SIAM J. SCI. COMPUT.
Vol. 33, No. 6, pp. 3132–3152

DESIGNING OPTIMAL SPECTRAL FILTERS FOR INVERSE PROBLEMS*

JULIANNE CHUNG†, MATTHIAS CHUNG‡, AND DIANNE P. O'LEARY§

Abstract. Spectral filtering suppresses the amplification of errors when computing solutions to ill-posed inverse problems; however, selecting good regularization parameters is often expensive. In many applications, data are available from calibration experiments. In this paper, we describe how to use such data to precompute optimal spectral filters. We formulate the problem in an empirical Bayes risk minimization framework and use efficient methods from stochastic and numerical optimization to compute optimal filters. Our formulation of the optimal filter problem is general enough to use a variety of assessments of goodness of the solution estimate, not just the mean square error. The relationship with the Wiener filter is discussed, and numerical examples from signal and image deconvolution illustrate that our proposed filters perform consistently better than well-established filtering methods. Furthermore, we show how our approach leads to easily computed uncertainty estimates for the pixel values.

Key words. ill-posed problem, filtering, regularization, singular value decomposition, stochastic programming, optimal design, optimal filtering, machine learning, image deblurring, Tikhonov, Wiener filter, Bayes risk, Bayesian risk, empirical risk

AMS subject classifications. 65F20, 65F22, 62C12

DOI. 10.1137/100812938

1. Introduction.

Ill-posed inverse problems arise in many applications, such as signal and image processing. Approximate solutions are typically computed using a *regularization method* that solves a nearby well-posed problem. Filters for spectral regularization, such as truncated singular value decomposition (SVD) and Tikhonov filtering, and various parameter selection methods, such as discrepancy principle and generalized cross-validation (GCV), have been proposed and studied [16]. Oftentimes, parameter selection is a most critical and expensive task.

Consider the commonly occurring situation where a signal is measured by some instrument such as a charge-coupled device camera or a magnetic resonance imaging detector. The goal is to reconstruct the signal, and reconstruction may need to be done very quickly, e.g., in real time. Expensive parameter choice methods may not be possible. However, it may be possible to obtain information from a series of calibration experiments, where known signals are used as input.

In this paper, we address the problem of how to use such calibration data in order to choose optimal spectral filters for the inverse problem. We reformulate the problem as an empirical Bayes risk minimization problem [3, 32] and consider efficient approaches for computing optimal filters. For example, one could minimize the mean squared error, the sum of absolute errors, or the maximum error (corresponding to

*Submitted to the journal's Computational Methods in Science and Engineering section October 27, 2010; accepted for publication (in revised form) July 20, 2011; published electronically November 1, 2011.

http://www.siam.org/journals/sisc/33-6/81293.html

†Department of Computer Science, University of Maryland, College Park, MD 20742 (jmchung@cs.umd.edu). This author's research was partially supported by NSF grant DMS 0902322.

‡Department of Mathematics, Texas State University – San Marcos, TX 78666 (mc85@txstate.edu).

§Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (oleary@cs.umd.edu). This author's research was partially supported by NSF grant DMS 1016266.

the 2-norm, 1-norm, and ∞-norm, respectively). Furthermore, optimal filters can be computed off-line, so that reconstructions can be done quickly.

Most previous work on selecting optimal or near-optimal regularization parameters (e.g., [25, 33, 9, 12]) has been restricted to Tikhonov regularization, and these methods are based on minimizing the predictive mean squared error. The Wiener filter, introduced in the 1940s, is a well-studied filter which seeks to minimize the expected value of the mean squared error [34]. Computing the Wiener filter requires additional information (i.e., the first two moments), and a variety of researchers have proposed methods to approximate the Wiener filter [1]. One of our proposed approaches for computing optimal filters provides a closed form approximation to the Wiener filter.

However, the mean squared error is not always the best choice for assessing goodness of the estimated solution. One novelty of our approach is the use of different choices, such as the Huber function and various p-norms, to assess goodness. These extensions provide practical solutions to problems which are often difficult to solve by other methods. Numerical results illustrate the benefits of using these different choices.

The paper is outlined as follows. Section 2 introduces the basic problem of computing an optimal filter within the framework of empirical Bayes risk minimization. Algorithms for computing the optimal filter are then discussed in section 3, where specific examples of filter representations and error assessments are presented. In section 4, we address implementation concerns such as computational stability and large-scale implementation. Then, numerical results are presented for 1D and 2D examples, demonstrating the performance of the optimal filters and comparing them to standard approaches. We also show how uncertainty estimates for pixel values can be easily obtained. Discussion and future work are presented in section 5.

2. Spectral filtering for ill-posed problems. In this section, we briefly provide some background on the properties of the underlying problem. Then we describe the problem of computing optimal filters for spectral regularization and formulate it as a empirical risk minimization problem.

Many scientific problems can be modeled as

$$\mathbf{b} = \mathbf{A}\boldsymbol{\xi} + \boldsymbol{\delta}, \tag{2.1}$$

where $\boldsymbol{\xi} \in \mathbb{R}^n$ is the desired solution, $\mathbf{A} \in \mathbb{R}^{n\times n}$ models the forward process, $\mathbf{b} \in \mathbb{R}^n$ represents the observed data, and $\boldsymbol{\delta} \in \mathbb{R}^n$ is additive random noise [16, 30]. The goal is to obtain an approximation of $\boldsymbol{\xi}$ given $\mathbf{b}, \mathbf{A}$, and prior knowledge of the probability distribution of $\boldsymbol{\delta}$. For this paper, we consider $n \times n$ problems, but we note that extension to the rectangular case is straightforward. Furthermore, we assume that the size and/or structure of matrix $\mathbf{A}$ is such that computing the SVD is computationally feasible.

In particular, let $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$, where $\boldsymbol{\Sigma}$ is a diagonal matrix containing the singular values, $\sigma_i, i = 1, \ldots, n$, and $\mathbf{U}$ and $\mathbf{V}$ contain the left and right singular vectors, $\mathbf{u}_i$ and $\mathbf{v}_i$, respectively. The singular values σ_i, $i = 1, \ldots, n$, are sorted, $\sigma_i \geq \sigma_\ell$ for $i < \ell$. Some important characteristics of discrete ill-posed inverse problems are that the singular values decay to zero and the singular vectors corresponding to small singular values exhibit highly oscillatory behavior. We assume that the sequence $|\mathbf{u}_i^\top \mathbf{b}|$ decreases to 0 faster than the sequence σ_i, implying that the problem satisfies the discrete Picard condition [16]. It is a well-known property of ill-posed inverse

problems that the naive inverse solution,

$$\mathbf{x}_{\text{inv}} = \sum_{i=1}^{n} \frac{\mathbf{u}_i^\top \mathbf{b}}{\sigma_i} \mathbf{v}_i \,,$$

will contain large errors and may not be a good approximation of $\boldsymbol{\xi}$.

Spectral filtering is one approach to stabilize or regularize the solution. The filtered solution can be written as

$$\begin{aligned} \mathbf{x}_{\text{filter}} &= \sum_{i=1}^{n} \phi_i \frac{\mathbf{u}_i^\top \mathbf{b}}{\sigma_i} \mathbf{v}_i \\ &= \mathbf{V}\mathbf{C}\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi} \,, \end{aligned}$$

where $\boldsymbol{\phi}$ is a vector of filter factors ϕ_i, $\mathbf{c} = \mathbf{U}^\top \mathbf{b}$, and $\mathbf{C} = \text{diag}(\mathbf{c})$. Oftentimes, the filter factors corresponding to larger singular values are chosen to be close to 1, while those corresponding to smaller singular values are close to 0.

The choice of the filter factors $\phi_i,\ i = 1, \ldots, n$, is crucial to the accuracy of the solution estimate. In this paper we assume that the filter factors can be represented or parameterized as $\boldsymbol{\phi}(\boldsymbol{\alpha})$, where $\boldsymbol{\phi} : \mathcal{A} \to \mathbb{R}^n$ and $\mathcal{A}$ is a nonempty closed subset of $\mathbb{R}^n$. We are interested in computing optimal filter factors to minimize the error between the filtered reconstruction and the true solution. The error vector is defined as

$$\mathbf{e}(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta}) = \mathbf{x}_{\text{filter}}(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta}) - \boldsymbol{\xi} \,,$$

with entries $e_i(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta})$, and the error is defined as

$$\text{ERR}(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta}) = \frac{1}{n} \sum_{i=1}^{n} \rho(e_i(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta})) \,, \tag{2.2}$$

where $\rho(z)$ is a nondecreasing function of $|z|$. For example, one could use

$$\rho(z) = \frac{1}{p}|z|^p \,, \tag{2.3}$$

for $p \geq 1$, in which case $\text{ERR} = \frac{1}{np}\|\mathbf{e}\|_p^p$, where $\|\mathbf{e}\|_p$ is the p-norm of vector $\mathbf{e}$. Other functions such as the Huber function can also be used and will be discussed in section 3.2.

In addition to the choice of $\rho(z)$, the optimal filter will depend on a variety of factors, including the operator $\mathbf{A}$, the noise in the observation $\mathbf{b}$, and the true realization $\boldsymbol{\xi}$. We make the following assumptions:

1. The matrix $\mathbf{A}$ is a deterministic operator; i.e., $\mathbf{A}$ is not subject to disturbances or noise.
2. The true realization was chosen following a given probability distribution $\mathcal{P}_{\boldsymbol{\xi}}$ on $\boldsymbol{\Xi} \subset \mathbb{R}^n$ that has finite second moments.
3. A probability distribution $\mathcal{P}_{\boldsymbol{\delta}}$ on $\boldsymbol{\Delta} \subset \mathbb{R}^n$ for the noise in the observation $\mathbf{b}$ is provided and has zero mean and finite second moments.

