathbb{R}}$ admitting representation*

$$\rho(c,C,b,B,M;y) = \sup_{u\in U}\left\{ \langle u, b + By\rangle - \frac{1}{2}\langle u, Mu\rangle \right\} , \tag{3.5}$$

where $U = \{u : C^T u \leq c\}$ is a polyhedral set containing the origin, M is symmetric positive semidefinite, $\in \mathbb{R}^k$, $B \in \mathbb{R}^{k\times n}$ with $null(B) = \{0\}$.



(a) quadratic (b) 1-norm (c) quantile, $\tau = 0.3$ (d) huber, $\kappa = 1$

(e) quantile huber (f) vapnik, $\varepsilon = 0.5$ (g) huber ins. loss (h) elastic net, $\alpha = 0.5$

Fig. 3.1: Common piecewise linear-quadratic (PLQ) losses.

Table 3.1: Dual representations of common PLQ penalties.

Penalty	Representation (3.5)	Selected references
Quadratic, Fig. 3.1a	$\sup_u \left\{ux - \frac{1}{2}u^2\right\}$	[27, 62]
1-norm, Fig. 3.1b	$\sup_{u\in[-1,1]} \{ux\}$	[34, 24, 23, 47]
Quantile, Fig. 3.1c	$\sup_{u\in[-\tau,(1-\tau)]} \{ux\}$	[39, 40, 38]
Huber, Fig. 3.1d	$\sup_{u\in[-\kappa,\kappa]} \left\{ux - u^2/2\right\}$	[36, 49, 44]
Q-Huber, Fig. 3.1e	$\sup_{u\in[-\kappa\tau,\kappa(1-\tau)]} \left\{ux - u^2/2\right\}$	[1]
Vapnik, Fig. 3.1f	$\sup_{u\in[0,1]^2} \left\langle \begin{bmatrix} 1 \\ -1 \end{bmatrix} x - \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, u \right\rangle$	[65, 35, 60, 5]
SEL, Fig. 3.1g	$\sup_{u\in[0,1]^2} \left\{ \left\langle \begin{bmatrix} 1 \\ -1 \end{bmatrix} x - \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, u \right\rangle - u^2/2 \right\}$	[19, 42, 22]
Elastic net, Fig. 3.1h	$\sup_{u\in[0,1]\times\mathbb{R}} \left\{ \left\langle \begin{bmatrix} 1 \\ 1 \end{bmatrix} x, u \right\rangle - u_2^2/2 \right\}$	[72, 71, 43, 21]

The eight loss functions illustrated in Figures 3.1a-3.1h are members of the PLQ class; dual representations (3.5) and references are given in Table 3.1. In the following section, we use these representations to develop fast algorithms.

Modeling with PLQ. We can classify PLQs according to three features: behavior at origin, symmetry, and tail growth. Several situations are considered in the simulation studies.
Origin. Nonsmooth behavior at origin promotes sparsity. When used as a regularizer, the 1-norm and quantile loss find sparse solutions. When used as a misfit, these losses fit some of the data exactly, quadratic behavior at the origin will fit data approximately, and the Vapnik misfit corresponds to uniform residuals.
Tail growth. Tail growth allows robustness to outliers. Vapnik, 1-norm, and Huber all have similar robustness properties. The quadratic and elastic net losses are not robust to outliers.
Symmetry. Asymmetric losses model cases where either (a) positive or negative responses are more likely (asymmetric regularizer), or (b) costs for over-estimating or under-estimating

observations are different (asymmetric loss).

The representation in Definition 3.1 is explicitly used to solve the generalized linear system identification problem (3.4). We first derive a PLQ representation calculus.

REMARK 3.1 (Affine composition). *Take any PLQ function $\rho(c,C,b,B,M;y)$. Suppose that $y = Ex+e$, where $x \mapsto Ex+e$ is an injective affine transformation in x. Then we have*

$$\rho(c,C,b,B,M;Ex+e) = \rho(c,C,b+Be,BE,M;x)$$

so the composition is also a PLQ function, with representation $c,C,b+Be,BE,M$.

REMARK 3.2 (PLQ addition). *Given two PLQ functions $\rho(c_1,C_1,b_1,B_1,M_1;y)$ and $\rho(c_2,C_2,b_2,B_2,M_2;y)$, the sum is also a PLQ function, with representation*

$$c=\begin{bmatrix}c_1\\c_2\end{bmatrix},\ C=\begin{bmatrix}C_1 & 0\\0 & C_2\end{bmatrix},\ b=\begin{bmatrix}b_1\\b_2\end{bmatrix},\ B=\begin{bmatrix}B_1\\B_2\end{bmatrix},\ M=\begin{bmatrix}M_1 & 0\\0 & M_2\end{bmatrix}.$$

The PLQ class is closed under addition and affine composition, allowing the design of a PLQ penalty that is well suited to a given application. For given PLQ penalties V and W, their sum (3.4) is also a PLQ penalty, with a representation that can be automatically constructed from individual components using the above remarks. Once a representation for (3.4) is constructed, can optimize it over any polyhedral set, as shown in the next section.

4. An Interior Point (IP) Approach. We now show how to solve

$$\min_y \quad \rho(c,C,b,B,M;y) \quad \text{s.t.} \quad A^T y \le a, \tag{4.1}$$

using interior point IP methods [41, 50, 68]. IP methods solve nonsmooth optimization problems by applying a damped Newton method to a homotopy path that parametrizes the underlying Karush-Kuhn-Tucker (KKT) system. In this regard, the first key observation is that the KKT system for (4.1) is an instance of a *monotone mixed linear complementarity problem* (MLCP) [68] since it can be written as

$$\begin{pmatrix}s\\r\\0\\0\end{pmatrix} = \left[\begin{array}{cc|cc} 0 & 0 & -C^T & 0\\ 0 & 0 & 0 & -A^T\\ \hline C & 0 & M & -B\\ 0 & A & B^T & 0\end{array}\right]\begin{pmatrix}q\\w\\u\\y\end{pmatrix} + \begin{pmatrix}c\\a\\b\\0\end{pmatrix} \tag{4.2}$$

with

$$0 \le \begin{pmatrix}q\\w\end{pmatrix},\ \begin{pmatrix}s\\r\end{pmatrix} \text{ and } \begin{pmatrix}q\\w\end{pmatrix}^T\begin{pmatrix}s\\r\end{pmatrix} = 0\,, \tag{4.3}$$

where the matrix in (4.2) is positive semi-definite (see Appendix for details). Consequently, it is possible to transform this MLCP into a monotone LCP and solve it by an interior point algorithm [41]. However, this transformation is arduous, especially in high dimensions, and may be prohibitively expensive [3, 33, 67]. In [67] it is noted that the transformation to an LCP is not essential if the matrix

$$\mathscr{H} := \begin{bmatrix}-C^T & 0\\ 0 & -A^T\\ M & -B\\ B^T & 0\end{bmatrix} \in \mathbb{R}^{(\ell+p+k+n)\times(k+n)} \tag{4.4}$$

is injective. In our context, the injectivity of this matrix can be established under mild conditions.

