Technical Report

TR-2012-010

Constrained Numerical Optimization Methods for Blind Deconvolution

by

A. Cornelio, E. Loli Piccolomini, J. G. Nagy

Mathematics and Computer Science

EMORY UNIVERSITY

CONSTRAINED NUMERICAL OPTIMIZATION METHODS FOR BLIND DECONVOLUTION

ANASTASIA CORNELIO*, ELENA LOLI PICCOLOMINI†, AND JAMES G. NAGY‡

Abstract. This paper describes a nonlinear least squares framework to solve a separable nonlinear ill-posed inverse problems that arises in blind deconvolution. It is shown that with proper constraints and well chosen regularization parameters, it is possible to obtain an objective function that is fairly well behaved and the nonlinear minimization problem can be effectively solved by a Gauss-Newton method. Although uncertainties in the data and inaccuracies of linear solvers make it unlikely to obtain a smooth and convex objective function, it is shown that implicit filtering optimization methods can be used to avoid becoming trapped in local minima. Computational considerations, such as computing the Jacobian, are discussed, and numerical experiments are used to illustrate the behavior of the algorithms. Although the focus of the paper is on blind deconvolution, the general mathematical model addressed in this paper, and the approaches discussed to solve it, arise in many other applications.

Key words. nonlinear least squares, variable projection, implicit filtering, Tikhonov regularization, blind deconvolution

AMS Subject Classifications: 65F20, 65F30

1. Introduction.

In many imaging situations, from astronomy to microscopy, the observed image is degraded by blurring and noise [1, 15, 22]. Although the blurring can be partially removed through the use of sophisticated, and expensive, imaging devices, computational post processing techniques are also often used to further improve the resolution of the image. This post processing is usually called *deconvolution* when the true blurring operator is known, whereas *blind deconvolution* implies that the blurring operator is not known.

Blind deconvolution can be modeled as an inverse problem of the form

$$\boldsymbol{b} = \boldsymbol{A}(\boldsymbol{y}_{\text{true}})\boldsymbol{x}_{\text{true}} + \boldsymbol{\eta} \tag{1.1}$$

where $\boldsymbol{b} \in \mathcal{R}^n$ is the measured, blurred and noisy image ($\boldsymbol{\eta}$ models the noise), and $\boldsymbol{x}_{\text{true}} \in \mathcal{R}^n$ represents the unknown true image. The vector $\boldsymbol{y}_{\text{true}} \in \mathcal{R}^m$ is unknown, and $\boldsymbol{A}(\boldsymbol{y})$ is a nonlinear operator, which models the blurring, that maps $\boldsymbol{y}$ into an $n \times n$ matrix. The matrix $\boldsymbol{A}(\boldsymbol{y})$ is typically severely ill-conditioned, with singular values that cluster at zero. We call equation (1.1) a *separable* inverse problem because the measured data depends linearly on the unknown vector $\boldsymbol{x}$, and nonlinearly on the unknown vector $\boldsymbol{y}$. In addition to blind deconvolution, separable inverse problems arise in many applications, such as super-resolution (which is an example of image data fusion) [7, 17], cryo-EM microscopy imaging [10, 11, 16, 23, 26, 29], and in seismic imaging applications [14].

Given the data $\boldsymbol{b}$ and the mapping $\boldsymbol{A}(\cdot)$, the aim is to compute approximations $\boldsymbol{x}$ and $\boldsymbol{y}$ of, respectively, $\boldsymbol{x}_{\text{true}}$ and $\boldsymbol{y}_{\text{true}}$. Generally this is done by defining an objective function, $f_0(\boldsymbol{x}, \boldsymbol{y})$, and then using an optimization algorithm to solve

$$\min_{\boldsymbol{x},\boldsymbol{y}} f_0(\boldsymbol{x}, \boldsymbol{y})\,. \tag{1.2}$$

*Department of Pure and Applied Mathematics, University of Modena and Reggio Emilia. Email: anastasia.cornelio@unimore.it.

†Department of Mathematics, University of Bologna. Email: elena.loli@unibo.it.

‡Department of Mathematics and Computer Science, Emory University. Email: nagy@mathcs.emory.edu. Research supported in part by the AFOSR under grant FA9550-09-1-0487 and by the NSF under grant DMS-0811031.

To solve the nonlinear problem (1.2) we use the variable projection method [12, 13, 18, 19, 25, 28, 30] to obtain a reduced cost functional,

$$f(\boldsymbol{y}) = f_0(\boldsymbol{x}(\boldsymbol{y}), \boldsymbol{y}), \tag{1.3}$$

that depends only on $\boldsymbol{y}$. In general, computing a minimum of f is difficult because it may not have a well defined global minimum, and there are many local minima in which an iterative solver, such as a standard Gauss-Newton method [20, 24], can become trapped. To address these challenges, we show that if we can compute physically realistic approximations of $\boldsymbol{x}_{\text{true}}$ for a fixed $\boldsymbol{y}$, then it is possible to obtain an objective function f having less local minima and whose global minimum is nearer to the exact solution.

Because the problem is ill-posed, usually the objective function f_0 incorporates regularization. Moreover, further knowledge about the application should be used to include appropriate constraints on $\boldsymbol{x}$ and $\boldsymbol{y}$. In this paper we consider problems where $\boldsymbol{x} \geq 0$, and we use a least squares fit-to-data term with Tikhonov regularization on $\boldsymbol{x}$. That is, our optimization problem is

$$\min_{\boldsymbol{x},\boldsymbol{y}} f_0(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{A}(\boldsymbol{y})\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda^2 \|\boldsymbol{x}\|_2^2 \quad \text{subject to } \boldsymbol{x} \geq \boldsymbol{0}. \tag{1.4}$$

We remark that in many applications, especially in image processing, a nonnegativity constraint on $\boldsymbol{x}$ is used to obtain physically meaningful solutions (i.e., when $\boldsymbol{x}$ contains pixels intensity values, these should be nonnegative).

An aim of this paper is to show that nonnegativity constraints result in a better behaved objective function, which results in computing a better solution of problem (1.2) when using a standard optimization method such as Gauss-Newton. We propose a new computationally convenient approximation of the Jacobian. In addition, we show that with proper constraints and good regularization parameter choice methods, the objective function $f_0(\boldsymbol{x}, \boldsymbol{y})$ will have the form

$$f_0(\boldsymbol{x}, \boldsymbol{y}) = f_s(\boldsymbol{x}, \boldsymbol{y}) + f_e(\boldsymbol{x}, \boldsymbol{y}),$$

where f_s is a smooth function that captures the large scale features of the objective function, and which has a well defined and easy to find global minimum, and the perturbation function f_e captures relatively small perturbations associated with unavoidable local minima. Implicit filtering [21] is then used to solve the optimization problem.