Ideally, an optimal filter would minimize the expected value of the errors with respect to the joint distribution of $\boldsymbol{\xi}$ and $\boldsymbol{\delta}$. Hence, the problem of finding an optimal spectral filter $\check{\boldsymbol{\phi}} = \boldsymbol{\phi}(\check{\boldsymbol{\alpha}})$ could be formulated as

$$\check{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) = \mathbf{E}_{\boldsymbol{\delta},\boldsymbol{\xi}}\Big\{\text{ERR}(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta})\Big\}, \tag{2.4}$$

where $f(\boldsymbol{\alpha})$ is the *Bayes risk* and (2.4) is the *Bayes risk minimization problem*. Solving optimization problem (2.4) provides the parameters $\check{\boldsymbol{\alpha}}$ that determine the optimal filter $\check{\boldsymbol{\phi}}$.

The Bayes risk minimization formulation allows us to incorporate probabilistic information in the form of probability distributions in order to design optimal filters. It is important to note that problem (2.4) can also be interpreted as a stochastic programming problem, and we refer the interested reader to books [31, 29, 27, 32], survey papers [8, 30], other contributions [21, 23, 18, 14], and references therein. In the next section, we discuss some approaches to approximate problem (2.4), and we describe numerical methods for optimization in this framework. In addition, a variety of functional representations, $\boldsymbol{\phi}(\boldsymbol{\alpha})$, and choices of $\rho(z)$ will be discussed.

3. Computing optimal spectral filters. Solving optimization problems like (2.4) can be difficult because the expected value $f(\boldsymbol{\alpha})$ (a multidimensional integral) cannot be explicitly computed with high accuracy. Monte Carlo-based sampling algorithms have been proposed as an efficient way to approximate solutions to stochastic programming problems. In general, two computational approaches are commonly used, *stochastic approximation* and *sample average approximation* [22]. The stochastic approximation method is an iterative minimization approach where a new sample from the distribution is used at each iteration to update the solution. Although low memory usage is one of the advantages, proper choice of step size and slow convergence are potential drawbacks. We do not consider this method in this paper.

On the other hand, the sample average approximation approach generates a set of N random samples $\boldsymbol{\xi}^{(1)}, \ldots, \boldsymbol{\xi}^{(N)} \in \boldsymbol{\Xi}$ and $\boldsymbol{\delta}^{(1)}, \ldots, \boldsymbol{\delta}^{(N)} \in \boldsymbol{\Delta}$ and approximates problem (2.4) with a sample average problem. More specifically, let $\mathbf{b}^{(k)} = \mathbf{A}\boldsymbol{\xi}^{(k)} + \boldsymbol{\delta}^{(k)}$, $\mathbf{c}^{(k)} = \mathbf{U}^{\top}\mathbf{b}^{(k)}$, and $\mathbf{C}^{(k)} = \operatorname{diag}(\mathbf{c}^{(k)})$, and define error vectors $\mathbf{e}^{(k)}(\boldsymbol{\alpha}) = \mathbf{x}_{\text{filter}}(\boldsymbol{\alpha}, \boldsymbol{\xi}^{(k)}, \boldsymbol{\delta}^{(k)}) - \boldsymbol{\xi}^{(k)}$, where $\mathbf{x}_{\text{filter}}(\boldsymbol{\alpha}, \boldsymbol{\xi}^{(k)}, \boldsymbol{\delta}^{(k)}) = \mathbf{V}\mathbf{C}^{(k)}\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}(\boldsymbol{\alpha})$. Then stochastic programming problem (2.4) is approximated with the sample average problem

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \; f_N(\alpha), \tag{3.1}$$

where

$$f_N(\boldsymbol{\alpha}) = \frac{1}{nN} \sum_{k=1}^{N} \sum_{i=1}^{n} \rho(e_i^{(k)}(\boldsymbol{\alpha})) \tag{3.2}$$

is the *empirical risk*. Hence, (3.1) is also known as the *empirical (Bayes) risk minimization problem*. Numerical optimization methods can be used to solve (3.1), and the desired optimal spectral filter is computed as $\hat{\boldsymbol{\phi}} = \boldsymbol{\phi}(\hat{\boldsymbol{\alpha}})$. Before we describe numerical methods, we provide some remarks on the connections between the Bayes risk minimization problem (2.4) and the empirical risk minimization problem (3.1).

The samples, $\boldsymbol{\xi}^{(k)}$, and noise realizations, $\boldsymbol{\delta}^{(k)}$, for $k = 1, \ldots, N$, constitute a *training set*. By the law of large numbers, $f_N(\boldsymbol{\alpha})$ converges pointwise with probability one to $f(\boldsymbol{\alpha})$ as $N \to \infty$. With the additional assumption that $f_N(\boldsymbol{\alpha})$ converges uniformly to $f(\boldsymbol{\alpha})$, it can be shown that $f_N(\hat{\boldsymbol{\alpha}})$ converges with probability one to $f(\check{\boldsymbol{\alpha}})$, where $\hat{\boldsymbol{\alpha}}$ and $\check{\boldsymbol{\alpha}}$ are the solutions of (3.1) and (2.4), respectively [27]. This is typically referred to as consistency of the empirical risk estimator of the optimal value. It is far more challenging to prove consistency of the empirical risk estimator of optimal solutions, that is, that $\hat{\boldsymbol{\alpha}} \to \check{\boldsymbol{\alpha}}$ as $N \to \infty$. Conditions for consistency of solutions are stated, e.g., in Theorem 5.3 in Shapiro, Dentcheva, and Ruszczynski [27].

In this paper, we provide a numerical investigation of the empirical risk minimization problem and point the interested reader to a comprehensive overview of convergence results for statistical learning theory which can be found in a book by Vapnik [32].

Also, since the expected value in problem (2.4) was approximated with the sample mean, the resulting optimal filter will be optimal *on average*, and it may not be optimal for a particular realization. The main disadvantage of (3.1) is that the size of the optimization problem grows with the number of samples. However, recent theoretical and computational studies have shown that the sample average approximation, together with an efficient optimization algorithm, can lead to reasonable and efficient implementations [20, 21, 22, 28].

For functions ρ that are twice differentiable and convex, such as (2.3) with $1 < p < \infty$, standard Newton-like optimization methods can be used to solve problem (3.1). For this we need gradient and Hessian approximations.

The Jacobian of the errors with respect to $\boldsymbol{\alpha}$ can be written as

$$\mathbf{J} = \frac{1}{nN} \begin{bmatrix} \mathbf{J}^{(1)} \\ \vdots \\ \mathbf{J}^{(N)} \end{bmatrix}, \quad \text{where } \mathbf{J}^{(k)} = \mathbf{V}\mathbf{C}^{(k)}\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}_{\boldsymbol{\alpha}}\,, \tag{3.3}$$

and the (i,j)th entry of matrix $\boldsymbol{\phi}_{\boldsymbol{\alpha}}$ is the derivative of ϕ_i with respect to the jth entry of $\boldsymbol{\alpha}$. The gradient of $f_N(\boldsymbol{\alpha})$ and the Gauss–Newton approximation of the Hessian can be written as

$$\mathbf{g} = \frac{1}{nN}\sum_{k=1}^{N} {\mathbf{J}^{(k)}}^{\top}\mathbf{g}^{(k)} \quad \text{and} \quad \mathbf{H} = \frac{1}{nN}\sum_{k=1}^{N} {\mathbf{J}^{(k)}}^{\top}\mathbf{D}^{(k)}\mathbf{J}^{(k)}\,, \tag{3.4}$$

where the ith entry of $\mathbf{g}^{(k)}$ is the derivative $\rho'(z)$ evaluated at $e_i^{(k)}(\boldsymbol{\alpha})$, and $\mathbf{D}^{(k)}$ is a diagonal matrix, where the ith diagonal element is the second derivative $\rho''(z)$ evaluated at $e_i^{(k)}(\boldsymbol{\alpha})$. This provides the basic tools required for the implementation of efficient optimization methods. In general, we propose to use the Gauss–Newton approach with appropriate line search [24] to numerically solve problem (3.1), but the specific choice of algorithm will depend on the choice of ρ and filter representation. Specific cases will be addressed in section 4.1.

Notice that $\boldsymbol{\phi}_{\boldsymbol{\alpha}}$ in (3.3) depends on the choice of functional representation for the filter factors. In the following section, we discuss some previously proposed methods for selecting filter factors and some desirable properties for an optimal filter, motivating the development of new functional representations.

3.1. Optimal filter formulations. Researchers often parameterize the filter factors, so that representation can be done with fewer variables. For example, a simple but common approach is to use the truncated-SVD, or TSVD, filter. The basic idea is to truncate the singular values which are smaller than a certain tolerance. For TSVD, the filter factors can be written as

$$\phi_i^{\text{tsvd}}(\alpha) = \begin{cases} 1 & \text{if } i \le \alpha, \\ 0 & \text{else,} \end{cases}$$

with $\alpha \in \mathcal{A}^{\text{tsvd}} = \{1, \ldots, n\}$. Another common filtering approach is Tikhonov filtering, where, given some value $\alpha \in \mathcal{A}^{\text{tik}} = \mathbb{R}$, the filter factors are computed as

$$\phi_i^{\text{tik}}(\alpha) = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2}.$$

Notice that both TSVD and Tikhonov filtering reduce the number of unknowns from n (i.e., n filter factors) to only one, α. In standard literature, this parameter α is often referred to as a regularization parameter. As mentioned in the introduction, a variety of methods have been proposed for selecting a good parameter, but there are drawbacks for each of these approaches [16].