THEOREM 4.1 (Injectivity of $\mathscr{H}$). *Suppose M is symmetric positive semidefinite and $null(B) = \{0\}$. Then the matrix $\mathscr{H}$ in* (4.4) *is injective if and only if*

$$\mathrm{Nul}\,(M) \cap \mathrm{Nul}\left(B^T\right) \cap \mathrm{Nul}\left(C^T\right) = \{0\}\,. \tag{4.5}$$

Condition (4.5) is satisfied if the stronger condition

$$\mathrm{Nul}\,(M) \cap \mathrm{Nul}\left(B^T\right) = \{0\} \tag{4.6}$$

holds. This latter condition is satisfied by all of the PLQ functions in Section 3.1 [9, 8]. We first specify the dimensions of the quantities appearing in (4.1). Let

$$b \in \mathbb{R}^k, C \in \mathbb{R}^{k\times l}, c \in \mathbb{R}^l, B \in \mathbb{R}^{k\times n}, A \in \mathbb{R}^{n\times p}, M \in \mathbb{R}^{k\times k}, \text{ and } a \in \mathbb{R}^p, \tag{4.7}$$

and set $N := 2l+2p+k+n$. Then, given $\mu \geq 0$, define $F_\mu : \mathbb{R}^N \to \mathbb{R}^N$ by

$$F_\mu(q,w,u,y,s,r) := \begin{pmatrix} C^T u + s - c \\ A^T y + r - a \\ Mu + Cq - By - b \\ B^T u + Aw \\ Qs - \mu \mathbf{1} \\ Wr - \mu \mathbf{1} \end{pmatrix}, \tag{4.8}$$

where $Q := \mathrm{diag}(q)$ and $W := \mathrm{diag}(w)$. The KKT conditions (4.2)-(4.3) are

$$F_0(q,w,u,y,s,r) = 0 \text{ for } s,q \in \mathbb{R}^l_+ \text{ and } r,w \in \mathbb{R}^p_+. \tag{4.9}$$

The variables $y \in \mathbb{R}^n$ and $u \in \mathbb{R}^k$ are those that appear in the definition of the PLQ function ρ (3.5), s and r are *slack variables*, and q, w are the dual variables that correspond to constraints $C^T u \leq c$ and $A^T y \leq a$. For any positive integer ℓ, we set $\mathbb{R}^\ell_+ := \{x \in \mathbb{R}^\ell : 0 \leq x_i,\, i = 1,2,\ldots,\ell\}$ and denote the interior of $\mathbb{R}^\ell_+$ by $\mathbb{R}^\ell_{++}$.

An interior point approach applies a damped Newton iteration to a relaxed version of the KKT system by solving (4.9) for $\mu > 0$ and letting μ carefully descend to zero. We choose an initial $(y^0,u^0) \in \mathbb{R}^n \times \mathbb{R}^k$ and $(s^0,r^0,q^0,w^0) \in \mathscr{D}_{++}$ where $\mathscr{D}_{++} := \mathbb{R}^l_{++} \times \mathbb{R}^p_{++} \times \mathbb{R}^l_{++} \times \mathbb{R}^p_{++}$, and then preserve positivity of the iterates $(s^\nu,r^\nu,q^\nu,w^\nu)$ at each iteration of the damped Newton method for (4.9). For this to succeed, the Newton iteration must be well-defined; in particular $F^{(1)}_\mu$ must be invertible at all iterates when $\mu > 0$. On $\mathscr{D}_{++}$, we have

$$F^{(1)}_\mu(q,w,u,y,s,r) = \left[\begin{array}{cc|cc|cc} 0 & 0 & -C^T & 0 & -I & 0 \\ 0 & 0 & 0 & -A^T & 0 & -I \\ \hline C & 0 & M & -B & 0 & 0 \\ 0 & A & B^T & 0 & 0 & 0 \\ \hline Q & 0 & 0 & 0 & S & 0 \\ 0 & W & 0 & 0 & 0 & R \end{array}\right], \tag{4.10}$$

where $S = \mathrm{diag}(s)$ and $R = \mathrm{diag}(r)$. We now show that the invertibility of $F^{(1)}_\mu$ is related to the condition (4.5) in Theorem 4.1.

THEOREM 4.2 (Invertibility of $F^{(1)}_\mu$). *Given $(u,y) \in \mathbb{R}^k \times \mathbb{R}^n$ and $(q,w,s,r) \in \mathscr{D}_{++}$, the matrix $F^{(1)}_\mu(q,w,u,y,s,r)$ is invertible if and only if the matrix*

$$\begin{bmatrix} M + CSQ^{-1}C^T & -B \\ B^T & ARW^{-1}A^T \end{bmatrix} \tag{4.11}$$

is invertible, which, in turn, is equivalent to condition (4.5).

Note that the central 2×2 block of $F_\mu^{(1)}(q,w,u,y,s,r)$ is precisely the block matrix appearing in the lower right of the MLCP matrix (4.2), and Theorem 4.2 relates (4.5) to the algebraic implementation of the interior point method for the general problem (4.1), detailed in Algorithm 1. To simplify notation we let $\chi = \begin{bmatrix} q^T & w^T & u^T & y^T & s^T & r^T \end{bmatrix}^T$.

Algorithm 1 Interior Point Method for (3.4).

Input: $x^0, y^0, u^0, s^0, r^0, q^0, \eta \in (0,1), \mu^0 \in \mathbb{R}_{++}, \text{tol} > 0$

1: **while** not converged **do**
2: $\quad d \leftarrow -(F_\mu^{(1)})^{-1} F_\mu$
3: $\quad t \leftarrow \max\ \tau \quad \text{s.t.} \quad \left\{ \begin{bmatrix} s + \tau d_s \\ r + \tau d_r \end{bmatrix}, \begin{bmatrix} q + \tau d_q \\ w + \tau d_w \end{bmatrix} \right\} > 0,\ \|F_\mu(\chi + \tau d)\| \leq (1-\eta)\|F_\mu(\chi)$
4: $\quad \chi \leftarrow \chi + td$
5: $\quad \mu \leftarrow 0.1(s^T q + r^T w)/(p+l)$
6: $\quad \text{converged} \leftarrow (\mu < \text{tol})$
7: **end while**
return χ

Let $\tau > 0$ and define

$$\mathscr{F}_+(\tau) := \left\{ (q,w,u,y,r,s) \;\middle|\; \begin{array}{c} (u,y) \in \mathbb{R}^k \times \mathbb{R}^n,\ (q,w,s,r) \in \mathscr{D}_{++} \\ \text{and equation (4.2) is satisfied} \\ \text{with } q^T s + w^T r \leq \tau \end{array} \right\}$$

and $\mathscr{C} := \{(q,w,u,y,r,s) \,|(4.9) \text{ holds for some } \mu > 0\}$. The set $\mathscr{C}$ is called the *central path*. The key to the complexity analysis for the algorithm is to ensure that the iterates hew sufficiently close to this path as μ descends to zero.

THEOREM 4.3 (Convergence Properties). *Consider any optimization problem of the form* (4.1) *satisfying* (4.7). *If* $\mathscr{F}_+(+\infty) \neq \emptyset$ *and* (4.5) *holds, then Algorithm 1 is implementable, the sets* $\mathscr{F}_+(\tau)$ *are non-empty, convex, and bounded for all* $\tau > 0$*, the central path is well-defined, and every cluster point of the central path as* $\mu \downarrow 0$ *is a KKT point for* (4.1).

The proof and details for computing the Newton step are given in the appendix. Matrices

$$T := M + CSQ^{-1}C^T \text{ and } \Omega := B^T T^{-1} B + ARW^{-1}A^T \tag{4.12}$$

and their inverses play a key role; T is invertible since (4.6) holds, and Ω is invertible since T is invertible and B is injective. The sparsity of these matrices determine the complexity of the algorithm, computing the Newton step is the main effort at each iteration. In all our PLQ examples, M and C are very sparse, and the matrix T is diagonal. The next result describes the per iteration complexity of the algorithm in this setting.

THEOREM 4.4 (PLQ Iteration Complexity). *If the matrices* $T_k := M + CS_kQ_k^{-1}C^T$ *in Algorithm 1 are diagonal, then every interior point iteration can be computed with complexity* $O(\min(n,k+p)^2(k+p+n))$.