This paper is organized as follows. In Section 2 we describe the variable projection method that can be used to obtain a reduced cost functional that depends only on $\boldsymbol{y}$, and which can then be solved using the Gauss-Newton method. In Section 3 we discuss computation of the Jacobian of the reduced cost functional, in the context of nonnegativity constraints on $\boldsymbol{x}$. In Section 4 we describe a model problem in blind deconvolution. In Section 5 we present some numerical experiments; in particular, we investigate how incorporating physically realistic constraints on $\boldsymbol{x}$ and using well chosen regularization parameters can have a significant effect on the behavior of the objective function. Concluding remarks are given in Section 6.

2. Variable Projection. The variable projection method [12, 13, 18, 19, 25, 28, 30] can be applied to the objective function given in equation (1.4), so that the linear variable $\boldsymbol{x}$ can be implicitly eliminated, resulting in a nonlinear reduced cost functional depending only on $\boldsymbol{y}$ (1.3).

In our case, $\boldsymbol{x}(\boldsymbol{y})$ is the *nonnegative* linear least squares solution of:

$$\boldsymbol{x}(\boldsymbol{y}) = \min_{\boldsymbol{x}\geq \boldsymbol{0}} \|\boldsymbol{A}(\boldsymbol{y})\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{x}\|_2^2. \tag{2.1}$$

Like in any nonnegatively constrained optimization problem, for any $\boldsymbol{x} \geq 0$ we can define:

- the active set by $\mathcal{A}(\boldsymbol{x}) = \{i \,|\, x_i = 0\}$;
- and a diagonal matrix $\boldsymbol{D}(\boldsymbol{x})$ by

$$\boldsymbol{D}(\boldsymbol{x})_{ii} = \begin{cases} 1 & \text{if} \quad i \notin \mathcal{A}(\boldsymbol{x}) \\ 0 & \text{if} \quad i \in \mathcal{A}(\boldsymbol{x}) \end{cases} .$$

In the following we will use the notation $\boldsymbol{D} = \boldsymbol{D}(\boldsymbol{x}(\boldsymbol{y}))$, where $\boldsymbol{x}(\boldsymbol{y})$ is defined by (2.1).

Therefore the solution of (2.1) can be written by a closed formula as in [4]:

$$\boldsymbol{x}(\boldsymbol{y}) = \big(\boldsymbol{A}(\boldsymbol{y})\boldsymbol{D}\big)^{\dagger}_{\lambda} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} \tag{2.2}$$

where $\big(\boldsymbol{A}(\boldsymbol{y})\boldsymbol{D}\big)^{\dagger}_{\lambda}$ is the pseudoinverse of the $2n \times n$ matrix $\begin{bmatrix} \boldsymbol{A}(\boldsymbol{y})\boldsymbol{D} \\ \lambda \boldsymbol{I} \end{bmatrix}$.

By using (2.2), we obtain the following minimization problem:

$$\min_{\boldsymbol{y}} f(\boldsymbol{y}) = \left\| \boldsymbol{A}(\boldsymbol{y})\big(\boldsymbol{A}(\boldsymbol{y})\boldsymbol{D}\big)^{\dagger}_{\lambda} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} - \boldsymbol{b} \right\|_2^2 \tag{2.3}$$

which can be solved with a classical nonlinear least squares method, such as the Gauss-Newton algorithm [24]. The computation of the Jacobian $\boldsymbol{J}$ of $f(\boldsymbol{y})$ is discussed in the next section.

Gauss-Newton Algorithm to solve: $\min_{\boldsymbol{y}} f(\boldsymbol{y})$

choose initial $\boldsymbol{y}_0$

for $l = 0, 1, 2, \ldots$

choose λ_l

$\boldsymbol{x}_l = \arg\min_{\boldsymbol{x}} \left\| \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{A}(\boldsymbol{y}_l)\boldsymbol{D} \\ \lambda_l \boldsymbol{I} \end{bmatrix} \boldsymbol{x} \right\|_2$

$\boldsymbol{r}_l = \boldsymbol{b} - \boldsymbol{A}(\boldsymbol{y}_l)\boldsymbol{x}_l$

$\boldsymbol{d}_l = \arg\min_{\boldsymbol{d}} \|\boldsymbol{J}_l\boldsymbol{d} - \boldsymbol{r}_l\|_2$, where $\boldsymbol{J}_l$ is the Jacobian of f

determine step length τ_l

$\boldsymbol{y}_{l+1} = \boldsymbol{y}_l + \tau_l \boldsymbol{d}_l$

end

In [8] the authors apply the variable projection method to problem without the nonnegative constraints. In [30] the authors deal with a nonnegative least squares problem, but it is a small size and well-posed problem that does not require any regularization. In the next section we discuss the important step of computing the Jacobian matrix.

3. Computing the Jacobian. The Gauss-Newton method requires the computation of $\boldsymbol{J}(f(\boldsymbol{y}))$, the Jacobian matrix of $f(\boldsymbol{y})$. Recall that $f(\boldsymbol{y})$ can be expressed as

$$f(\boldsymbol{y}) = \left\| \boldsymbol{A}(\boldsymbol{y})(\boldsymbol{A}(\boldsymbol{y})\boldsymbol{D})^{\dagger}_{\lambda} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} - \boldsymbol{b} \right\|_2^2 \tag{3.1}$$

and that the k-th column of $\boldsymbol{J}(f(\boldsymbol{y}))$ is $\mathrm{vec}\big(\boldsymbol{J}_k f\big)$, where

$$\boldsymbol{J}_k f = \frac{\partial}{\partial \boldsymbol{y}_k} \Big(\boldsymbol{A}(\boldsymbol{y})(\boldsymbol{A}(\boldsymbol{y})\boldsymbol{D})^{\dagger}_{\lambda} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} \Big) . \tag{3.2}$$

To reduce the computational complexity of (3.1), a pseudo Jacobian can be computed, for example with one of the following formulations.