Optimal filters can be computed for both TSVD and Tikhonov filter representations, in which case problems (2.4) and (3.1) become optimization problems of one variable. In the case of TSVD, problem (3.1) can be solved by using a discrete version of the golden section search algorithm to calculate the *optimal TSVD filter*, $\hat{\boldsymbol{\phi}}^{\mathrm{tsvd}}$. The *optimal Tikhonov filter*, $\hat{\boldsymbol{\phi}}^{\mathrm{tik}}$, can also be computed by 1D optimization. In this case, (3.1) is a nonlinear optimization problem, and the Jacobian, which is $nN \times 1$, can be computed using the fact that the ith component of vector $\boldsymbol{\phi}_\alpha$ is given by $\frac{-2\alpha\sigma_i^2}{(\sigma_i^2+\alpha^2)^2}$. It is important to remark that there is an inherent trade-off with these standard approaches: although optimization in one dimension is much easier, the filter factors are restricted to a particular functional representation.

In this paper, we consider two other approaches to constructing optimal filters. In the first, each filter factor is chosen independently. That is, let

$$\phi_i^{\mathrm{err}}(\boldsymbol{\alpha}) = \alpha_i, \tag{3.5}$$

where $\boldsymbol{\alpha} \in \mathcal{A}^{\mathrm{err}} = \mathbb{R}^n$. Then we solve the optimal filter problem, obtaining the *optimal error filter*, $\hat{\boldsymbol{\phi}}^{\mathrm{err}}$. In this case $\boldsymbol{\phi}_{\boldsymbol{\alpha}} = \mathbf{I}$, and the Jacobian blocks can be simplified to $\mathbf{J}^{(k)} = \mathbf{V}\mathbf{C}^{(k)}\boldsymbol{\Sigma}^{-1}$. We remark that the optimal error filter can lead to excellent solutions, but a potential limitation is that a sufficient number of samples in the training set must be provided for this optimal filter to perform well on samples not in the training set. Empirically, we expect that smoother filter factors would perform better. Thus, we introduce two approaches that result in smoother filters.

A second approach to calculating optimal filters is to use a spline representation of the filter $\boldsymbol{\phi}(\boldsymbol{\alpha})$. Let $\tau_j \in [\sigma_1, \sigma_n]$, $j = 1, \ldots, m$, be distinct points. Then, for given $\alpha_j, j = 1, \ldots, m$, the spline representation s of the filter factors can be defined in terms of the data (τ_j, α_j). Evaluating the spline at σ_i gives the filter factors

$$\phi_i^{\mathrm{spl}}(\boldsymbol{\alpha}) = s(\boldsymbol{\tau}, \boldsymbol{\alpha}; \sigma_i),$$

where vectors $\boldsymbol{\tau}$ and $\boldsymbol{\alpha}$ contain entries τ_j and α_j, respectively.

In the optimal filter formulation, $\boldsymbol{\alpha} \in \mathcal{A}^{\mathrm{spl}} = \mathbb{R}^m$ contains the unknown parameters that define the spline and hence the filter factors. Thus, (3.1) has been reduced to an optimization problem in m variables, where $m < n$. The Jacobian is $nN \times m$, and $\boldsymbol{\phi}_{\boldsymbol{\alpha}}$ contains the derivative of the spline with respect to the knots, which is easy to obtain [4]. The resulting filter, $\hat{\boldsymbol{\phi}}^{\mathrm{spl}}$, will be referred to as the *optimal spline filter*.

Finally, smoothing via convolution is another approach that can be used to compute a smooth version of the optimal error filter. We write

$$\boldsymbol{\phi}^{\mathrm{smooth}} = \mathbf{K}\hat{\boldsymbol{\phi}}^{\mathrm{err}},$$

where $\mathbf{K}$ denotes a smoothing matrix. Since smoothing is done as a postprocessing step, the resulting filter, $\boldsymbol{\phi}^{\mathrm{smooth}}$, is not necessarily optimal and is referred to as the *smooth filter*.

3.2. Assessing the error. Error in the optimal filter formulation is measured using the function $\rho(z)$. In this section we provide some choices for ρ and describe simplifications that can lead to efficient implementations.

3.2.1. Mean squared error, $p = 2$. A common error function to use is the mean squared error (MSE), or specifically, the squared 2-norm of the error. In this case, we have $\rho(z)$ as in (2.3) with $p = 2$, and equation (3.2) becomes $f_N(\boldsymbol{\alpha}) = \frac{1}{2nN}\sum_{k=1}^{N} \|\mathbf{e}^{(k)}\|_2^2 = \frac{1}{2nN}\mathbf{e}^\top\mathbf{e}$, where

$$\mathbf{e} = \begin{bmatrix} \mathbf{e}^{(1)} \\ \vdots \\ \mathbf{e}^{(N)} \end{bmatrix}.$$

Thus, we have $\mathbf{g}^{(k)} = \frac{1}{2}\,\mathbf{e}^{(k)}$ and $\mathbf{D} = \frac{1}{2}\,\mathbf{I}$, and the gradient and Hessian approximation simplify to

$$\mathbf{g} = \frac{1}{2nN}\mathbf{J}^\top\mathbf{e} \quad \text{and} \quad \mathbf{H} = \frac{1}{2nN}\mathbf{J}^\top\mathbf{J},$$

respectively. This formulation leads to a least squares problem, and the necessary condition, $\mathbf{J}^\top\mathbf{e} = \mathbf{0}$, can be written as

$$\sum_{k=1}^{N} \boldsymbol{\phi}_{\boldsymbol{\alpha}}^\top \boldsymbol{\Sigma}^{-1}\mathbf{C}^{(k)}\mathbf{V}^\top(\mathbf{V}\mathbf{C}^{(k)}\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}(\boldsymbol{\alpha}) - \boldsymbol{\xi}^{(k)}) = \mathbf{0}. \tag{3.6}$$

In the optimal error filter case, (3.1) is a linear least squares problem, and (3.6) is a necessary and sufficient condition for a minimum. With simple algebraic manipulation and the assumption that $\sum_{k=1}^{N} {\mathbf{C}^{(k)}}^2$ is nonsingular, the optimal error filter $\hat{\boldsymbol{\phi}}^{\text{err}}$ can be computed explicitly as

$$\hat{\boldsymbol{\phi}}^{\text{err}} = \boldsymbol{\Sigma}\left(\sum_{k=1}^{N} {\mathbf{C}^{(k)}}^2\right)^{-1}\left(\sum_{k=1}^{N} \mathbf{C}^{(k)}\mathbf{V}^\top\boldsymbol{\xi}^{(k)}\right). \tag{3.7}$$

There is a close connection between the optimal MSE filter and the Wiener filter [34]. In particular, we show that the optimal MSE filter (3.7) provides an approximation to the Wiener filter. Suppose that the random variables $\boldsymbol{\delta}$ and $\boldsymbol{\xi}$ are uncorrelated. Then, using (3.7), the ith component of the solution is given by

$$\begin{aligned} \hat{\boldsymbol{\phi}}_i^{\text{err}} &= \sigma_i \,\frac{\sum_{k=1}^{N}\left[\mathbf{U}^\top\mathbf{b}^{(k)}\right]_i\left[\mathbf{V}^\top\boldsymbol{\xi}^{(k)}\right]_i}{\sum_{k=1}^{N}\left[\mathbf{U}^\top\mathbf{b}^{(k)}\right]_i^2} \\ &= \frac{\sigma_i^2\sum_{k=1}^{N}(\mathbf{v}_i^\top\boldsymbol{\xi}^{(k)})^2 + \sigma_i\sum_{k=1}^{N}(\mathbf{u}_i^\top\boldsymbol{\delta}^{(k)})(\mathbf{v}_i^\top\boldsymbol{\xi}^{(k)})}{\sigma_i^2\sum_{k=1}^{N}(\mathbf{v}_i^\top\boldsymbol{\xi}^{(k)})^2 + 2\sigma_i\sum_{k=1}^{N}(\mathbf{u}_i^\top\boldsymbol{\delta}^{(k)})(\mathbf{v}_i^\top\boldsymbol{\xi}^{(k)}) + \sum_{k=1}^{N}(\mathbf{u}_i^\top\boldsymbol{\delta}^{(k)})^2}. \end{aligned} \tag{3.8}$$

Using the assumption that signal and noise are uncorrelated and the assumption in section 2 that the noise has zero mean, we have for large N that

$$\frac{1}{N}\sum_{k=1}^{N}(\mathbf{u}_i^\top\boldsymbol{\delta})(\mathbf{v}_i^\top\boldsymbol{\xi}) \approx \mathbf{E}_{\boldsymbol{\delta},\boldsymbol{\xi}}\left[(\mathbf{u}_i^\top\boldsymbol{\delta})(\mathbf{v}_i^\top\boldsymbol{\xi})\right] = \mathbf{E}_{\boldsymbol{\delta}}\left[\mathbf{u}_i^\top\boldsymbol{\delta}\right]\mathbf{E}_{\boldsymbol{\xi}}\left[\mathbf{v}_i^\top\boldsymbol{\xi}\right] = 0,$$

and hence

$$\hat{\boldsymbol{\phi}}_i^{\text{err}} \approx \sigma_i^2\left(\sigma_i^2 + \frac{\frac{1}{N}\sum_{k=1}^{N}(\mathbf{u}_i^\top\boldsymbol{\delta}^{(k)})^2}{\frac{1}{N}\sum_{k=1}^{N}(\mathbf{v}_i^\top\boldsymbol{\xi}^{(k)})^2}\right)^{-1}.$$