If $n < k+p$, we obtain the complexity above by forming and inverting Ω in (4.12). Otherwise, we apply the Sherman-Morrison-Woodbury formula to form and invert a matrix in dimension (k+p). The choice is made by IPsolve based on the dimensions of the inputs. Turning our attention back to system identification, n is the dimension of the impulse response, while k and l depend on m, in particular $k \geq m$, while l depends on the structure of the PLQ penalties used to build the estimate (3.4).

COROLLARY 4.5 (SysID Iteration Complexity). *If the constraint matrix A contains* $O(n)$

entries (as e.g. with box constraints), while matrices B and C have on the order of m entries, each interior point iteration can be solved with complexity $O(\min(m,n)^2(m+n))$.

The above result shows that IP computational complexity scales favorably with the number of measurements m which, in system identification, is typically much larger than the number of unknown impulse response coefficients n.

5. Numerical studies. The new approach is now tested using numerical studies.

5.1. Introductory example: robust estimation with inequality constraints . Results in Fig. 1.1 illustrate the well established fact that the estimator $SS+L_2$ based on the quadratic loss is vulnerable to outliers (recall that *SS* refers to stable spline in Section 2.1). Here we exploit the general framework of the previous section to design a robust estimator by replacing the quadratic (Gaussian) with the absolute value (Laplace) loss. This leads to the estimator $SS+L_1$ defined by

$$\hat{x} = \arg\min_x \|z-\Phi x\|_1 + \gamma x^T Q^{-1} x\,. \tag{5.1}$$

This objective can be transformed into form (3.4) and then into (4.1) as described in Section 3. As in the quadratic case (2.8), the solution depends on the unknown parameters γ and α (which enters Q). To solve (5.1), γ and α are estimated via cross validation, splitting the data into a training and a validation set of equal size. The "optimal" hyperparameters values are obtained by searching over a two dimensional grid. In particular, α and γ assume values on the grid defined, respectively, by the MATLAB commands

```
A=[0.01 0.05:0.05:0.95 0.99],
B=logspace(log10(g/100),log10(g*100),50),
```

with `g` set to the value of γ adopted by $SS+L_2$.

The top panels of Fig. 1.1 display the impulse response estimates obtained by $SS+L_1$ (dashed line). The advantage of the new robust formulation is evident. While $SS+L_1$ and $SS+L_2$ exhibit a similar performance under nominal conditions (top left panel), $SS+L_1$ outperforms $SS+L_2$ in presence of outliers (top right panel), returning an impulse response estimate much close to truth. The robustness of the L_1 loss w.r.t. large model deviations is due to the fact that it pushes some residuals to zero. It thus detects which measurements are more accurate, essentially treating them as constraints during the fitting.

To further compare $SS+L_2$ and $SS+L_1$, Fig. 5.1 plots the fit (2.2) returned by these two estimators as a function of the regularization parameter γ (with α constant, fixed to its estimate). The figure reveals that the adoption of the quadratic loss makes it difficult to choose the regularization parameter: many values of γ lead to poor estimates, e.g. $\gamma \leq 1$ leads to outliers overfitting. On the other hand the fit profile associated with $SS+L_1$ is more stable, and is uniformly better than $SS+L_2$.

We have also tested the Vapnik loss formulation of the stable spline estimator. The fit profile is displayed in Fig. 5.1 (parameters α and ε are constant, set to their cross validated estimates) and is similar to the one obtained by $SS+L_1$. Even though it requires estimation of the additional parameter ε, an advantage of the Vapnik loss over the L_1 is its data compression capability: it detects the so called support vectors which contain those measurements influencing the estimate, see [34] for details.

The last estimator tested embodies the information that the data come from a relaxation system. This means that the impulse response is a completely monotonic function (e.g. see Section 4 in [20]) and, hence, its derivatives $f^{(\ell)}$ satisfy $(-1)^\ell f^{(\ell)}(t) \geq 0$ for $t \geq 0$, $\ell = 0,1,\ldots$. Since $x = Ly$, in discrete-time this information is (approximately) encoded in (4.1) by setting

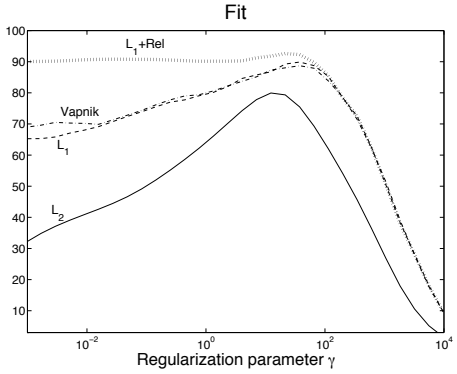


Fig. 5.1: Introductory example. Fit, as a function of the regularization parameter γ, obtained using the stable estimator equipped with the L_2, L_1 and Vapnik loss, and combining the L_1 loss with the information that data come from a relaxation system ($L_1 + Rel$).

a to the null vector and

$$A^T = \begin{bmatrix} -I_n \\ D^1 \\ \dots \\ (-1)^{k-1} D^k \end{bmatrix} L,$$

with k a sufficiently large integer and D an $n \times n$ lower triangular Toeplitz matrix whose first column is $[1,-1,0,\dots]^T$. Let $SS + Rel$ denote the estimator (5.1) complemented with the above constraints ($k = 5$), with the parameter α set to the estimate used by $SS + L_1$. The corresponding fit is reported in Fig. 5.1: $SS + Rel$ shows impressive performance for a wide range of regularization parameter values. Interestingly, the model incorporating the complete monotonicity constraint competes favorably with the overfit residuals even for very low values of γ. This example illustrates the observation that the inclusion of additional model information in the form of constraints often helps to regularize the estimation process while simultaneously improving the fit.

5.2. Computational efficiency: comparison of IPsolve, TFOCS and libSVM. In this section, we compare the computational efficiency of the IPsolve approach with those of two widely used publicly available packages, TFOCS [11] and libSVM [16]. While in fields such as machine learning it is often assumed that the number of unknowns is much larger than the data set size $m \gg n$, in system identification it is typically assumed that $n \gg m$. This makes the per iteration complexity $O(n^2(m+n))$ of our algorithm (see Corollary 4.5) particularly appealing. We illustrate the advantages for the specific problem

$$\arg\min_x \|z - \Phi x\|_\varepsilon + \gamma x^T Q^{-1} x \,, \tag{5.2}$$

where $\|\cdot\|_\varepsilon$ denotes the Vapnik loss (Fig. 3.1f), with noisy data z generated from the impulse response used in the introductory example (Fig. 1.1, top panel). The objective is specified using parameters $\gamma = 10$, $\alpha = 0.5$ in (2.9), and $\varepsilon = 0.1$.

5.2.1. TFOCS. The TFOCS algorithm [11] is based on the proximal point algorithm, and can be applied to generic convex minimization problems. The TFOCS software[3] can

[3] `http://cvxr.com/tfocs`

handle a broader class of problems than IPsolve (i.e. problems that are not piecewise linear quadratic). The relevant standard form for TFOCS is the problem

$$\min \phi(x) = f(x) + \frac{\mu}{2}\|x - x_0\|^2 + h(\mathscr{A}(x) - b), \tag{5.3}$$

where the proximity operator for both f and h can be efficiently computed. TFOCS combines dual smoothing techniques with optimal first-order methods [11, 51, 52] and is therefore capable of solving large-scale problems (much larger than those that can be solved with other general convex solvers, such as the MATLAB package CVX [31, 32].). Like IPsolve, it is a general purpose software that can be used to solve (5.2).