1. The Jacobian of the function $f(\boldsymbol{y})$ can be approximated, as proposed in [30], with the Jacobian of

$$g(\boldsymbol{y}) = \left\| \boldsymbol{A}(\boldsymbol{y})(\boldsymbol{A}(\boldsymbol{y}))^{\dagger}_{\lambda} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} - \boldsymbol{b} \right\|_2^2, \tag{3.3}$$

   where $\boldsymbol{A}(\boldsymbol{y})^{\dagger}_{\lambda}$ is the pseudoinverse of the matrix $\begin{bmatrix} \boldsymbol{A}(\boldsymbol{y}) \\ \lambda \boldsymbol{I}_n \end{bmatrix}$. Approximating $\boldsymbol{J}(f)$ with $\boldsymbol{J}(g)$ means neglecting the constraints on $\boldsymbol{x}$ in the Jacobian computation.
2. In [18, 19], Kaufman proposed a pseudo-Jacobian computation for the unconstrained objective function $g(\boldsymbol{y})$. By denoting with $\nabla_k(\boldsymbol{A})$ the $n \times n$ matrix (corresponding to the k-th column of the Jacobian):

$$\nabla_k(\boldsymbol{A}) = \frac{\partial \boldsymbol{A}(\boldsymbol{y})}{\partial \boldsymbol{y}_k}$$

   and with $\nabla_k(\boldsymbol{A}_\lambda)$ the $2n \times n$ matrix

$$\nabla_k(\boldsymbol{A}_\lambda) = \frac{\partial \boldsymbol{A}_\lambda(\boldsymbol{y})}{\partial \boldsymbol{y}_k} ,$$

   after some algebraic manipulation, we obtain [30]:

$$\boldsymbol{J}_k g = \Big[\nabla_k(\boldsymbol{A}) - \boldsymbol{A}(\boldsymbol{A}_\lambda^H \boldsymbol{A}_\lambda)^{-1} \Big(\nabla_k(\boldsymbol{A}_\lambda)^H \boldsymbol{A} + \boldsymbol{A}_\lambda^H \nabla_k(\boldsymbol{A}_\lambda) \Big) \Big] \boldsymbol{A}_\lambda^{\dagger} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} , \tag{3.4}$$

   where $\boldsymbol{A}^H$ is used to denote the Hermitian (complex conjugate) transpose of $\boldsymbol{A}$. In [18, 19] the second term is ignored and the proposed *Kaufman simplified* Jacobian is:

$$\widetilde{\boldsymbol{J}}_k g = \big(\nabla_k(\boldsymbol{A}) - \boldsymbol{A}\boldsymbol{A}_\lambda^{\dagger} \nabla_k(\boldsymbol{A}) \big) \boldsymbol{A}_\lambda^{\dagger} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} = (\boldsymbol{I} - \boldsymbol{A}\boldsymbol{A}_\lambda^{\dagger}) \nabla_k(\boldsymbol{A}) \boldsymbol{A}_\lambda^{\dagger} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix} \tag{3.5}$$

Before considering the nonnegative case, we remark that

- $\nabla_k(\boldsymbol{A}\boldsymbol{D}) = \nabla_k(\boldsymbol{A})\boldsymbol{D}$, and
- $\boldsymbol{D}\boldsymbol{x} = \boldsymbol{x}$, if $\boldsymbol{x}$ is expressed as in (2.1)

Now, considering the closed formula of the nonnegative least squares solution (2.2), we can extend the approximation (3.5) to $f(\boldsymbol{y})$ so that

$$\begin{aligned}\boldsymbol{J}_k f &= \nabla_k(\boldsymbol{A})(\boldsymbol{AD})^\dagger_\lambda \begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{0}\end{bmatrix} + \boldsymbol{A}\nabla_k(\boldsymbol{AD})^\dagger_\lambda \begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{0}\end{bmatrix} \\ &= \nabla_k(\boldsymbol{A})\boldsymbol{x} + \boldsymbol{A}\nabla_k\Big(\big((\boldsymbol{AD})^H(\boldsymbol{AD}) + \lambda^2\boldsymbol{I}\big)^{-1}(\boldsymbol{AD})^H\Big)\begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{0}\end{bmatrix} \\ &= \nabla_k(\boldsymbol{A})\boldsymbol{x} - \boldsymbol{A}\big((\boldsymbol{AD})^H(\boldsymbol{AD}) + \lambda^2\boldsymbol{I}\big)^{-1}\big[\nabla_k(\boldsymbol{AD})^H(\boldsymbol{AD}) + \\ &\quad + (\boldsymbol{AD})^H\nabla_k(\boldsymbol{AD})\big]\big((\boldsymbol{AD})^H(\boldsymbol{AD}) + \lambda^2\boldsymbol{I}\big)^{-1}(\boldsymbol{AD})^H\begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{0}\end{bmatrix} \\ &= \nabla_k(\boldsymbol{A})\boldsymbol{x} - \big[\boldsymbol{A}\big((\boldsymbol{AD})^H(\boldsymbol{AD}) + \lambda^2\boldsymbol{I}\big)^{-1}\nabla_k(\boldsymbol{AD})^H(\boldsymbol{AD}) + \\ &\quad + \boldsymbol{A}(\boldsymbol{AD})^\dagger_\lambda\nabla_k(\boldsymbol{AD})\big](\boldsymbol{AD})^\dagger_\lambda\begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{0}\end{bmatrix}\end{aligned} \tag{3.6}$$

By applying Kaufman's simplification in (3.6) we obtain

$$\begin{aligned}\widetilde{\boldsymbol{J}}_k f =& \nabla_k(\boldsymbol{A})\boldsymbol{x} - \boldsymbol{A}(\boldsymbol{AD})^\dagger_\lambda\nabla_k(\boldsymbol{AD})\boldsymbol{x} = \\ =& \nabla_k(\boldsymbol{A})\boldsymbol{x} - \boldsymbol{A}(\boldsymbol{AD})^\dagger_\lambda\nabla_k(\boldsymbol{A})\boldsymbol{D}\boldsymbol{x} = \big(\boldsymbol{I} - \boldsymbol{A}(\boldsymbol{AD})^\dagger_\lambda\big)\nabla_k(\boldsymbol{A})\boldsymbol{x}\end{aligned} \tag{3.7}$$

Let us briefly comment on the computation of the pseudoinverse $(\boldsymbol{AD})^\dagger_\lambda$. Without loss of generality, we may suppose that the active set contains $n-r$ elements and thus

$$\boldsymbol{D} = \begin{bmatrix}\boldsymbol{I}_r & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}.$$

If we partition the first r and the remaining $n-r$ columns of $\boldsymbol{A}$, that is $\boldsymbol{A} = [\,\boldsymbol{A}_1 \;\; \boldsymbol{A}_2\,]$, then $\boldsymbol{AD} = [\,\boldsymbol{A}_1 \;\; \boldsymbol{0}]$. Since $(\boldsymbol{AD})^\dagger = \begin{bmatrix}\boldsymbol{A}_1^\dagger\\ \boldsymbol{0}\end{bmatrix}$ [4] , by denoting $(\boldsymbol{A}_1)^\dagger_\lambda$ to be the pseudoinverse of the $(n+r)\times r$ matrix $\begin{bmatrix}\boldsymbol{A}_1\\ \lambda\boldsymbol{I}_r\end{bmatrix}$, it follows that we can define

$$(\boldsymbol{AD})^\dagger_\lambda = \begin{bmatrix}(\boldsymbol{A}_1)^\dagger_\lambda & \boldsymbol{0}\\ \boldsymbol{0} & \lambda^{-1}\boldsymbol{I}_{n-r}\end{bmatrix}. \tag{3.8}$$