Since the mean power spectral densities corresponding to the noise and undegraded image can be written as $S^i_{\boldsymbol{\delta}} = \mathbf{E}_{\boldsymbol{\delta}}(\mathbf{u}_i^\top \boldsymbol{\delta})^2 \approx \frac{1}{N}\sum_{k=1}^N (\mathbf{u}_i^\top \boldsymbol{\delta}^{(k)})^2$ and $S^i_{\boldsymbol{\xi}} = \mathbf{E}_{\boldsymbol{\xi}}(\mathbf{v}_i^\top \boldsymbol{\xi})^2 \approx \frac{1}{N}\sum_{k=1}^N (\mathbf{v}_i^\top \boldsymbol{\xi}^{(k)})^2$, respectively, the optimal error filter under the above assumptions provides an approximation of the Wiener filter [34, 6, 11]:

$$\boldsymbol{\phi}_i^{\text{Wiener}} = \sigma_i^2 \left(\sigma_i^2 + \frac{S^i_{\boldsymbol{\delta}}}{S^i_{\boldsymbol{\xi}}} \right)^{-1} . \tag{3.9}$$

The approximation to the Wiener filter was derived using the empirical risk minimization framework, and it is possible to show that, with some additional assumptions, the optimal error filter is equivalent to the Tikhonov filter in the Bayesian framework [30]. Assume that probability distributions $\mathcal{P}_{\boldsymbol{\delta}}$ and $\mathcal{P}_{\boldsymbol{\xi}}$ are normal with means 0 and variances η^2 and β^2, respectively. Then an analytic solution can be derived in the Bayesian framework, and in this case, the optimal error filter is the Tikhonov filter, $\phi_i^{\text{err}}(\boldsymbol{\alpha}) = \phi_i^{\text{tik}}(\alpha)$, with $\alpha = \frac{\eta}{\beta}$.

3.2.2. 1-norm. In many cases, it is desirable to minimize the sum of the absolute errors. This corresponds to the 1-norm of the error, where $\rho(z) = |z|$. Then it is well known [5] that minimizing $f(\boldsymbol{\alpha})$ is equivalent to solving the constrained optimization problem,

$$\begin{aligned} \min_{(\boldsymbol{\alpha},\mathbf{t})} \quad & \frac{1}{N}\sum_{k=1}^N \mathbf{1}^\top \mathbf{t}^{(k)} \\ \text{subject to} \quad & -\mathbf{t}^{(k)} \le \mathbf{e}^{(k)}(\boldsymbol{\alpha}) \le \mathbf{t}^{(k)}, \quad k = 1, \ldots, N, \end{aligned}$$

where $\mathbf{1}$ is a vector of ones and $\mathbf{t}^{(k)} \in \mathbb{R}^n$. This linear programming problem can be computationally difficult to solve if the number of unknowns is large. More specifically, the above problem has (number of parameters in $\boldsymbol{\alpha}$) $+ nN$ unknowns and $2nN$ constraints, making it unrealistic for many problems.

A more computationally appealing approach is to use an approximation of the 1-norm. The main difficulty with $\rho(z) = |z|$ is that it has a discontinuous first derivative at $z = 0$, causing challenges for optimization algorithms. It is common to instead approximate the 1-norm using the Huber function [19]:

$$\rho(z) = \begin{cases} |z| - \frac{\beta}{2} & \text{if } |z| \ge \beta, \\ \\ \frac{1}{2\beta} z^2 & \text{if } |z| < \beta, \end{cases}$$

for some $\beta \in \mathbb{R}^+$. This function is continuously differentiable, and efficient gradient-based optimization methods can be used to solve (3.1).

3.2.3. ∞-norm. In many cases, it is desirable to minimize the worst case error for a reconstruction, which corresponds to error

$$\text{ERR}(\boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\delta}) = \max_i |e_i(\boldsymbol{\alpha})| . \tag{3.10}$$

Then the problem of computing an optimal filter for a training set can be written as

$$\min_{\boldsymbol{\alpha}} \frac{1}{N}\sum_{k=1}^N \|\mathbf{e}^{(k)}(\boldsymbol{\alpha})\|_\infty ,$$

which can be reformulated as a nonlinear constrained optimization problem in the following way [5]:

$$\begin{aligned} \min_{(\boldsymbol{\alpha},\mathbf{t})} \quad & \frac{1}{N}\sum_{k=1}^{N} t^{(k)} \\ \text{subject to} \quad & -t^{(k)} \le e_i^{(k)}(\boldsymbol{\alpha}) \le t^{(k)}\,, \quad k=1,\dots,N,\ i=1,\dots,n\,. \end{aligned}$$

Similar to the reformulation for the 1-norm, a major disadvantage of this approach is that the number of unknowns and constraints can become prohibitively large. In particular, we have (number of parameters in $\boldsymbol{\alpha}$)$+N$ unknowns and $2nN$ constraints.

For the optimal error filter, the above optimization problem is a linear programming problem. Interior point methods for linear programming problems can be used to tackle this optimization problem.

3.2.4. *p*-norm. Other p-norms with $1 < p < 2$ or $p > 2$ can also be used. Since the p-norms for $p > 2$ have continuous first and second derivatives, those cases are rather simple extensions. However, for $1 < p < 2$, additional smoothing is required to use Newton-like algorithms, or an iteratively reweighted least squares approach [5] may be used. Here, we developed a polynomial smoothing similar to the Huber function from section 3.2.2 which is twice continuously differentiable, i.e.,

$$\rho(z) = \begin{cases} c|z|^2 + b|z|^3 + a|z|^4 & \text{if } |z| \le \beta, \\ \\ \frac{1}{p}|z|^p & \text{if } |z| > \beta, \end{cases}$$

with $a = \beta^{p-4}\,\frac{(p-2)(p-3)}{2p}$, $b = -\beta^{p-3}\,\frac{(p-2)(p-4)}{p}$, and $c = \beta^{p-2}\,\frac{(p-3)(p-4)}{2p}$. The approximation gap is largest for p close to 1 and vanishes as $p \to 2$.

4. Computational considerations and numerical results. Section 4.1 addresses some of the computational concerns that may arise during optimization, such as overcoming numerical instabilities, choice of algorithm, and preconditioning. Then in section 4.2, numerical results and comparisons are presented. It is important to remark here that the particular choice of filter representation and ρ may depend on the application, and the results presented here serve mainly as illustrations.

4.1. Computational considerations. Notice that the Jacobian, defined in (3.3), contains the matrix $\boldsymbol{\Sigma}^{-1}$. Due to division by very small singular values, constructing the Jacobian and Hessian approximation matrices as described can result in badly scaled matrices and hence severe numerical instabilities, especially for ill-posed problems. To overcome this, we incorporate the singular values directly into the definition of the filter's parameters. For example, for the optimal error filter, we define $\tilde{\boldsymbol{\phi}}^{\text{err}} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}^{\text{err}}$, optimize over $\tilde{\boldsymbol{\phi}}^{\text{err}}$, and compute the desired filter using componentwise multiplication with σ_i. Mathematically, this approach is equivalent to optimizing over $\boldsymbol{\phi}^{\text{err}}$, but in this way we avoid division by small singular values and work with a better conditioned problem. Similar ideas can be used in the other cases.

The optimization algorithm used to numerically solve problem (3.1) depends on the choice of ρ and the choice of filter representation. For the p-norm with $p \ge 2$, a Gauss–Newton algorithm has good theoretical convergence properties and performs well in practice. Since (3.8) provides an explicit formula for computing the optimal error (and hence the smooth) filter in the case $p = 2$, direct computation is recommended. A discrete golden section search algorithm can be used for all choices of

TABLE 1

Suggested numerical methods to use for different combinations of filter representations and choices of ρ. GSS refers to a discrete golden section search algorithm. GN corresponds to the Gauss–Newton method, and LS refers to solving a linear least squares system. IPM-N and IPM-L represent interior point methods for nonlinear and linear problems, respectively. Filters Tik, spl, and err refer to the Tikhonov, spline, and error filters defined in section 3.1.

Filter /	TSVD	Tik	spl	err
Huber	GSS	GN	GN	GN
$1 < p < 2$	GSS	GN	GN	GN
$p = 2$	GSS	GN	GN	LS
$p > 2$	GSS	GN	GN	GN
$p = \infty, p = 1$	GSS	IPM-N	IPM-N	IMP-L

ρ with the TSVD filter. As described in section 3.2, reformulating problems with $p = \infty$ and $p = 1$ as constrained optimization problems warrants the use of interior point methods. For the Huber function, theoretical studies suggest the use of efficient gradient-based optimization methods; however, Gauss–Newton methods show good performance in practice, and we follow this approach. These choices of methods are summarized in Table 1. We do not claim that these are the best approaches to use in all cases, but we provide them as good suggestions.

For large-scale problems where the number of unknowns and/or the number samples in the training set may become very large, special attention should be given to efficient implementations. Since the main computational cost in the Gauss–Newton method comes from solving a linear system at each step, it is common practice to use iterative methods such as the conjugate gradient method. Explicitly constructing the Jacobian and approximate Hessian matrices is not only unnecessary, but also costly (especially for 2D problems). Due to the fact that iterative methods require only the matrix-vector and possibly matrix-transpose-vector operations, we use object-oriented programming in our numerical experiments to perform the necessary operations without ever computing the matrix.

Furthermore, preconditioning can be used to accelerate convergence of iterative methods in the Gauss–Newton framework. If the diagonal matrices in (3.4) can be written as $\mathbf{D}^{(k)} = \mathbf{I} + \hat{\mathbf{D}}^{(k)}$, where $\hat{\mathbf{D}}^{(k)}$ is a low rank matrix, then $\mathbf{P} = \frac{1}{nN}\sum_{k=1}^{N} {\mathbf{J}^{(k)}}^{\top}\mathbf{J}^{(k)}$ can serve as a good preconditioner. Although fewer iterations may provide a good enough approximation in numerical experiments, Theorem 4.1 provides an upper bound on the number of iterations required for termination.