5.2.2. libSVM. libSVM is an optimization package[4] aimed at support vector problems, including problems of type 5.2. It uses pre-compiled routines with several interfaces, including one for Matlab. libSVM is designed for a broad range of support vector problems, including kernel machines; for our problem of interest, the formulation available through libSVM is a slight modification of (5.2):

$$\arg\min_{x,b} \|z - \Phi x - b\mathbf{1}\|_\varepsilon + \gamma x^T Q^{-1} x. \tag{5.4}$$

The inclusion of intercept b is not optional for libSVM; we therefore compared libSVM with IPsolve on (5.4). libSVM solves the *dual* problem to (5.4):

$$\min_{\alpha \in \frac{1}{\gamma}\mathbb{B}_\infty} \frac{1}{2}\alpha^T (Q^{T/2}\Phi^T \Phi Q^{1/2})\alpha + \varepsilon\|\alpha\|_1 + \langle z, \alpha\rangle, \quad \text{s.t.} \quad 1^T\alpha = 0. \tag{5.5}$$

This is a quadratic problem, and libSVM solves it using a specialized decomposition approach. By focusing on the dual, libSVM is able to handle linear and nonlinear SVM and SVR; it has been widely applied in practice.

In the standard system identification context, we have $n << m$; and as shown in Corollary 4.5, the number of arithmetic operations required to implement each iteration is $O(n^2(n+m))$. In contrast, a naive approach for solving (5.5) requires $O(m^3)$ operations. While the approach of [16] is far from naive, it is optimized for kernel machines, where one must restrict all computation to the m-dimensional dual (since the primal dimension may be infinite); in contrast IPsolve exploits the structure of the problem, performing most computations for (5.2) in an n-dimensional space.

5.2.3. Experimental setup and results. We compare all three approaches for problem (5.2) using $n \in \{100, 150, 200, 250, 300\}$ and $m \in \{1000, 2000, 5000, 10000, 20000\}$. The scales of (m,n) are chosen to reflect the common system identification context, where $n << m$. We run each algorithm until their available optimality criteria fall below $\varepsilon = 10^{-6}$; The precise criteria for the three algorithms are as follows.

IPsolve uses a relative magnitude of the Karush-Kuhn-Tucker system; it terminates when $\frac{\|F_{\text{current}}\|}{\|F_{\text{initial}}\|} < \varepsilon$, for $F = F_0$ in (4.8).

TFOCS allows the user to select one of several stopping criterias; some are based on optimality of problem (5.3); but there is also a relative criteria based on iterate convergence, $\frac{\|x_{k+1} - x_k\|}{\max(1, \|x_{k+1}\|)} < \varepsilon$ that allows the algorithm to terminate early. This is the criteria we selected; internal optimality criteria required a far larger number of iterations, during which the objective value did not change significantly. To optimize performance of TFOCS, we experimented with choice of first-order solvers, but found the default algorithm (Auslender & Teboulle's

[4] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

m / n	1K	2K	5K	10K	20K
100	0.376	0.730	1.333	2.666	5.365
150	0.426	0.886	2.782	4.225	7.862
200	0.464	1.149	3.040	4.796	10.105
250	0.464	1.273	3.541	6.176	11.463
300	0.715	1.633	3.966	7.418	13.773

m / n	1K	2K	5K	10K	20K
100	3.320	5.292	5.815	10.896	20.483
150	4.882	4.196	6.800	11.194	26.349
200	3.545	3.985	9.282	10.710	32.644
250	3.840	5.167	8.722	14.787	30.289
300	3.256	4.471	6.787	11.186	17.403

ROD(TFOCS): 10^{-5}

m / n	1K	2K	5K	10K	20K
100	0.0418	0.0046	0.0024	0.0044	0.0022
150	0.0179	0.0080	0.0015	0.0068	0.0260
200	0.0175	0.0135	0.0053	0.0055	0.0399
250	0.0074	0.0055	0.0028	0.0208	0.0100
300	0.1423	0.1424	0.1756	0.0982	0.1748

Table 5.1: Top tables: CPU time (seconds) taken by IPsolve (left) and TFOCS (right) to solve (5.2) as a function of the dimension n of x and of the data set size m. Bottom table: relative (signed) objective difference of the solutions. IPsolve *always* gets the lower objective value; and it is uniformly faster for all problems tested. Note that TFOCS is very accurate.

single-projection method) to be the best. We also tuned the 'restart' option to restart the step-length computation every 1000 iterations; as this improved performance of TFOCS, as recommended by the authors.

libSVM convergence criteria for the SVR problem, as explained in [16], is based on the iterate α satisfying a KKT criteria for (5.5) within ε (in the infinity norm).

In summary, we have two similar optimization problems, (5.2) and (5.4), and three sets of convergence criteria. To fairly compare the algorithms, we run IPsolve against TFOCS on problem (5.2), and IPsolve against libSVM on problem (5.4). In each case, we tabulate both timing results, and also show an 'accuracy' heuristic, which is the signed *relative objective difference* (ROD):

$$\mathrm{ROD}(*) := (f_{(*)} - f_{\mathrm{IPsolve}})/f_{\mathrm{IPsolve}}$$

A positive ROD indicates IPsolve found the lower objective value; a relative scale is chosen because we consider a range of problem sizes.

Results of the numerical study are shown in Tables 5.1 and 5.2. IPsolve gets uniformly better objective values for all experiments, and performs faster that TFOCS for all problem sizes. Notably, TFOCS is very accurate at the settings we compared, with all ROD values less than 10^{-5}.

libSVM is more competitive in its timing, but also less accurate, with some ROD values exceed 10^{-2}, i.e. libSVM primal values are more than 1% larger than those of IPsolve on some of the problems. Overall, IPsolve converges faster for approximately half of the problems; it especially has an advantage for large m and small n as expected. It should be noted that libSVM has strange behavior for the $n = 300$ case; it converges very quickly, but the solutions are less accurate than for other problems.

5.3. Monte Carlo study in the presence of outliers. We now consider a Monte Carlo study of 1000 sample runs. For each run, a random single-input single-output (SISO) continuous time model of order 30 is generated and then sampled at three times its bandwidth using Matlab commands:

m / n	1K	2K	5K	10K	20K
100	0.293	0.629	1.303	2.427	5.011
150	0.421	0.791	2.061	4.197	7.522
200	0.504	1.111	2.791	4.827	9.040
250	0.621	1.148	3.263	6.086	10.460
300	0.719	1.271	3.865	7.314	13.077

m / n	1K	2K	5K	10K	20K
100	0.764	0.343	2.305	8.528	32.944
150	0.124	0.485	2.930	11.628	46.038
200	0.153	0.586	3.691	14.909	59.223
250	0.161	0.667	4.348	17.885	72.330
300	0.052	0.107	0.450	1.515	5.330

rod(libSVM): 10^{-3}

m / n	1K	2K	5K	10K	20K
100	1.8080	0.2988	0.0284	0.1225	0.0037
150	1.4702	0.1488	0.0241	0.0997	0.0049
200	3.5449	0.1984	0.0172	0.1130	0.0004
250	1.5892	0.1723	0.0141	0.1643	0.0080
300	5.4052	21.9701	3.8528	0.4366	1.2052

Table 5.2: Top tables: CPU time (seconds) taken by IPsolve (left) and libSVM (right) to solve (5.4) as a function of the dimension n of x and of the data set size m. Bottom table: relative (signed) objective difference of the solutions. Note that IPsolve *always* gets the lower objective value, and it is faster for the majority of the problems. Note also that libSVM is a lot less accurate than TFOCS (accuracy measured by IPsolve objective).