The QR decomposition can be used to compute $(\boldsymbol{A}_1)^\dagger_\lambda$. Note that $(\boldsymbol{A}_1)^\dagger_\lambda$ has smaller dimension than $\boldsymbol{A}^\dagger_\lambda$, and so computing Kaufman's simplified Jacobian with (3.7) is more convenient than using (3.5). Moreover, in order to compute $(\boldsymbol{AD})^\dagger_\lambda$ through (3.8) we need $\boldsymbol{D} = \boldsymbol{D}(\boldsymbol{x}_{true})$, which is unknown. But in [20] and [6] the authors prove that the gradient and the Newton projection methods, respectively, identify the binding constraint at $\boldsymbol{x}_{true}$ in a finite number of iterations. That is, there exists a $\overline{k}$ such that the active set $\mathcal{A}(\boldsymbol{x}_{true}) = \mathcal{A}(\boldsymbol{x}_k)$ for each $k > \overline{k}$. Since $\boldsymbol{D}$ is defined by the active set, this ensures us that

$$\boldsymbol{D}(\boldsymbol{x}_{true}) = \boldsymbol{D}(\boldsymbol{x}_k) \qquad \text{for each } k > \overline{k}.$$

In practice this means that, at each iteration of the Gauss-Newton algorithm, in order to compute the Jacobian matrix $\boldsymbol{J}_l$ we can use $\boldsymbol{D} = \boldsymbol{D}(\boldsymbol{x}_l)$.

In any case, it should be observed that in large scale problems (such as blind deconvolution), the computation of the pseudoinverse for a general matrix $\boldsymbol{A}$ is computationally very expensive. A better alternative should be to compute $(\boldsymbol{AD})^\dagger_\lambda\boldsymbol{v}$ as

the solution of problem (2.1) with data vector $\boldsymbol{v}$. Moreover, in large scale problems such as blind deconvolution, the matrix $\boldsymbol{A}$ is not explicitly constructed, but instead is accessed implicitly through the parameter vector $\boldsymbol{y}$. For this reason, in all the numerical experiments presented in Section 5, we compute by finite difference methods $\boldsymbol{J}(f(\boldsymbol{y}))$ (see (3.2)) or its approximation $\boldsymbol{J}(g(\boldsymbol{y}))$ (see (3.4)).

4. Blind Deconvolution Model Problem. To describe the blind deconvolution problem, we first need a mathematical model of the image formation process. One simple and often used model is the convolution equation,

$$b(s,t) = p(s,t) * x(s,t) + \eta(s,t)\,,$$

where $x(s,t)$ is a function representing the true image, which is convolved with a point spread function (PSF), $p(s,t)$, and after including $\eta(s,t)$, to model additive noise, we obtain the observed image, $b(s,t)$. The convolution equation assumes that the blur is spatially (i.e., shift) invariant, so it cannot be used to model more difficult problems. In addition, it does not indicate that it is often the case that the PSF can be represented in terms of only a few parameters. Although we could use a general integral equation to model the image formation process, because data are given in discrete form, we prefer to move directly to the discretized equation (1.1), where

- $\boldsymbol{b}$ is a vector representing the observed, blurred and noisy image.
- $\boldsymbol{x}$ is vector representing the unknown true image we wish to reconstruct.
- $\boldsymbol{A}(\boldsymbol{y})$ is an ill-conditioned matrix, representing the PSF, defined by a vector of parameters, $\boldsymbol{y}$. $\boldsymbol{A}$ may be sparse and/or structured. For example, if the blur is spatially invariant and periodic boundary conditions are imposed, then $\boldsymbol{A}$ has a circulant structure.
- $\boldsymbol{y}$ is a vector of parameters defining the (true) blurring operation. For example, in the case of spatially invariant blurs, $\boldsymbol{y}$ could simply be the pixel (image space) values of the PSF. Or $\boldsymbol{y}$ could be a small set of parameters that define the PSF. For example, in the case of a simple Gaussian PSF, we have

$$p(s,t) = \frac{1}{2\pi\sqrt{\delta}} \exp\left(-\frac{1}{2} \begin{bmatrix} s & t \end{bmatrix} \begin{bmatrix} \sigma_1^2 & \rho^2 \\ \rho^2 & \sigma_2^2 \end{bmatrix}^{-1} \begin{bmatrix} s \\ t \end{bmatrix} \right) \tag{4.1}$$

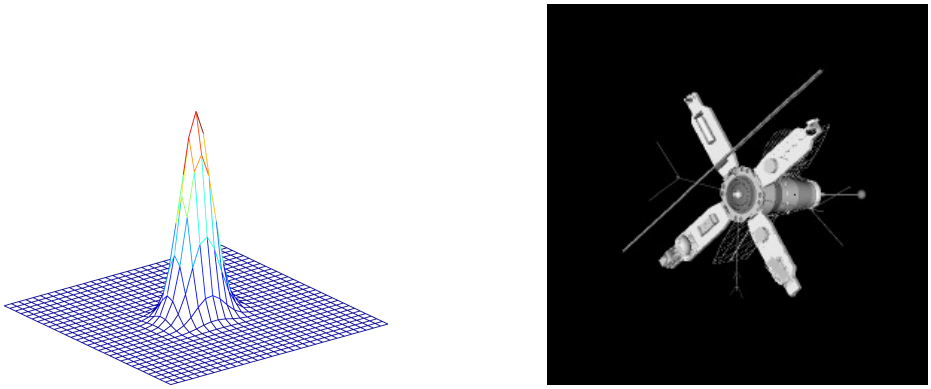
  where $\delta = \sigma_1^2\sigma_2^2 - \rho^4 > 0$. In this case the parameter vector $\boldsymbol{y}$ contains only three values, σ_1, σ_2 and ρ.
- $\boldsymbol{\eta}$ is a vector that represents unknown additive noise in the measured data. Generally $\boldsymbol{\eta}$ is a combination of background and readout noise, where the background noise is modeled as a Poisson random process with fixed Poisson parameter, and the readout noise is modeled as a Gaussian random process with zero mean and fixed variance [2, 5, 31, 32].