THEOREM 4.1. *Let $\mathbf{D}^{(k)} = \mathbf{I} + \hat{\mathbf{D}}^{(k)}$, where $\hat{\mathbf{D}}^{(k)}$ has rank $\eta^{(k)}$. Then the preconditioned conjugate gradient method applied to $\mathbf{Hs} = -\mathbf{g}$ with preconditioner $\mathbf{P}$ terminates with the solution in at most $\min(1 + \sum_{k=1}^{N} \eta^{(k)}, m)$ steps, where m is the number of elements in $\boldsymbol{\alpha}$. The unpreconditioned conjugate gradient approach terminates in at most m steps.*

Proof. From (3.4), we have

$$\begin{aligned}\mathbf{H} &= \frac{1}{nN}\sum_{k=1}^{N} {\mathbf{J}^{(k)}}^{\top}(\mathbf{I} + \hat{\mathbf{D}}^{(k)})\mathbf{J}^{(k)} \\ &= \mathbf{P} + \frac{1}{nN}\sum_{k=1}^{N} {\mathbf{J}^{(k)}}^{\top}\hat{\mathbf{D}}^{(k)}\mathbf{J}^{(k)}.\end{aligned}$$

Since $\mathbf{J}^{(k)}$ has full column rank, we have that $\text{rank}(\mathbf{J}^{(k)\top}\hat{\mathbf{D}}^{(k)}\mathbf{J}^{(k)}) = \text{rank}(\hat{\mathbf{D}}^{(k)}) = \eta^{(k)}$. Then by the subadditivity property of rank,

$$\text{rank}\left(\frac{1}{nN}\sum_{k=1}^{N}\mathbf{J}^{(k)\top}\hat{\mathbf{D}}^{(k)}\mathbf{J}^{(k)}\right) \leq \sum_{k=1}^{N}\eta^{(k)}\,.$$

Thus, the preconditioned conjugate gradient method is guaranteed to converge in at most $1+\sum_{k=1}^{N}\eta^{(k)}$ steps. However, regardless of preconditioning, the conjugate gradient method is guaranteed to converge in at most m steps, providing the appropriate bound. ∎

It is important to notice that for cases with small m, preconditioning may not be necessary. However, for cases where m may be large, such as with the optimal error filter, $\sum_{k=1}^{N}\eta^{(k)}$ may be significantly less than m, so the preconditioned conjugate gradient algorithm can have faster convergence. Furthermore, algebraic simplifications result in a convenient formula for computing $\mathbf{P}$ explicitly,

$$\mathbf{P} = \frac{1}{nN}\boldsymbol{\phi}_{\boldsymbol{\alpha}}^{\top}\boldsymbol{\Sigma}^{-1}\left(\sum_{k=1}^{N}\mathbf{C}^{(k)\,2}\right)\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}_{\boldsymbol{\alpha}}\,,$$

so solving systems with $\mathbf{P}$ can be done efficiently.

4.2. Numerical results. Now that we have addressed some of the computational concerns, we present some numerical results that illustrate the performance of the optimal filters. First, we generated a training set of N observations. Optimal filters were then calculated using the sample average approximation approach described in section 3 and using appropriate numerical optimization techniques (see Table 1). To validate the performance of the optimal filters, we generated a *validation set* of M different observations. The reconstructed filtered solution $\mathbf{x}_{\text{filter}}$ was then compared to the corresponding true solution $\boldsymbol{\xi}$. The methods were evaluated by computing the following:

1. Error (ERR) computed as in (2.2) and (3.10).
2. Relative Error (REL) computed as ERR in (2.2) and (3.10) divided by
$$\frac{1}{n}\sum_{i=1}^{n}\rho(\xi_i) \quad \text{and} \quad \max_i |\xi_i|, \quad \text{respectively.}$$
3. Signal-to-noise ratio (SNR_ρ) with respect to the choice of ρ used in the minimization, computed as $10\cdot\log_{10}(1/\text{REL})$.
4. Standard signal-to-noise ratio (SNR) computed as $10\cdot\log_{10}(\frac{\|\boldsymbol{\xi}\|_2^2}{\|\mathbf{x}_{\text{filter}}-\boldsymbol{\xi}\|_2^2})$.

It is important to notice that the first three of these are defined with respect to the particular choice of ρ used in the optimization. However, the last always uses the 2-norm. This standard SNR is commonly used in the image processing community to evaluate images and is shown here for comparison purposes.

In this paper, we present results for three problems: Problems 1 and 2 are 1D examples, and Problem 3 is a 2D example. Below are details of implementation. For the optimal spline filter, we use a cubic spline with "not-a-knot" boundary conditions. We select the knots to be spaced logarithmically between the smallest and the largest singular value. We used 15 knots in Problems 1 and 2 and 50 knots in Problem 3. For the smooth filter, smoothing operator $\mathbf{K}$ was a discrete Gaussian kernel with reflexive boundary conditions [17], and we selected a standard deviation of 0.01 for Problems 1 and 2, and a standard deviation of 5 for Problem 3.

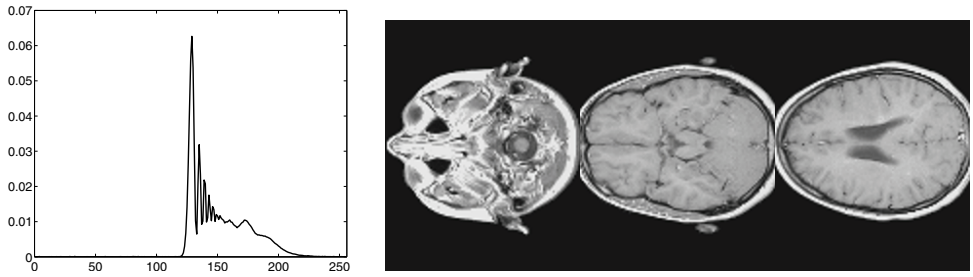

FIG. 1. (a) *Convolution kernel used in Problems* 1 *and* 2. (b) *Columns of these images are used for constructing the training set for Problems* 1 *and* 2.

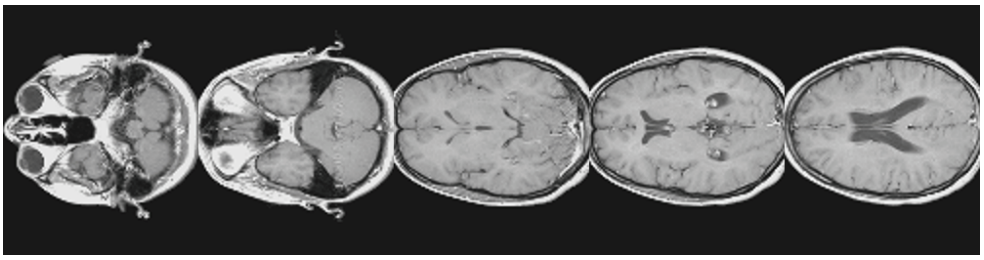

FIG. 2. *Columns of these images are used for constructing the validation set for Problems* 1 *and* 2.

4.2.1. One-dimensional examples.

Problem 1. The first example uses 1D convolution to compare the optimal filters with standard filtering approaches. Training signals are obtained by taking columns of the MRI images shown in Figure 1(b), then applying the convolution kernel found in Figure 1(a), and incorporating Gaussian white noise with noise levels in the range of 0.001 and 0.01. To be more specific, a noise level of 0.01 means $\|\boldsymbol{\delta}\|_2^2/\|\mathbf{A}\boldsymbol{\xi}\|_2^2 = 0.01$. Matrix $\mathbf{A}$ is 256×256 and has Toeplitz structure. In this example, we use 600 training samples and 2800 validation samples, each sample being 256×1. The validation set consists of noisy, blurred signals that were obtained using column slices of different MRI images. Five of the fourteen images used in the validation set are shown in Figure 2.

In our numerical results, we compute the optimal Tikhonov filter (opt-Tik) and the optimal error filter (opt-error) using the 2-norm, and we evaluate these filters using the validation set. For each validation sample, we also compute the Tikhonov solutions where the regularization parameter is selected by minimizing the corresponding GCV function [10] or by minimizing the MSE, $\|\boldsymbol{\xi} - \mathbf{x}_{\text{filter}}\|_2^2$. These standard approaches are referred to as Tikhonov-GCV (Tik-GCV) and Tikhonov-MSE (Tik-MSE), respectively. Although Tikhonov-MSE is impractical since it requires knowledge of the true signals in the validation set, we compute it for comparison purposes. An important remark is that the optimal Tikhonov filter and optimal error filter use training data, so the filters can be computed off-line. On the other hand, regularization parameters must be computed on the fly in the Tikhonov-GCV and Tikhonov-MSE approaches.