	Oe+Or	*SS+Or*	*Oe+CV*	*SS+CV*	*SS+ML*
L_2 loss	64.31	68.9	-303.2	59.7	63.1
L_1 loss	79.9	82.3	-73.1	71.8	72.8

Table 5.3: Monte Carlo study (subsection 5.3). Average fit achieved by the PEM and stable spline estimators using the L_2 and the L_1 loss.

```
m=rss(30); b=bandwidth(m); f = b*3*2*pi;
md=c2d(m,1/f,'zoh'); md.d = 0;
```

We accept only models with all poles in a disk of radius 0.95 in the complex plane. The model `md`, initially at rest, is given a Gaussian white noise input with unit variance filtered by a randomly generated model obtained by the same process already described. The input delay is always equal to 1. We then generate 1000 measurements contaminated by outliers, and use them to reconstruct the impulse response. The measurement errors are a mixture of two normals given by $e_i \sim 0.7\mathbf{N}(0,\sigma^2) + 0.3\mathbf{N}(0,100\sigma^2)$, with σ^2 equal to the variance of the noiseless output divided by 100. Thus, with probability 0.3, a measurement becomes an 'outlier', since the corresponding simulated error has standard deviation 10σ.

We compare the performance of five different estimators over the 1000 runs of the simulation. Each estimator uses either quadratic or 1-norm loss for data fidelity, and all formulations treat the system input delay and initial conditions as known. The estimators are enumerated below.

Oe+Or is the classical PEM approach (MATLAB command `oe`), and we compare the quadratic loss (2.5) with the 1-norm loss $V(x) = \sum_{t=1}^{m} |y(t) - G(q,x)u(t)|$. Candidate models are rational transfer functions (2.4) with polynomials B and C of the same order. The estimator is not implementable in practice, since it uses an oracle which selects the model order maximizing the percentage fit measure (2.2), and provides the *best achievable* PEM performance.

Oe+CV is an implementable analogue of *Oe+Or* that uses cross-validation to estimate model

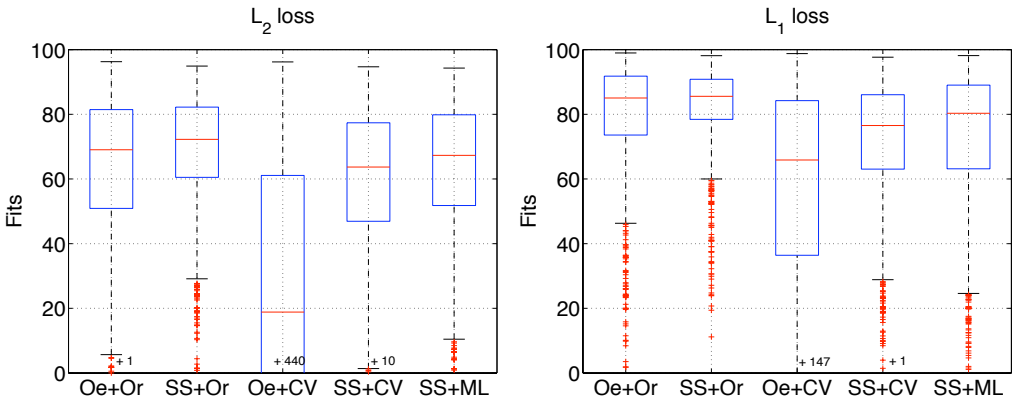


Fig. 5.2: Monte Carlo study (subsection 5.3). Boxplot of the 1000 percentage fit measures (2.2) obtained by PEM and stable spline estimators equipped with the L_2 loss (left panel) and the L_1 loss (right panel).

order. Data are split into training and validation sets of equal size. For every model order ranging from 1 to 30, a model is trained using command `oe` on the training set. We choose the order that minimizes the sum of squared prediction errors on the validation set, obtained by using the command `predict` with null initial conditions. Once the order is found, the final model is computed by `oe` using all measurements in training and validation sets.
SS+Or is the stable spline estimator using (2.9). Again, we compute the fit using both the quadratic loss (2.8) and the 1-norm loss (5.1) for the purpose of comparison. The number of estimated impulse response coefficients is 200, i.e. $\dim(x) = 200$. This estimator also uses an oracle which gives values for the hyperparameters α and γ that maximize the percentage fit measure (2.2). As is the case for *Oe+Or*, this method is not implementable, but provides a baseline for the *best possible* performance of a stable spline estimator.
SS+ML is an implementable analogue of *SS+Or* that uses marginal likelihood maximization to estimate hyperparameters. **For the quadratic loss**, we estimate σ^2 (the noise variance) by fitting a low-bias FIR model of order p, (see e.g. in [30] for details), and then set

$$\hat{\sigma}^2 = \sum_{t=1}^{m} \left(y(t) - \hat{G}(q)u(t)\right)^2 / (m-p), \tag{5.6}$$

where $\hat{G}$ is the pth-order FIR obtained by least-squares. We estimate α and γ by maximizing the *marginal likelihood*

$$(\hat{\lambda}, \hat{\alpha}) = \arg\min_{\lambda,\alpha} \left\{ z^T \Sigma^{-1} z + \log\det(\Sigma) \right\}, \quad \Sigma := \lambda \Phi Q \Phi^T + \hat{\sigma}^2 I_m \,. \tag{5.7}$$

Then let $\hat{Q} = Q(\hat{\alpha})$ and $\hat{\Sigma} = \Sigma(\hat{\lambda}, \hat{\alpha})$ be the estimates of Q and Σ with $\lambda = \hat{\lambda}$ and $\alpha = \hat{\alpha}$. From (2.8), the final impulse response estimate becomes $\hat{x} = \hat{\lambda}\hat{Q}\Phi^T\hat{\Sigma}^{-1}z$. **For the 1-norm loss**, we model components of E in (5.11) as independent Laplacian random variables with variance σ^2. For known hyperparameters, the negative log posterior of $(x|z)$ is

$$\sqrt{\frac{2}{\sigma^2}}\|z - \Phi x\|_1 + \frac{1}{2\lambda} x^T Q^{-1} x$$

with constant terms omitted. Then (5.1) is the MAP estimator of x given z if

$$\gamma = \frac{\sigma^2}{2\sqrt{2}\lambda} \ . \tag{5.8}$$

We take λ and α to be the same estimates as obtained under Gaussian noise assumptions (i.e. optimizing (5.7)), then use (5.8) and (2.9) to obtain γ and Q, respectively.

SS+CV is nearly identical to *SS+ML*, but with hyperparameters estimated by cross-validation. Data are split into a training and validation set of equal size and the best values of γ and α are found over a two dimensional grid. Specifically, the α-γ grid is `A` $\times$ `B` , where A and B are given by the MATLAB commands:

```
A=[0.01 0.05:0.05:0.95 0.99]
B=logspace(log10(g/100),log10(g*100),50)
```

with `g` taken to be the value of γ used in *SS+ML*. The plots in Fig. 5.2 show Matlab box-plots of the 1000 percentage fit measures (2.2) obtained by the five estimators. The rectangle contains the inter-quartile range ($25-75\%$ percentiles) of the fits, with median shown with a red line. The "whiskers" outside the rectangle display the upper and lower bounds of all the numbers, not counting what are deemed outliers, plotted separately as "+". Table 5.3 also reports the average percentage fit values.
The left panel of Fig. 5.2 shows the fits achieved by the quadratic estimators. Oracle-based procedures highlight the advantage of the stable spline estimators: *SS+Or* shows better performance than *Oe+Or*. The performance gap increases in implementable estimators, when hyperparameters are learned from data (as necessary in practical situations). *SS+CV* and *SS+ML* have similar performance, and both are much better than *Oe+CV*.
The right panel of Fig. 5.2 displays the fits achieved by all five estimators when using the 1-norm data fidelity loss function. These estimators are more robust against outliers, and all fits improve significantly. Furthermore, as in the previous case, performance of stable spline estimators is superior to that of the classical system identification procedures.