The aim of image deblurring algorithms is to reconstruct an approximation of the image $\boldsymbol{x}_{\text{true}}$. Because the convolution model is often assumed for the image formation process, the term *deconvolution* is typically used when $\boldsymbol{y}_{\text{true}}$ (i.e., the true blurring operator) is known, whereas *blind deconvolution* implies that $\boldsymbol{y}_{\text{true}}$ is not known.

We are interested in the blind deconvolution problem, which is modeled as a separable nonlinear inverse problem. We remark that if a good estimate of the blurring operator is known, then it may be appropriate to use a regularized total least squares model [27]. However, these techniques are not applicable for general blind deconvolution problems, and therefore we consider the nonlinear least squares formulation described earlier in the paper.

5. Numerical Examples. To motivate the constrained approach to solving the blind deconvolution problem, we first present some results obtained with the Gauss-Newton method described in section 2. We consider both unconstrained and constrained solvers and we compare the results obtained in the two cases. Finally, to explain the observed behavior, we consider a simple example in which it is possible to investigate the objective function.

5.1. Nonnegative Constraint. We begin by illustrating how imposing nonnegative constraints on $\boldsymbol{x}$ provides better results for both the linear and nonlinear variables. In particular we consider as a true object the satellite image, blurred by the Gaussian PSF (4.1) defined by the parameter $\boldsymbol{y}_{true} = (\sigma_1, \sigma_2, \rho) = (1.5, 2, 0.5)$ and corrupted by 5% white Gaussian noise. The PSF and the true object $\boldsymbol{x}$ we use are shown in Fig. 5.1. We solve the problem with the Gauss-Newton method described in section 2 with the starting point $\boldsymbol{y}_0$ equal to $(10, 12, 5)$ in all the presented experiments of this section.



(a) Gaussian PSF (b) True object, $\boldsymbol{x}_{\text{true}}$.

FIG. 5.1. *Test data: Left is a Gaussian PSF as given in equation (4.1) with* $\mathbf{y}_{\text{true}} = [\sigma_1, \sigma_2, \rho]^T$, *where* $\sigma_1 = 1.5$, $\sigma_2 = 2$, *and* $\rho = 0.5$. *Right is the true object,* $\mathbf{x}_{\text{true}}$.

We begin by using Tikhonov regularization in the deconvolution problem for the computation of $\boldsymbol{x}(\boldsymbol{y})$. That is, for each $\boldsymbol{y}$ we compute:

$$\boldsymbol{x}(\boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \|\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{y})\boldsymbol{x}\|_2^2 + \lambda^2 \|\boldsymbol{x}\|_2^2. \tag{5.1}$$

The first consideration is to determine what to use for the regularization parameter λ. Since the "optimal" value for λ depends not only on the data $\boldsymbol{b}$ but also on the matrix $\boldsymbol{A}(\boldsymbol{y})$, it makes sense to use different values for different $\boldsymbol{y}$. It would be difficult to specify these values a priori, and thus it is appropriate to attempt to use a regularization parameter choice method, such as weighted generalized cross validation (WGCV) [9].

Then we consider including a nonnegativity constraint in the deconvolution; that is, we compute $\boldsymbol{x}(\boldsymbol{y})$ as:

$$\boldsymbol{x}(\boldsymbol{y}) = \arg\min_{\boldsymbol{x} \geq 0} \|\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{y})\boldsymbol{x}\|_2^2 + \lambda^2 \|\boldsymbol{x}\|_2^2. \tag{5.2}$$

To solve this constrained deconvolution problem we use a standard gradient projection method [20, 24]. Although methods have been developed to select regularization

parameters for constrained problems directly from the data (see, e.g., [3]), they are less developed than in the unconstrained case. In our tests, the regularization parameter has been selected as the value minimizing, at each step l of the Gauss-Newton algorithm, the error:

$$\frac{\|\boldsymbol{y}_{\text{true}} - \boldsymbol{y}_l\|_2}{\|\boldsymbol{y}_{\text{true}}\|_2} \tag{5.3}$$

where $\boldsymbol{y}_l$ is the estimated vector of parameters at step l. In fact, we have observed that the value λ_l minimizing the relative error on the $\boldsymbol{x}$ vector:

$$\frac{\|\boldsymbol{x}_{\text{true}} - \boldsymbol{x}_l\|_2}{\|\boldsymbol{x}_{\text{true}}\|_2}$$

produces a larger error on the parameters $\boldsymbol{y}$. We have also observed that the values of λ_l selected by minimzing (5.3) are almost the same for all the Gauss Newton iterations; hence, we used a constant λ.

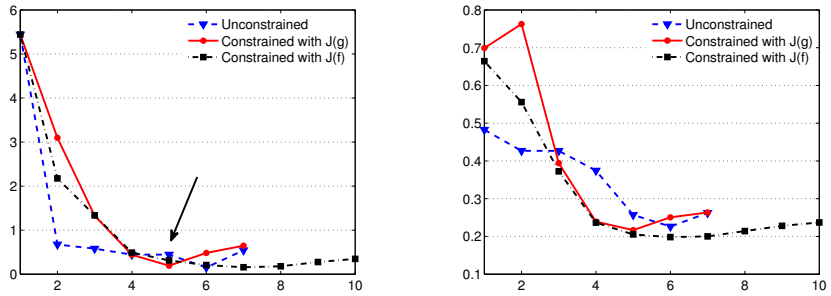


FIG. 5.2. *Plots of the relative errors during the Gauss-Newton iterations. Left is the error of the nonlinear parameter* $\mathbf{y}$*; right is the error of the image* $\mathbf{x}$*.*

In Fig. 5.2 we compare the relative errors obtained by computing, in the blind deconvolution process, $\boldsymbol{x}(\boldsymbol{y})$ as the solution of the unconstrained problem (5.1) (blue triangles) and of the constrained problem (5.2), where the results using the true Jacobian $\boldsymbol{J}(f)$ (black squares) is compared with results when $\boldsymbol{J}(f)$ is approximated with the Jacobian for the unconstrained problem, $\boldsymbol{J}(g)$ (red dots). All the Jacobians are computed by centered finite difference approximations.

From these results we see that imposing nonnegative constraints allows to better eliminate the noise, especially in the black background; see Fig. 5.3, where we display surface mesh plots of a portion of the reconstructed images, compared with the truth. From the figures, it is evident that the constrained approach gives better approximations of both the $\boldsymbol{x}$ and $\boldsymbol{y}$ parameters. The left plot of Fig. 5.2 shows that the $\boldsymbol{y}$ error curve of the Gauss-Newton method has a semiconvergent behaviour in all the three cases, but it is flatter when imposing the nonnegative constraints. When a good solution is computed, the objective function becomes flat, as shown in Fig. 5.4. To make the method automatic, we propose stopping the Gauss-Newton iterations when

$$\frac{|f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}_k)|}{f(\boldsymbol{x}_{k+1})} \leq \frac{\gamma}{2}.$$

We used as a good value for γ the level of noise in the observed data.