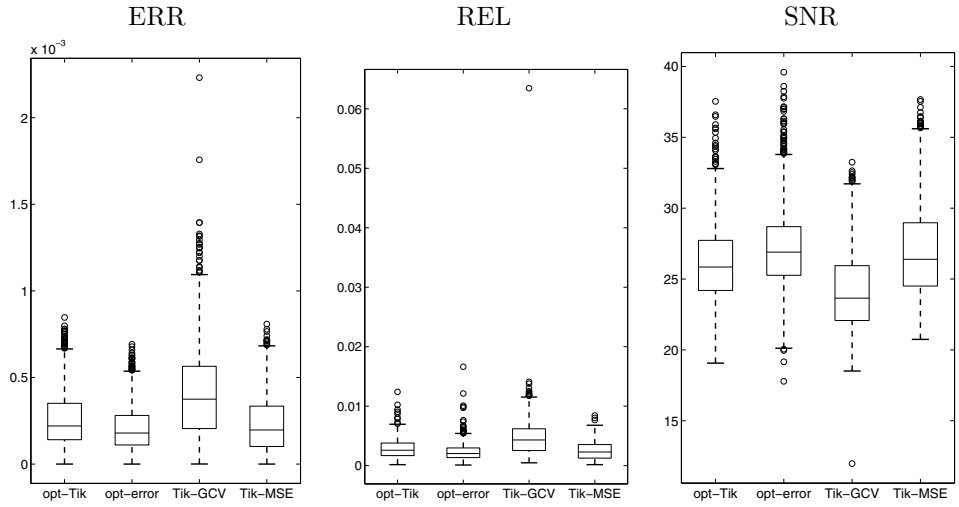

FIG. 3. *Box plots presenting error (ERR), relative error (REL), and SNR for the validation set used in Problem* 1. *All results use the* 2*-norm. Optimal filters, denoted opt-Tik and opt-error, use the training set, and Tik-GCV and Tik-MSE correspond to standard approaches.*

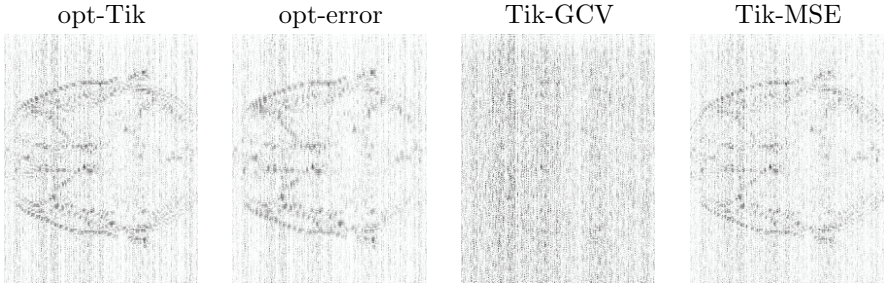

FIG. 4. *Error images (in inverted colormap, where white corresponds to* 0*) for samples in the validation set in Problem* 1. *All results use the p-norm with* $p = 2$.

Problem 1 *results.* Error is assessed by ERR, REL, and SNR to evaluate the performance of the filters. (SNR and SNR_ρ are equivalent since $p = 2$.) Box plots, or box-and-whisker plots, are presented in Figure 3 to provide a descriptive statistical illustration of the results. The box part of the plots conveniently presents the median value along with the 25th and 75th percentiles. The whiskers correspond to the extreme data points, and outliers are plotted individually.

From these results, we notice that the optimal error filter performs slightly better than all of the Tikhonov-type filters, even if the MSE is used. Furthermore, we see that the optimal Tikhonov filter can produce results that are comparable to those of the Tikhonov-MSE approach, and standard GCV consistently produces larger errors and smaller SNRs. Figure 4 presents absolute error images for one set of validation signals, in inverted colormap, so that white corresponds to zero error. From these illustrations, it is evident that using a training set to develop optimal filters can be a more accurate approach than using standard regularization parameter selection methods, even when the true solution is known.

In the above example, we used the 2-norm of the error to make fair comparisons with standard approaches, but in the next example, we investigate the use of different choices of ρ and different filter representations for the optimal filters in the 1D case.

Problem 2. We use the same setup as described for Problem 1, but include a higher noise range, allowing additive Gaussian white noise with noise levels ranging between 0.01 and 0.05. We compute optimal filters that minimize errors defined by the Huber function, the p-norm with $p = 2$, and the ∞-norm. For each ρ, the optimal TSVD (opt-TSVD), optimal Tikhonov, optimal spline (opt-spline), optimal error, and smooth filters are computed. For results corresponding to the ∞-norm, we use MATLAB's interior point methods for linear and nonlinear problems, which can be accessed using built-in functions `linprog` and `fmincon` with appropriate input options. Memory limitations from MATLAB's implementation of these algorithms restrict our training set to 30 samples for the ∞-norm, but the results are still illustrative and serve as a proof of concept.

Problem 2 *results.* Box plots in Figure 5 compare the performance of the optimal filters on the validation set. The smooth filter is also included in the comparisons. From these results we see that the optimal spline and optimal error filter, as well as the smooth filter, perform better than the optimal TSVD and optimal Tikhonov filter for the Huber function and the p-norm with $p = 2$. For the ∞-norm, the optimal error filter and the smooth filter perform worse than the other approaches on the validation set, which can be attributed to the fact that only a small number of signals were used in the training set.

4.2.2. Two-dimensional example. Filtering examples arise naturally in image processing applications. In this section, we use an example from image deblurring to illustrate how optimal filters can be computed and to compare the performance of different filters and different choices of ρ for a 2D problem. Image deblurring problems can be modeled as in (2.1), where $\mathbf{b}$ is the observed blurred noisy image, $\mathbf{A}$ represents the blurring process, $\boldsymbol{\xi}$ is the true image, and $\boldsymbol{\delta}$ is additive noise. Matrix $\mathbf{A}$ is typically very large and sparse. However, depending on the assumed boundary conditions for the image, $\mathbf{A}$ can be very structured, and fast transform algorithms such as the fast Fourier transform (FFT) and discrete cosine transform (DCT) can be used to efficiently diagonalize $\mathbf{A}$. See [17] for more information about structured matrices and image deblurring.

Problem 3. For the training set, we use eight images of satellites with 100 rigid manipulations of each, giving a total of 800 training images. The rigid transformations include rotation, translation, and magnification, all defined using a normally distributed parameter. The blur used in the forward model is a symmetric Gaussian point spread function, and reflexive boundary conditions are assumed. After applying the blur operator to each of the 800 images, we include normally distributed Gaussian noise with a noise level that was randomly selected (equally distributed) from a range of 0.1 to 0.15. Then for the validation set, we use eight different images of satellites with 100 random rigid manipulations each and allow the noise level to vary between 0.1 and 0.15 (equally distributed). Four of the training images and four of the validation images are shown in the top and bottom rows, respectively, of Figure 6. Each of the images is 256×256 pixels.

Problem 3 *results.* For the 2D example, we use the Huber function and a p-norm with $p = 1.5$, 2, and 4. Box plots presenting the performance of the computed filters

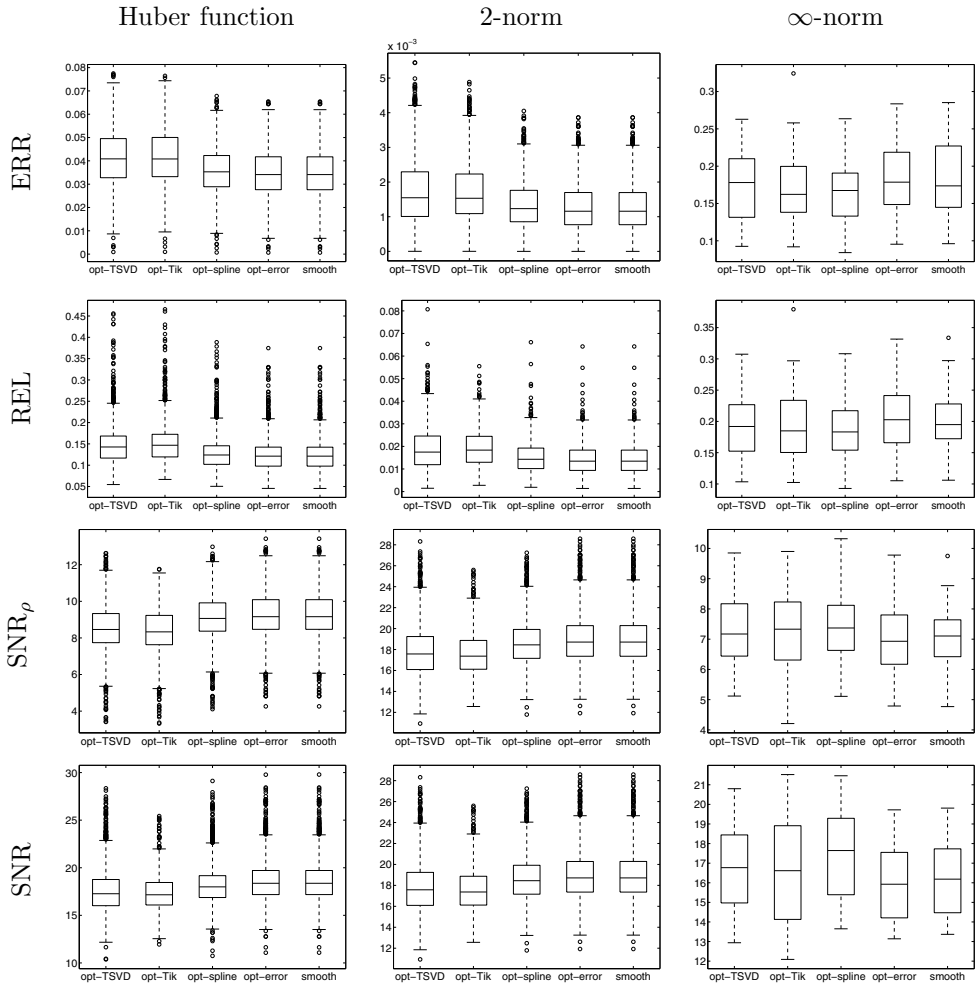

FIG. 5. *Box plots for error (ERR), relative error (REL), SNR$_\rho$, and SNR on the validation set for Problem* 2. *We use the Huber function, the* 2*-norm, and the* ∞*-norm. Optimal filters are denoted with prefix "opt-."*

on the validation set are presented in Figure 7. The general trend is that the optimal spline, optimal error, and smooth filters produce smaller errors and larger SNRs than optimal TSVD and optimal Tikhonov. We remark that the noise level is larger in this example, which partly explains why the performance of Tikhonov filtering is not as good.