5.4. Assessment of cerebral hemodynamics using magnetic resonance imaging. The quantitative assessment of the cerebral blood flow is essential to the understanding of brain function. An important technique is bolus-tracking magnetic resonance imaging (MRI), which relies on established principles for tracer kinetics of nondiffusible tracers [70]. These principles allow for the quantification of cerebral hemodynamics by solving a linear system identification problem. The system output is the measured tracer concentration within a given tissue volume of interest, while the system input is the measured arterial function. The impulse response is proportional to the so called *tissue residue function*, and is known to be positive and unimodal. It carries fundamental information on the system under study, e.g. the cerebral blood flow is given by its maximum value. However, impulse response estimation is especially difficult for this problem: even if the noise can be reasonably modeled as Gaussian, the problem is often ill-conditioned and only a few noisy output samples are available [69].
We consider realistic simulation studies, using four different types of estimators all based on the quadratic loss. The first two rely on the classical PEM paradigm. They are *Oe+Or*, described in the previous subsection, with the maximum allowed model order equal to 10, and *Oe+AICc*, which uses the AICc criterion [37] for model complexity selection. *SS+ML* uses the stable spline kernel (2.10), estimates hyperparameters via marginal likelihood optimization, and then uses (2.8) to find the final impulse response, where $x \in \mathbb{R}^{100}$. The last estimator *SS+ML+um* also estimates hyperparameters via marginal likelihood optimization, and then incorporates nonnegativity and unimodality information. Specifically, we minimize objective (2.8) (with hyperparameter estimates identical to those used by *SS+ML*), subject to *inequality*

constraints that impose unimodality:

$$D_1^k x_1^k \geq 0, \quad D_{k+1}^n x_{k+1}^n \leq 0, \quad x \geq 0\,. \tag{5.9}$$

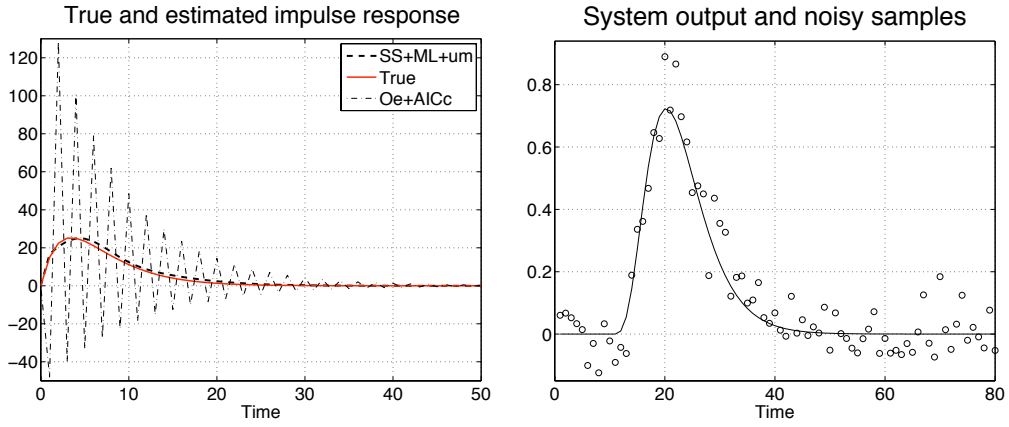


Fig. 5.3: Cerebral hemodynamics. *(Left)*: true impulse response (solid), PEM estimate with AICc to select the model order (dashdot) and the stable spline estimate incorporating unimodality constraints (dashed). *(Right)*: Noiseless output (solid) and the measurements (○).

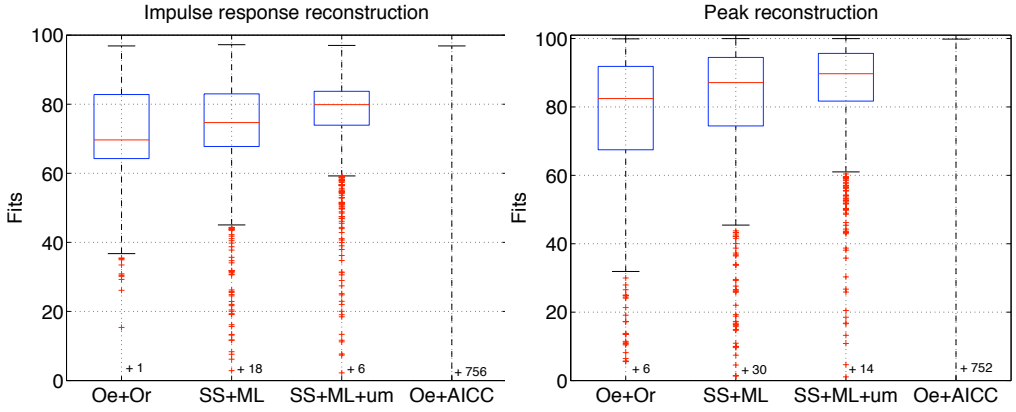


Fig. 5.4: Assessment of cerebral hemodynamics. Boxplot of the 1000 percentage fit measures (2.2) for the impulse response reconstruction (left) and peak reconstruction (right) obtained by PEM equipped with an oracle and with AICc for model order selection, by the stable spline estimator and by the stable spline estimator incorporating unimodality constraints.

Here, $D \in \mathbb{R}^{n\times n}$ is the discrete derivative operator, i.e. a lower triangular Toeplitz matrix with first column $[1,-1,0,\ldots]^T$, D_1^k and D_{k+1}^n contain, respectively, the first k and the last $n-k$ rows of D, and analogously for x_1^k and x_{k+1}^n. In terms of (4.1), this is specified setting a

to the null vector and

$$A = L^T \begin{bmatrix} -(D_1^k)^T & (D_{k+1}^n)^T & -I_n \end{bmatrix}. \tag{5.10}$$

We solve the problem for each k, obtaining a set of solutions $\hat{x}(k)$, and then select the best $\bar{k}$ for (2.8) by setting $\bar{k} = \arg\min_k \|z - \Phi\hat{x}(k)\|^2 + \gamma(\hat{x}(k))^T Q^{-1}\hat{x}(k)$, with the final estimator given by $\hat{x}(\bar{k})$, the best unimodal estimator. We begin by considering the simulation described in [69]. The system input is the typical arterial function $u(t) = (t-10)^3 e^{-\frac{2t}{3}}$ if $t > 10$ and zero otherwise, while the impulse response is the dispersed exponential displayed in Fig. 5.3 (left panel, solid line). This response is to be reconstructed from the 80 noisy output samples reported in Fig. 5.3 (right panel). These measurements are generated as in subsection II.A of [69], using parameters typical of a normal subject with a signal to noise ratio equal to 20 and discretizing the problem using unit sampling instants.
The left panel of Fig. 5.3 shows the estimate by *Oe+AICc* (dashdot). The reconstructed profile is far from the truth and contains many non-physiological oscillations: the asymptotic theory underlying AICc does not compensate for ill-conditioning. The same panel shows the *SS+ML+um* estimate (dashed line). This estimate is very close to truth, outlining the importance of regularization and the inclusion of addition information in the form of constraints when handling these data poor situations. This result is confirmed by a Monte Carlo study of 1000 runs where independent noise realizations are generated at every run. Fig. 5.4 shows the Matlab boxplots of the 1000 percentage fit measures (2.2) achieved by the four estimators in the reconstruction of the impulse response (left panel) and of its peak (right panel). Most of the time *Oe+AICc* returns negative fits, while the outcomes from *Oe+Or* and *SS+ML* are similar with good performance (keep in mind *Oe+Or* is not implementable in practice). Finally, notice that *SS+ML+um* has the best performance vis-à-vis the percentage fit measures (2.2), see also Table 5.4 for the average fits.