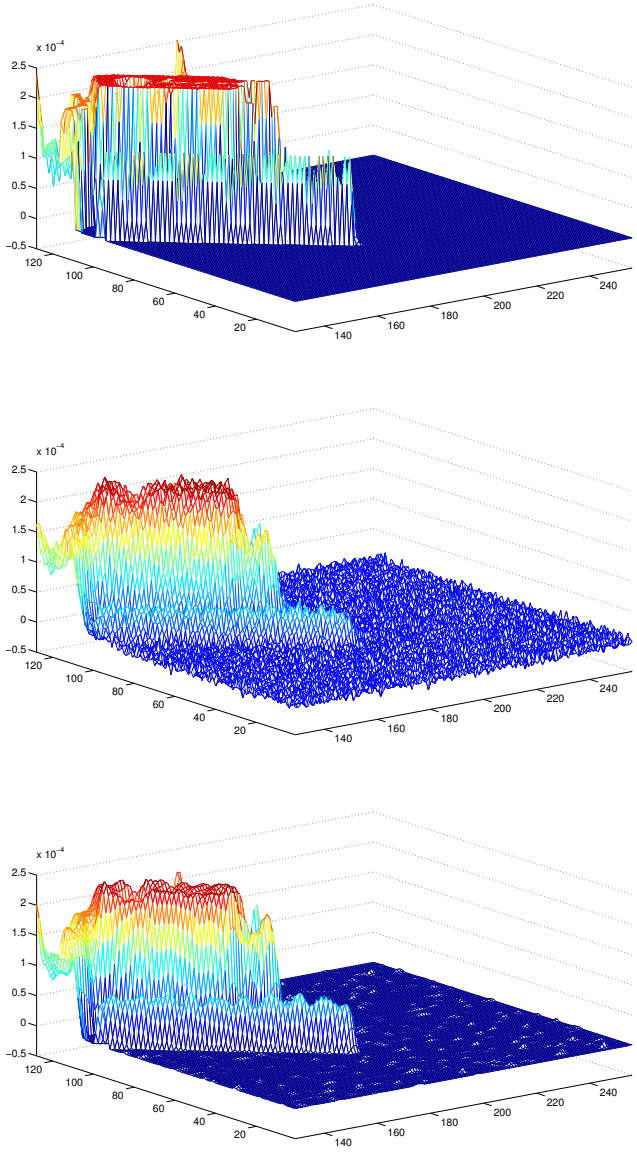


FIG. 5.3. *Surfaces of the true object and the reconstructed images* $\mathbf{x}$*: Top is the satellite image. In the second row is the image solution of the unconstrained problem. Bottom is the image obtained by imposing nonnegative constraints.*

For what concerns the Jacobian computation, the approximation of $\boldsymbol{J}(f)$ with $\boldsymbol{J}(g)$ slightly affects the best errors on $\boldsymbol{x}$ and $\boldsymbol{y}$ obtained at the fifth Gauss-Newton iteration while saving computational time.

5.2. Objective function investigations. Our aim is now to explain why the results obtained with a nonnegative model (5.2) are better than the results of the unconstrained formulation (5.1). We consider a simpler example than before, where

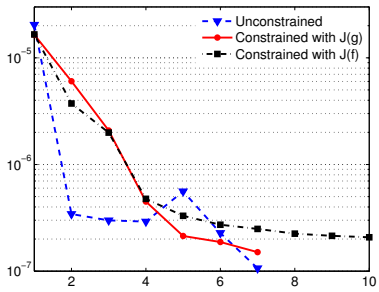


FIG. 5.4. *Objective function behaviour*

the parameters in (4.1) are $\sigma_1 = \sigma_2 = 2$, $\rho = 0$. In this case the objective function depends on only one parameter. We compare the objective functions $g(\boldsymbol{y})$ (3.3) and $f(\boldsymbol{y})$ (3.1). Recall that the only difference between $g(\boldsymbol{y})$ and $f(\boldsymbol{y})$ is the computation of the solution $\boldsymbol{x}(\boldsymbol{y})$ of the deconvolve step. In all the tests we solve the deconvolution problems (respectively (5.1) and (5.2)) as we did in the previous subsection.

We first evaluate $g(\boldsymbol{y})$ for various values of $\boldsymbol{y}$ and we obtain the plot shown in Fig. 5.5.

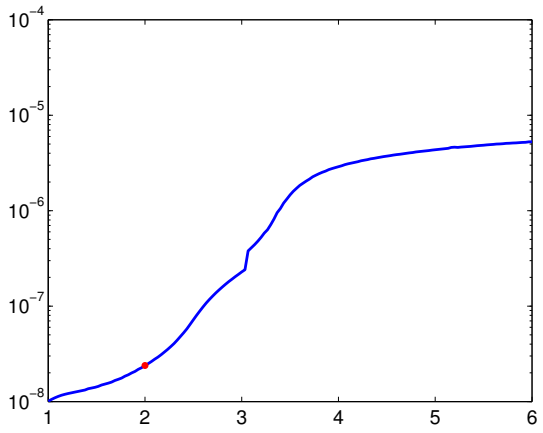


FIG. 5.5. *Plot of of the objective function* $g(\mathbf{y})$. *The regularization parameter needed to compute* $\mathbf{x}(\mathbf{y})$ *is chosen using a weighted generalized cross validation method. The red dot indicates* $f(\mathbf{y}_{\text{true}})$.

For this example we see that the objective function does not have a minimum at the desired solution $\boldsymbol{y}_{\text{true}} = 2$. We next investigate how the accuracy of computing $\boldsymbol{x}(\boldsymbol{y})$ in our deconvolve step affects the behavior of the objective function. Recall that to obtain the objective function shown in Fig. 5.5 we used Tikhonov regularization to compute $\boldsymbol{x}(\boldsymbol{y})$, using a weighted GCV scheme to choose regularization parameters. It is possible that an alternative parameter choice method will result in an objective function with a well defined global minimum. However, rather than testing the plethora of other parameter choice methods, we instead find the "optimal" λ that

results in an $\boldsymbol{x}(\boldsymbol{y})$ that minimizes the error,

$$\frac{\|\boldsymbol{x}(\boldsymbol{y}) - \boldsymbol{x}_{\text{true}}\|_2}{\|\boldsymbol{x}_{\text{true}}\|_2}\,. \tag{5.4}$$

Of course this is not possible in a realistic situation when the true object is not known, but it does provide us with an idea of what is potentially possible. Using this approach to find the "optimal" λ at each iteration, the objective function for various values of $\boldsymbol{y}$ is shown in Fig. 5.6.