The computed optimal filters are presented in Figure 8 for the four different choices of ρ. The filters are presented as functions of the size of the singular values. It is interesting to note the different shapes that the filters take depending on the choice of ρ. For clarity of display, we do not show the smooth filter, which in all cases resembles the optimal error filter but with a thinner spread. To give an idea of the computational time required to compute these filters, for the p-norm with $p = 2$, computing the optimal TSVD, optimal Tikhonov, optimal spline, and optimal error filter for this problem took roughly 606 seconds, 1787 seconds, 265 seconds, and 237

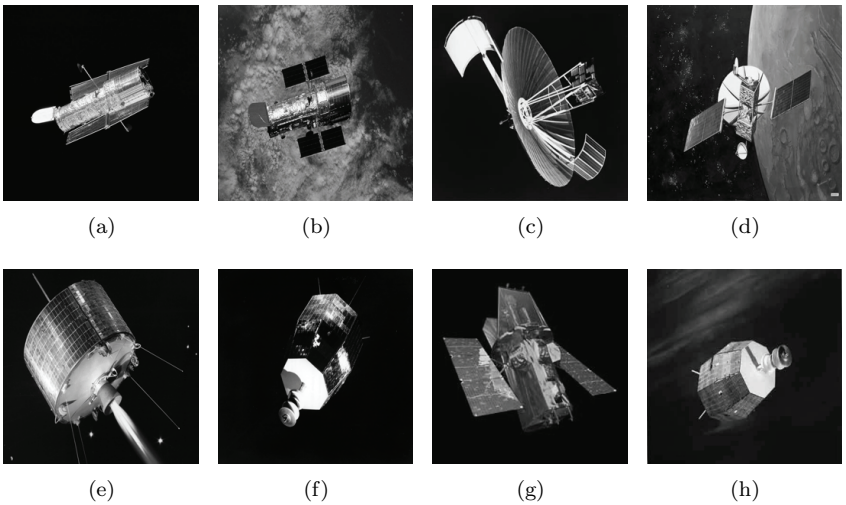

FIG. 6. *Images shown in* (a)–(d) *are some of the prototype true images used for generating the training set in Problem* 3. *Rotations, translations, and magnifications of the true images are considered. To validate the performance of the optimal filters, we generate a validation set of rotated, translated, and magnified images. Sample images used in the validation set are shown in* (e)–(h).

seconds, respectively, on a Macbook Pro with OS-X 10.6 and 8GB memory, running MATLAB 7.10.0 (64-bit).

For one sample from the validation set, we present in Figure 9 the absolute error images between the reconstructed image and the true image (with inverted colormap so that white corresponds to zero error). It is evident from these images that the Huber function in all filter representations does a better job at approximating the background or smooth information in the images. Furthermore, in all cases, the optimal spline and optimal error filters tend to perform better than optimal TSVD and optimal Tikhonov.

Finally, we remark that the training images can be used to obtain uncertainty estimates for the reconstructions. For each computed optimal filter, we reconstruct all of the training images and evaluate the average error per pixel. The expected error in each pixel is approximated by the sample mean

$$\mu_i = \frac{1}{N}\sum_{k=1}^{N} e_i^{(k)}\,, \qquad i = 1,\ldots,n\,,$$

and the average error in reconstructing all of the training images

$$\bar{\mu} = \frac{1}{n}\sum_{i=1}^{n} \mu_i$$

is presented in Table 2 for both the training and validation images. In Table 2 we also report in parentheses the standard deviation ν of the nN error samples. These results demonstrate that the mean and standard deviation of the pixel error for the training images are good estimates for the mean and standard deviation of the pixel error for the validation images. Since the standard deviation is small, we have a good estimate of the uncertainty in the estimate of each pixel value.

FIG. 7. *Box plots for error (ERR), relative error (REL), SNR$_\rho$, and SNR on the validation set for Problem* 3. *We use the Huber function and the p-norm with $p = 1.5, 2$, and* 4. *Optimal filters are denoted with prefix "opt-."*

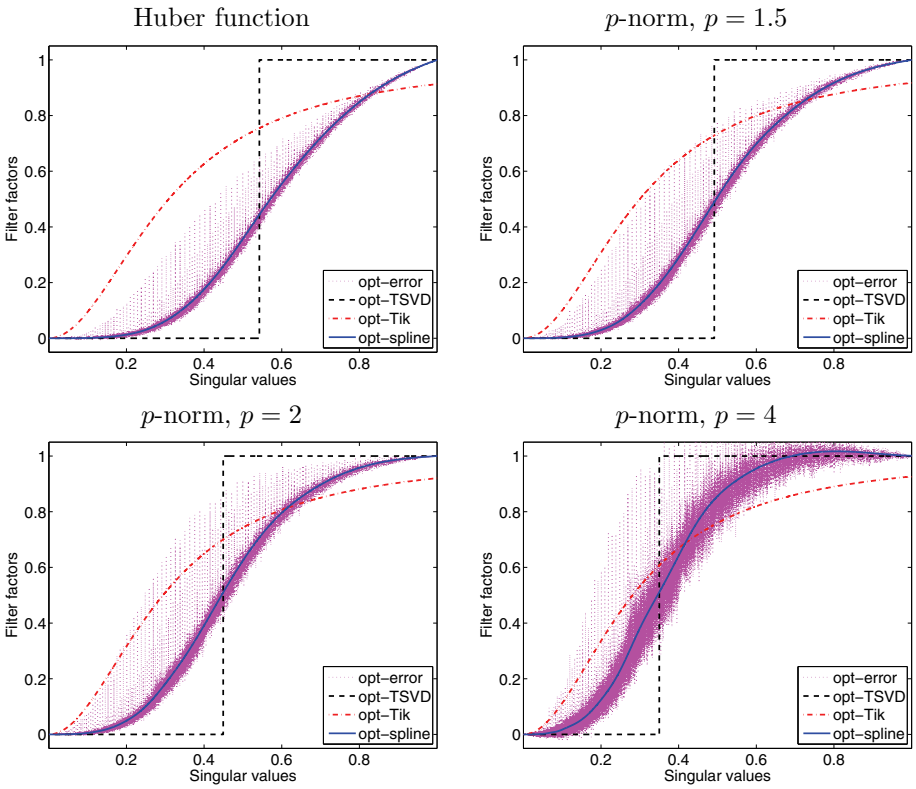

FIG. 8. *Computed optimal filters with respect the the size of the singular values for Problem* 3. *We use the Huber function and the p-norm with* $p = 1.5, 2,$ *and* 4.

5. Conclusions and discussion. Computing regularization parameters for spectral filtering can be difficult and expensive; however, in this paper, we have described a robust and efficient approach to computing optimal filters for an inverse problem using a training set. We formulated the problem as an empirical Bayes risk minimization problem and described sophisticated numerical optimization algorithms that can be used to compute optimal filters. A variety of error measures and filter representations were considered. The optimal filters are desirable because they can be computed off-line, leading to faster on-line computations.

Since data from a specific application are typically very similar in nature, only a few samples in the training set may be required to compute a good filter. Furthermore, in many applications, a precise noise level may be difficult to obtain. Our approach allows the experimentalist to provide a range of noise levels. Although a potential disadvantage of our approach is that the SVD of $\mathbf{A}$ is required, structure or sparsity can often be exploited.

Future directions include adapting the approaches to different choices of ρ [5] or different noise distributions. Incorporating additional constraints on the spectral filters or reconstructions, such as nonnegativity or sparsity, may also be desired. Extensions to learning regularization functionals could also be considered [15]. An efficient algorithm for the ∞-norm for large-scale problems with large training sets is a topic of interest. Methods for combining sources can reduce the size of the problem [13], while efficient numerical interior point methods can tackle the problem algorith-