	Oe+Or	*SS+ML*	*SS+ML+um*	*Oe+AICc*
Imp. response	71.9	69.8	76.3	-3497.1
Peak	77.1	76.2	84.3	-26.3

Table 5.4: Average percentage fit (2.2) achieved by the PEM and stable spline estimators relative to impulse response and peak reconstruction.

5.5. Stable and sparse estimation. A multiple input single output (MISO) system can be written as

$$z = \sum_{j=1}^{p} \Phi^j x^j + E, \tag{5.11}$$

where each vector x^j contains the coefficients of the j-th impulse response. This problem is challenging since the number of unknowns can be much larger than the number of output measurements. One strategy is to use sparse regularization on x^j to simultaneously perform variable selection and estimation. This problem arises in dynamic networks, where sparsity is used to detect structural connectivity.
To design a sparsity promoting stable PLQ estimator, take Q be the stable spline kernel in

(2.9) and let let $y^j = L^{-1}x^j,\ Q = LL^T$. Then solve the problem

$$\min_y \left\| z - \sum_{j=1}^{p} \Phi^j L y^j \right\|^2 + \sum_{j=1}^{p} \gamma \|y^j\|_1 \,. \tag{5.12}$$

5.5.1. A Monte Carlo study. A simple numerical study is used to test (5.12). Output data are generated as described in Section 5.3, but no outliers are introduced. In the MISO, x^1 is the impulse response plotted in the top panels of Fig. 1.1 while, for $j = 2,\ldots,10$, the remaining x^j are null. Each x^j contains 100 impulse response coefficients. Thus, the estimator seeks to reconstruct 1000 unknowns from 400 measurements. The regularization parameter γ and the kernel parameter α are obtained using the cross validation strategy described in the Section 5.3. At each Monte Carlo run we compute the fit (2.2) related just to x^1. We also compute the fit obtained by

$$\min_y \left\| z - \Phi^1 L y^1 \right\|^2 + \gamma \|y^1\|^2 \,, \tag{5.13}$$

which corresponds to an oracle classical estimator that knows only x^1 may be different from zero. After 100 runs the average fits of (5.12) and (5.13) were 91.8 and 93.4, that is, there was little loss in fit relative to an oracle estimator.

5.5.2. A comparison with FISTA. The problem (5.12) is the sum of a smooth and simple function, and can therefore be optimized by primal-only first-order methods. The Fast Iterative Shrinkage Algorithm (FISTA) [10] is an *optimal* first-order method, i.e. it can obtain a guaranteed rate of convergence of $O(\frac{1}{k^2})$, matching the best complexity of first-order methods for convex problems (with faster rates under stronger assumptions). However, the rate of convergence also depends on the condition number of the problem.
To compare IPsolve with this optimal first-order method in the high-dimensional system identification setting, we used the MISO problem to generate three scenarios: 400 measurements vs. $1000, 5000$, and 10000 unknown impulse responses. We solved (5.12) using IPsolve, and then ran FISTA to see how long it would take it to obtain the same function value.
The condition number of L is 3.5×10^5 in each experiment. The condition numbers of $A = \Phi LL^T\Phi^T$ are given in the table. When $n >> m$, IPsolve uses the Woodbury-Sherman-

	IP time	*IP iters*	*FISTA time*	*FISTA iters*	*Cond A*
400 x 1000	0.89	15	1.8	635	3.0×10^7
400 x 5000	1.96	16	20	1204	1.27×10^4
400 x 10000	3.8	18	49.8	1458	2.83×10^3

Table 5.5: Numerical comparison of IPsolve with FISTA. For each problem, FISTA is run until it hits the objective function achieved by IPsolve.

Morisson formula to form and invert an $m \times m$ matrix at each iteration. Therefore the dominant costs of each iteration are $O(m^2(m+n))$, linear in n. Each iteration of FISTA is dominated by the computation of the gradient, which is $2mn$. On the other hand, the number of iterations required by FISTA is far less predictable than iterations required by IPsolve. The key difference between IPsolve and first-order methods are that its complexity per iteration is

cubic in $\min(m,n)$, while that of FISTA is quadratic. However, in regimes where $m << n$ or $n >> m$, and the systems may be ill-conditioned, IPsolve is competitive. Finally, the reader should keep in mind that (5.12) has a special form, and more general PLQ estimators cannot be solved with primal-only methods such as FISTA, but can still be solved using IPsolve.

6. Conclusions. This paper extends stable spline estimators to allow general modeling of misfit measures, regularizers, and constraints. Quadratic losses and regularizers can now be replaced by general PLQ functions. Furthermore, affine inequality constraints on the unknown impulse response can also be incorporated, providing a simple mechanism for the inclusion of information on domain restriction, monotonicity, and unimodality of the signal. This can have a profound impact on the quality of the recovery and can significantly improve the fit when a regularizing prior is required for identification (as illustrated in Fig. 5.1).
The new framework allows the user to formulate and explore new system identification procedures, balancing robustness against outliers, introducing sparsity promoting priors, or introducing additional information by means of affine inequality constraints. If the resulting nonparametric model is to be used for control purposes, our estimates can be projected onto suitable low-order approximations using e.g. the approaches described in [12].
All of the extensions have been implemented in the open source package IPsolve. Numerical comparisons in Section 5.2 showed that IPsolve is well-suited to ill-conditioned problems that arise in system identification, and outperforms competing alternatives, including TFOCS and libSVM when $n << m$ and first-order methods such as FISTA when $n >> m$.
Multiple examples illustrate the power of the new framework in comparison to classical approaches. An important example comprises impulse responses known to be positively or completely monotonic, in the presence of outliers. Classic approaches (using PEM and rational transfer function models (2.4)) solve a non-convex, possibly non-differentiable high-dimensional inequality constrained optimization problem ($\dim(x)$ specifies domain complexity) *for each postulated model structure*. Model order selection, a delicate issue even in the unconstrained case, becomes, *a fortiori*, even harder. These drawbacks have greatly limited the use of algorithms for inequality constrained/non-smooth linear system identification.
In contrast, we consider only a low-dimensional hyperparameter vector (λ, α), and use cross-validation over a grid of hyperparameter values, solving an inequality constrained PLQ optimization problem for each choice of parameters. The model selection process is intuitive, and the entire approach is efficiently implementable, since the number of arithmetic operations required for the evaluation of each choice of hyperparameters is proportional to that of standard RLS approaches. Spline kernel modeling with PLQ optimization pave the way to for new applications of robust, sparse, and inequality constrained linear system identification.
Acknowledgements The authors are grateful to Dr. Stephen Becker, for help using TFOCS. Dr. Aravkin's research has been funded by the WRF Data Science Professorship.

REFERENCES

[1] A. Aravkin, P. Kambadur, A. Lozano, and R. Luss. Orthogonal matching pursuit for sparse quantile regression. In *Data Mining (ICDM), International Conference on*, pages 11–19. IEEE, 2014.
[2] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.
[3] M. Antinescu, G. Lesaja, and F. Potra. Equivalence between different formulations of the linear complementary problem. *Optimization Methods and Software*, 7:265–290, 1997.
[4] A. Aravkin, B. M. Bell, J. V. Burke, and G. Pillonetto. An ℓ_1-Laplace robust Kalman smoother. *IEEE Transactions on Automatic Control*, 56(12):2898–2911, 2011.
[5] A. Aravkin, B. M. Bell, J. V. Burke, and G. Pillonetto. Learning Using State Space Kernel Machines. In *Proc. IFAC World Congress 2011*, Milan, Italy, 2011.
[6] A. Aravkin, J. Burke, A. Chiuso, and G. Pillonetto. Convex vs non-convex estimators for regression and

sparse estimation: the mean squared error properties of ard and glasso. *Journal of Machine Learning Research*, 15:217–252, 2014.