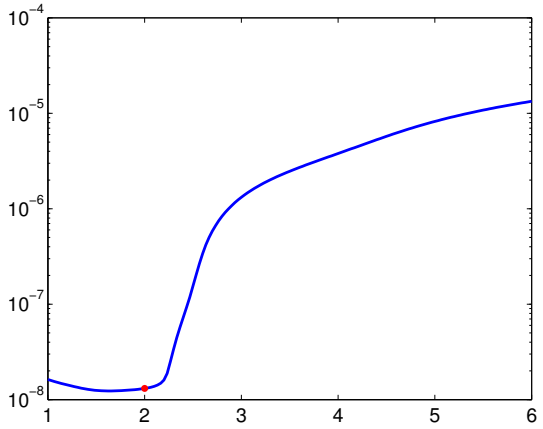


FIG. 5.6. *Plot of of the objective function* $g(\mathbf{y})$. *The regularization parameter needed to compute* $\mathbf{x}(\mathbf{y})$ *is chosen to minimize* $\|\mathbf{x}(\mathbf{y}) - \mathbf{x}_{\text{true}}\|_2/\|\mathbf{x}_{\text{true}}\|_2$. *The red dot indicates* $g(\mathbf{y}_{\text{true}})$.

In this case, the objective function has a global minimum. But unfortunately the value of $\boldsymbol{y}$ at which the objective function reaches its global minimum is not the desired value $\boldsymbol{y}_{\text{true}}$. Moreover, the objective function is very flat near the global minimum, which can make it difficult to recognize.

We could go one step further and suppose we are lucky to have a deconvolve method that always computed the exact solution, $\boldsymbol{x}(\boldsymbol{y}) = \boldsymbol{x}_{\text{true}}$. In this extremely unrealistic case, we obtain the objective function shown in Fig. 5.7.

Although it is impossible to have a deconvolution solver that always computes $\boldsymbol{x}(\boldsymbol{y}) = \boldsymbol{x}_{\text{true}}$, this, along with the case when we computed $\boldsymbol{x}(\boldsymbol{y})$ with an optimal regularization parameter, does suggest that we can help the optimization method by using a deconvolution solver that computes more physically realistic approximations of $\boldsymbol{x}_{\text{true}}$. One simple example to do this is to include a nonnegativity constraint within the deconvolution solver and compute the objective function $f(\boldsymbol{y})$ instead of $g(\boldsymbol{y})$.

To test the constrained solver, we begin by using the optimal regularization parameters for the unconstrained problem, and compare them with more appropriate values for the constrained problem, found through experimentation[1]. The results are shown in Fig. 5.8.

We observe that the nonnegativity constraint, along with well chosen regularization parameters, is very effective in producing an objective function $f(\boldsymbol{y})$ whose global

[1]We started with the λ values of the unconstrained problem, and reduced them systematically, eventually by a factor of 200, until we observed values that produced an objective function whose global minimum occurs near $\boldsymbol{y}_{\text{true}}$.

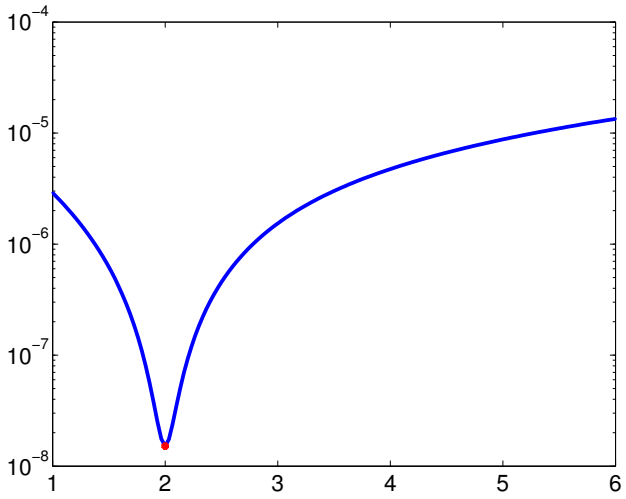


FIG. 5.7. *Plot of of the objective function* $g(\mathbf{y})$. *Here we use* $\mathbf{x}(\mathbf{y}) = \mathbf{x}_{\text{true}}$. *The red dot indicates* $f(\mathbf{y}_{\text{true}})$.

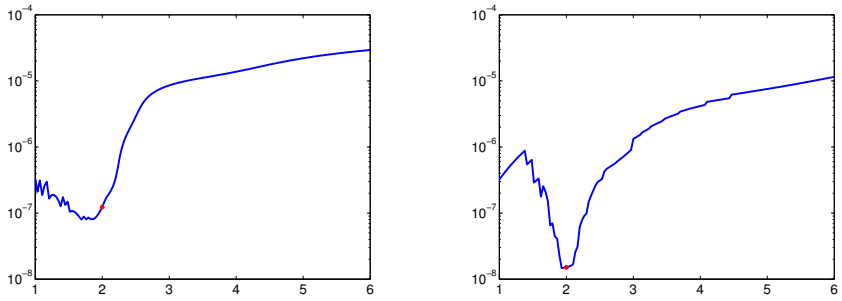


FIG. 5.8. *Plot of of the objective function* $f(\mathbf{y})$, *where* $\mathbf{x}(\mathbf{y})$ *is obtained by solving, to high accuracy, the nonnegative constrained Tikhonov regularized deconvolution problem. The red dot indicates* $f(\mathbf{y}_{\text{true}})$. *The plot of* $f(\mathbf{y})$ *on the left was obtained by using* λ_{opt}, *the optimal* λ*'s for the unconstrained problem (see Fig. 5.6), and the plot on the right was obtained using* $\lambda_{opt}/200$.

minimum is near $f(\boldsymbol{y}_{\text{true}})$. However, we note that $f(\boldsymbol{y})$ has several local minima, and the optimization algorithms can easily become trapped in one of these, believing it has reached a global optimal solution.

5.3. IMFILT: An Optimization Method to Avoid Local Minima. As observed in the previous subsection, even if we are able to solve the deconvolution problem well, there is still the issue of the optimization algorithm becoming trapped in local minima. However, if the objective function is fairly well behaved, such as in the case of the right plot in Fig. 5.8, then the *implicit filtering* optimization methods developed by Kelley [21] can be effective. The implicit filtering methods were designed specifically for objective functions that have the form

$$f(\boldsymbol{y}) = f_s(\boldsymbol{y}) + f_e(\boldsymbol{y}) \tag{5.5}$$

where f_s is a smooth function, and f_e is a perturbation function that models the small oscillations that could cause f to have many local minima.