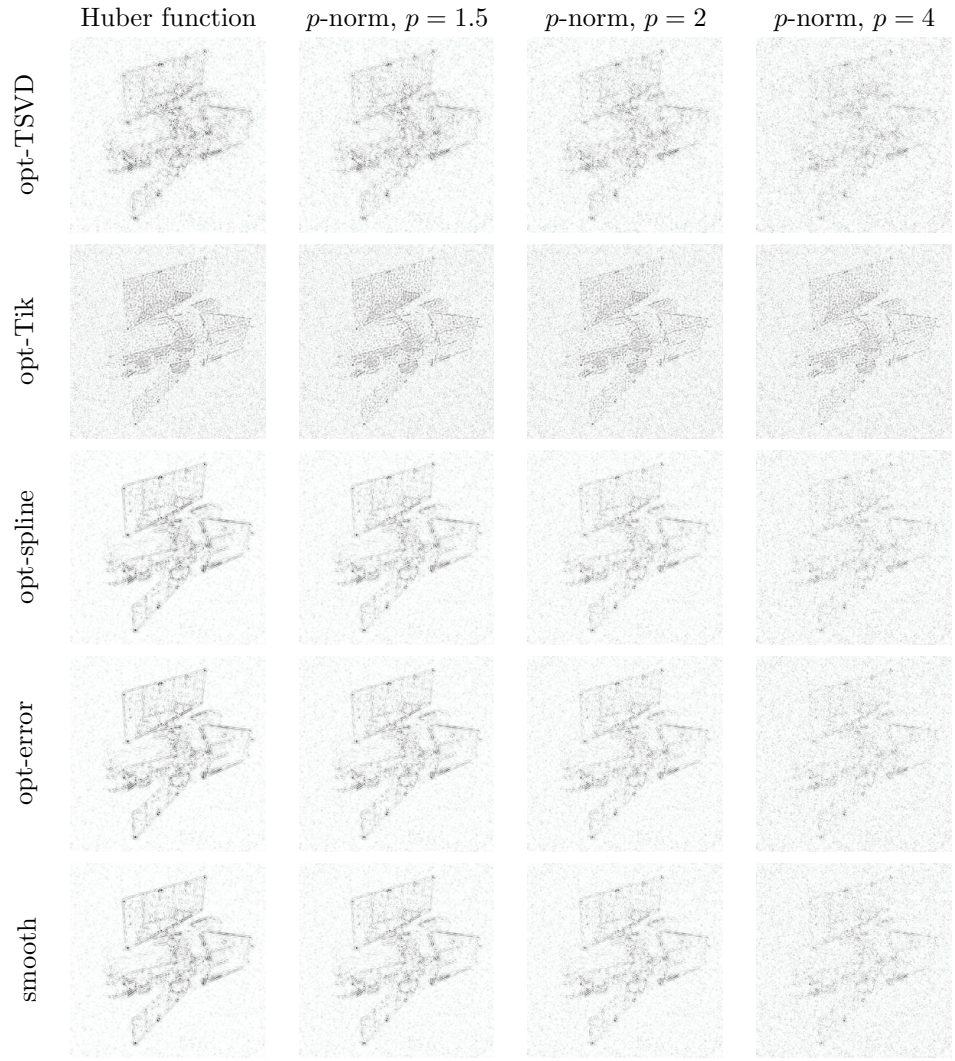

FIG. 9. *Comparison of error images (in inverted colormap so that white corresponds to zero) for one sample from the validation set for Problem* 3. *We use the Huber function and the p-norm with* $p = 1.5, 2,$ *and* 4.

mically [35]. New developments in the stochastic programming community on hybrid methods or adaptive filters may help with large data sets and provide improvements in computing optimal filters.

Last, we remark that the optimal filter problem can be interpreted as an *optimal experimental design* problem, where the filter factors represent design parameters [2, 26, 7]. Since the optimal filters presented in this paper are optimal in mean, they correspond to an average optimality criterion, i.e., A-optimality design, in the optimal design literature. However, other design objectives can also be considered. For instance, optimization of the worst-case scenario might be of interest in quality control, leading to min-max designs [2, 26].

TABLE 2

Comparison of average error, $\bar{\mu}$, and standard deviation, ν, of the pixel error for training and validation sets for Problem 3. *"T" corresponds to training images, and "V" corresponds to validation images. We use the Huber function and the p-norm with $p = 1.5, 2,$ and* 4. *Four filter representations are considered: optimal-TSVD, optimal-Tikhonov, optimal-spline, and optimal-error.*

		Huber function	p-norm, $p = 1.5$	p-norm, $p = 2$	p-norm, $p = 4$
opt-TSVD	T	-2.35e-4 (5.75e-3)	-2.04e-4 (5.36e-3)	-1.79e-4 (5.05e-3)	-1.29e-4 (4.37e-3)
	V	-2.54e-4 (6.03e-3)	-2.21e-4 (5.63e-3)	-1.95e-4 (5.31e-3)	-1.43e-4 (4.60e-3)
opt-Tik	T	-1.94e-2 (7.79e-3)	-1.84e-2 (7.46e-3)	-1.77e-2 (7.24e-3)	-1.63e-2 (6.79e-3)
	V	-2.32e-2 (1.01e-2)	-2.20e-2 (9.60e-3)	-2.12e-2 (9.29e-3)	-1.96e-2 (8.67e-3)
opt-spline	T	-5.50e-4 (5.53e-3)	-3.64e-4 (4.93e-3)	-2.18e-4 (4.56e-3)	1.01e-4 (3.95e-3)
	V	-6.08e-4 (5.80e-3)	-4.00e-4 (5.18e-3)	-2.34e-4 (4.79e-3)	1.33e-4 (4.17e-3)
opt-error	T	-5.61e-4 (4.75e-3)	-3.30e-4 (4.02e-3)	-1.42e-4 (3.58e-3)	2.61e-4 (3.15e-3)
	V	-6.34e-4 (4.99e-3)	-3.71e-4 (4.24e-3)	-1.57e-4 (3.78e-3)	3.25e-4 (3.37e-3)

Acknowledgments. We would like to thank the referees for helpful comments that improved the presentation. The satellite images used in the image deblurring problem were obtained from NASA's website: www.nasa.gov.

REFERENCES

[1] A. AHLÉN AND M. STERNAD, *Wiener filter design using polynomial equations*, IEEE Trans. Signal Process., 39 (2002), pp. 2387–2399.
[2] A. C. ATKINSON, A. N. DONEV, AND R. TOBIAS, *Optimum Experimental Designs, with SAS*, Oxford University Press, Oxford, UK, 2007.
[3] J. O. BERGER, *Statistical Decision Theory and Bayesian Analysis*, Springer, New York, 1985.
[4] C. DE BOOR, *A Practical Guide to Splines*, Springer, Heidelberg, 2001.
[5] S. BOYD AND L. VANDENBERGHE, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.
[6] T. F. CHAN AND J. SHEN, *Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods*, SIAM, Philadelphia, 2005.
[7] M. CHUNG AND E. HABER, *Experimental design for biological systems*, SIAM J. Control Optim., submitted.
[8] D. FOUSKAKIS AND D. DRAPER, *Stochastic optimization: A review*, Internat. Statist. Rev., 70 (2002), pp. 315–349.
[9] D. GHIGLIA, *Space-invariant deblurring given N independently blurred images of a common object*, J. Opt. Soc. Amer. A, 1 (1984), pp. 398–402.
[10] G. GOLUB, M. HEATH, AND G. WAHBA, *Generalized cross-validation as a method for choosing a good ridge parameter*, Technometrics, 21 (1979), pp. 215–223.
[11] R. C. GONZALEZ AND R. E. WOODS, *Digital Image Processing*, Prentice–Hall, Upper Saddle River, NJ, 2002.
[12] L. GUAN AND R. WARD, *Restoration of randomly blurred images by the Wiener filter*, IEEE Trans. Acoust. Speech Signal Process., 37 (1989), pp. 589–592.
[13] E. HABER, M. CHUNG, AND F. HERMANN, *An effective method for parameter estimation with PDE constraints with multiple right hand sides*, SIAM J. Optim., submitted.
[14] E. HABER, L. HORESH, AND L. TENORIO, *Numerical methods for experimental design of large-scale linear ill-posed inverse problems*, Inverse Problems, 24 (2008), 55012.
[15] E. HABER AND L. TENORIO, *Learning regularization functionals—A supervised training approach*, Inverse Problems, 19 (2003), pp. 611–626.
[16] P. C. HANSEN, *Rank-deficient and Discrete Ill-posed Problems: Numerical Aspects of Linear Inversion*, SIAM Monogr. Math. Model. Comput. 4, SIAM, Philadelphia, 1997.
[17] P. C. HANSEN, J. G. NAGY, AND D. P. O'LEARY, *Deblurring Images: Matrices, Spectra, and Filtering*, Fundam. Algorithms 3, SIAM, Philadelphia, 2006.
[18] L. HORESH AND E. HABER, *Sensitivity computation of the l1 minimization problem and its application to dictionary design of ill-posed problems*, Inverse Problems, 25 (2009), 095009.
[19] P. J. HUBER, *Robust estimation of a location parameter*, Ann. Math. Statist., 35 (1964), pp. 73–101.

[20] G. Lan, A. Nemirovski, and A. Shapiro, *Validation analysis of mirror descent stochastic approximation method*, Math. Program., to appear; DOI: 10.1007/x10107-011-0442-6.
[21] J. Linderoth, A. Shapiro, and S. Wright, *The empirical behavior of sampling methods for stochastic programming*, Ann. Oper. Res., 142 (2006), pp. 215–241.
[22] W. Mak, D. Morton, and R. Wood, *Monte Carlo bounding techniques for determining solution quality in stochastic programs*, Oper. Res. Lett., 24 (1998), pp. 47–56.
[23] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, *Robust stochastic approximation approach to stochastic programming*, SIAM J. Optim., 19 (2009), pp. 1574–1609.
[24] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, New York, 1999.
[25] D. P. O'Leary, *Near-optimal parameters for Tikhonov and other regularization methods*, SIAM J. Sci. Comput., 23 (2001), pp. 1161–1171.
[26] F. Pukelsheim, *Optimal Design of Experiments*, Classics Appl. Math. 50, SIAM, Philadelphia, 2006.
[27] A. Shapiro, D. Dentcheva, and R. Ruszczynski, *Lectures on Stochastic Programming: Modeling and Theory*, MOS-SIAM Ser. Optim. 9, SIAM, Philadelphia, 2009.
[28] A. Shapiro and A. Nemirovski, *On complexity of stochastic programming problems*, Appl. Optim., 99 (2005), pp. 111–146.
[29] J. C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*, Wiley, San Francisco, 2003.
[30] L. Tenorio, *Statistical regularization of inverse problems*, SIAM Rev., 43 (2001), pp. 347–366.
[31] S. P. Uryasev and P. M. Pardalos, *Stochastic Optimization: Algorithms and Applications*, Springer, Heidelberg, 2001.
[32] V. N. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998.
[33] E. De Vito, L. Rosasco, A. Caponnetto, U. De Giovannini, and F. Odone, *Learning from examples as an inverse problem*, J. Mach. Learning Res., 6 (2005), pp. 883–904.
[34] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*, Wiley, New York, 1949.
[35] L. B. Winternitz, S. O. Nicholls, A. Tits, and D. P. O'Leary, *A constraint-reduced variant of Mehrotra's predictor-corrector algorithm*, Comput. Optim. Appl., (2011), DOI: 10.1007/s10589-010-9389-4.