[7] A. Aravkin, J. Burke, and G. Pillonetto. A statistical and computational theory for robust and sparse Kalman smoothing. In *Proceedings of the 16th IFAC Symposium on System Identification (SysId 2012)*, 2012.

[8] A. Aravkin, J. V. Burke, and G. Pillonetto. Linear system identification using stable spline kernels and plq penalties. *To appear in IEEE Conf. Decision and Control (CDC), March 2013*, 2013.

[9] A. Aravkin, J. V. Burke, and G. Pillonetto. Sparse/robust estimation and kalman smoothing with nonsmooth log-concave densities: Modeling, computation, and theory. *Journal of Machine Learning Research*, 14:2689–2728, 2013.

[10] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal of Imaging Sciences*, 2(1):183–202, 2009.

[11] S. Becker, E. Candes, and M. Grant. Templates for convex cone problems with applications to sparse signal recovery. *Mathematical Programming Computation*, 3(3):165–218, 2011.

[12] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015.

[13] J. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Series in Statistics. Springer, second edition, 1985.

[14] M. Bertero. Linear inverse and ill-posed problems. *Advances in Electronics and Electron Physics*, 75:1–120, 1989.

[15] F. P. Carli. On the maximum entropy property of the first-order stable spline kernel and its implications. In *Control Applications (CCA), 2014 IEEE Conference on*, pages 409–414. IEEE, 2014.

[16] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[17] T. Chen, H. Ohlsson, and L. Ljung. On the estimation of transfer functions, regularizations and Gaussian processes - revisited. *Automatica*, 48(8):1525–1535, 2012.

[18] A. Chiuso and G. Pillonetto. A bayesian approach to sparse dynamic network identification. *Automatica*, 48(8):1553 – 1565, 2012.

[19] W. Chu, S. S. Keerthi, and C. J. Ong. A unified loss function in bayesian framework for support vector regression. *Epsilon*, 1(1.5):2, 2001.

[20] F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2001.

[21] C. De Mol, E. De Vito, and L. Rosasco. Elastic-net regularization in learning theory. *Journal of Complexity*, 25(2):201–230, 2009.

[22] O. Dekel, S. Shalev-Shwartz, and Y. Singer. Smooth epsiloon-insensitive regression by loss symmetrization. In *Journal of Machine Learning Research*, pages 711–741, 2005.

[23] D. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289–1306, 2006.

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

[25] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–150, 2000.

[26] S. Farahmand, G. B. Giannakis, and D. Angelosante. Doubly Robust Smoothing of Dynamical Processes via Outlier Sparsity Constraints. *IEEE Transactions on Signal Processing*, 59:4529–4543, 2011.

[27] D. A. Freedman. *Statistical models: theory and practice*. cambridge university press, 2009.

[28] J. Gao. Robust l1 principal component analysis and its {B}ayesian variational inference. *Neural Computation*, 20(2):555–572, Feb. 2008.

[29] F. Girosi. Models of noise and robust estimates. A.I. Memo 1287, Artificial Intelligence Laboratory, 1287, Massachusetts Institute of Technology, 1991.

[30] G. Goodwin, M. Gevers, and B. Ninness. Quantifying the error in estimated transfer functions with application to model order selection. *IEEE Transactions on Automatic Control*, 37(7):913–928, 1992.

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

[32] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, Mar. 2014.

[33] O. Güler. Generalized linear complementarity problems. *Mathematics of Operations Research*, 20:441–448, 1995.

[34] T. Hastie and R. Tibshirani. Generalized additive models. In *Monographs on Statistics and Applied Probability*, volume 43. Chapman and Hall, London, UK, 1990.

[35] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning. Data Mining, Inference and Prediction*. Springer, Canada, 2001.

[36] P. J. Huber. *Robust Statistics*. John Wiley and Sons, 2004.

[37] C. Hurvich and C. Tsai. Regression and time series model selection in small samples. *Biometrika*, 76:297–307, 1989.

[38] R. Koenker. *Quantile Regression*. Cambridge University Press, 2005.

[39] R. Koenker and G. Bassett. Regression quantiles. *Econometrica*, pages 33–50, 1978.

[40] R. Koenker and O. Geling. Reappraising medfly longevity: A quantile regression survival analysis. *Journal of the American Statistical Association*, 96:458468, 2001.

[41] M. Kojima, N. Megiddo, T. Noma, and A. Yoshise. *A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems*, volume 538 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, Germany, 1991.

[42] Y.-J. Lee, W.-F. Hsieh, and C.-M. Huang. ε-ssvr: a smooth support vector machine for ε;-insensitive regression. *Knowledge and Data Engineering, IEEE Transactions on*, 17(5):678–685, 2005.

[43] Q. Li, N. Lin, et al. The bayesian elastic net. *Bayesian Analysis*, 5(1):151–170, 2010.

[44] W. Li and J. Swetits. The linear l1 estimator and the huber m-estimator. *SIAM Journal on Optimization*, 8(2):457–475, 1998.

[45] L. Ljung. *System Identification, Theory for the User*. Prentice Hall, 1999.

[46] D. MacKay. Bayesian interpolation. *Neural Computation*, 4:415–447, 1992.

[47] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, 17(1):53–69, Jan. 2008.

[48] J. S. Maritz and T. Lwin. *Empirical Bayes Method*. Chapman and Hall, 1989.

[49] R. A. Maronna, D. Martin, and Yohai. *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley, 2006.

[50] A. Nemirovskii and Y. Nesterov. *Interior-Point Polynomial Algorithms in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, USA, 1994.

[51] Y. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Dokl. Akad. Nauk SSSR*, 269(3):543–547, 1983.

[52] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

[53] G. Pillonetto, A. Chiuso, and G. De Nicolao. Regularized estimation of sums of exponentials in spaces generated by stable spline kernels. In *Proceedings of the IEEE American Cont. Conf., Baltimora, USA*, 2010.

[54] G. Pillonetto, A. Chiuso, and G. D. Nicolao. Prediction error identification of linear systems: a nonparametric Gaussian regression approach. *Automatica*, 47(2):291–305, 2011.

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

[56] G. Pillonetto and G. De Nicolao. Pitfalls of the parametric approaches exploiting cross-validation or model order selection. In *Proceedings of the 16th IFAC Symposium on System Identification (SysId 2012)*, 2012.

[57] G. Pillonetto, F. Dinuzzo, T. Chen, G. D. Nicolao, and L. Ljung. Kernel methods in system identification, machine learning and function estimation: a survey. *Automatica*, March, 2014.

[58] M. Pontil and A. Verri. Properties of support vector machines. *Neural Computation*, 10:955–974, 1998.

[59] C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[60] B. Schölkopf, A. Smola, R. Williamson, and P. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.

[61] G. Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[62] G. Seber and C. Wild. *Nonlinear Regression*. Wiley Series in Probability and Statistics. Wiley, 2003.

[63] T. Söderström and P. Stoica. *System Identification*. Prentice-Hall, 1989.

[64] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society, Series B.*, 58(1):267–288, 1996.

[65] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, USA, 1998.

[66] 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.

[67] S. Wright. A path-following interior point algorithm for linear and quadratic problems. *Annals of Operations Research*, 62:103–130, 1996.

[68] S. J. Wright. *Primal-Dual Interior-Point Methods*. Siam, Englewood Cliffs, N.J., USA, 1997.

[69] F. Zanderigo, A. Bertoldo, G. Pillonetto, and C. Cobelli. Nonlinear stochastic regularization to characterize tissue residue function in bolus-tracking mri: Assessment and comparison with svd, block-circulant svd, and tikhonov. *IEEE Transactions on Biomedical Engineering*, 56(5):1287–1297, 2009.

[70] K. Zierler. Equations for measuring blood flow by external monitoring of radioisotopes. *Circ. Res.*, 16:309–321, 1965.

[71] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[72] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.