The basic idea of implicit filtering is based on using finite difference approximations to compute derivatives. For example, a forward difference approximation of the gradient has the form

$$[\nabla_h f(\boldsymbol{y})]_i = \frac{f(\boldsymbol{y} + h\boldsymbol{e}_i) - f(\boldsymbol{y})}{h},$$

where $\boldsymbol{e}_i$ is the ith unit vector, and h is a, usually small, scalar. In the case of nonlinear least squares, a similar approach can be used to approximate the Jacobian. These approximations for the derivatives can then be used in optimization methods, such as gradient descent or Gauss-Newton, in place of the analytic derivative. Implicit filtering methods exploit the fact that the value for h can have an effect on convergence. In particular, implicit filtering introduces another level of iteration that repeatedly reduces h, e.g., $h_l = 2^{-(l+s)}$ where l is the outer iteration index, and $s \geq 0$ can be used to indicate a starting scale. At each outer iteration, the optimization method is used to solve the minimization problem with the finite difference approximations corresponding to the current value of h_l.

To illustrate its potential for the blind deconvolution problem, we obtained the recent version of Kelley's MATLAB software [21], and applied it to the situation shown in the right plot of Fig. 5.8. We first solved the problem using a standard damped Gauss-Newton method. Using the the initial guess $\boldsymbol{y}_0 = 1.5$ the damped Gauss-Newton iteration gets trapped in a local minimum at $\boldsymbol{y} = 1.489508423766336$. Moreover, using the different initial guess $\boldsymbol{y}_0 = 1.75$, the damped Gauss-Newton iteration gets trapped in a different local minimum, $\boldsymbol{y} = 1.745316540312289$.

However, using the default options in Kelley's implicit filtering method, the iteration converges to a very good approximation of $\boldsymbol{y}_{\text{true}} = 2$ for both initial guesses. Specifically, with the initial guess $\boldsymbol{y}_0 = 1.5$ the implicit filtering iteration converges to $\boldsymbol{y} = 2.010364508143452$, and with $\boldsymbol{y}_0 = 1.75$ it converges to $\boldsymbol{y} = 1.997891906915320$.

6. Concluding Remarks. In this paper we have illustrated that a substantial amount of uncertainty arises when solving blind deconvolution problems, and more generally when solving separable ill-posed inverse problems. This uncertainty is caused not only by noise and measurement errors, but also by imprecise regularization parameters and linear solvers, and it suggests that it is perhaps too ambitious to hope that one can design an objective function that is convex with a unique global minimum. Moreover, typical optimization algorithms converge to local minima; hence they can compute undesired solutions when the objective function is not convex (as in Fig. 5.5) or it is flat near the global minimum (as in Fig. 5.6). In this paper we showed that imposing constraints on some parameters, i.e. solving the ill-posed problem with greater accuracy, produces an objective function more suitable for the common optimization algorithms, such as Gauss-Newton.

Moreover we argue that, for an efficient solution of separable ill-posed problems, the goal should be to design tools that achieve an objective of the form given by equation (5.5), so that a robust implicit filtering scheme can be used to solve the optimization problem. Specifically, to effectively solve a separable ill-posed inverse problem, one should:

- Use robust and efficient methods to solve the linear inverse problems, with the goal to compute physically accurate approximations of $\boldsymbol{x}_{\text{true}}$ for a current set of parameters $\boldsymbol{y}_l$ at each nonlinear iteration. Because these methods need to be used within a nonlinear solver, it is essential that they automatically

adapt to the information available at the current iteration, and that they require little or no input from the user.

- Although a substantial amount of work has been done to develop methods to automatically choose regularization parameters for unconstrained deconvolution problems, relatively little has been done for the constrained problem. Research on this topic could make a significant impact for separable nonlinear inverse problems.

We have illustrated, using an example from blind deconvolution, if one can achieve these goals, then implicit filtering methods [21], which have been developed to solve optimization problems with objective functions of the form given by equation (5.5), can be effective for ill-posed inverse problems.

REFERENCES

[1] H. Andrews and B. Hunt, *Digital Image Restoration*, Prentice-Hall, Englewood Cliffs, NJ, 1977.

[2] J. Bardsley and C. R. Vogel, *A nonnegativly constrained convex programming method for image reconstruction*, SIAM J. Sci. Comput., 25 (2004), pp. 1326–1343.

[3] J. M. Bardsley and J. Goldes, *Regularization parameter selection methods for ill-posed Poisson maximum likelihood estimation*, Inverse Problems, 25 (2009), pp. doi: 10.1088/0266–5611/25/9/095005.

[4] J. M. Bardsley, J. Merikoski, and R. Vio, *The stabilizing properties of nonnegativity constraints in least-squares image reconstruction*, IJPAM, 43 (2008), pp. 95–109.

[5] J. M. Bardsley and J. G. Nagy, *Covariance-preconditioned iterative methods for nonnegatively constrained astronomical imaging*, SIAM J. Matrix Anal. Appl., 27 (2006), pp. 1184–1197.

[6] D. P. Bertsekas, *Projected newton methods for optimization problems with simple constraints*, SIAM J. Control and Optimization, 20 (1982), pp. 221–246.

[7] J. Chung, E. Haber, and J. Nagy, *Numerical methods for coupled super-resolution*, Inverse Problems, 22 (2006), pp. 1261–1272.

[8] J. Chung and J. Nagy, *An efficient iterative approach for large-scale separable nonlinear inverse problems*, SIAM J. Sci. Comput., 31 (2010), pp. 4654–4674.

[9] J. Chung, J. G. Nagy, and D. P. O'Leary, *A weighted GCV method for Lanczos hybrid regularization*, Elec. Trans. Num. Anal., 28 (2008), pp. 149–167.

[10] J. Chung, P. Sternberg, and C. Yang, *High performance 3-D image reconstruction for molecular structure determination*, International Journal of High Performance Computing Applications, 24 (2010), pp. 117–135.

[11] J. Frank, *Three-Dimensional Electron Microscopy of Macromolecular Assemblies*, Oxford University Press, New York, 2006.

[12] G. H. Golub and V. Pereyra, *The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate*, SIAM J. Numer. Anal., 10 (1973), pp. 413–432.

[13] ———, *Separable nonlinear least squares: the variable projection method and its applications*, Inverse Problems, 19 (2003), pp. R1–R26.

[14] E. Haber, U. M. Ascher, and D. Oldenburg, *On optimization techniques for solving nonlinear inverse problems*, Inverse Problems, 16 (2000), pp. 1263–1280.

[15] P. C. Hansen, J. G. Nagy, and D. P. O'Leary, *Deblurring Images: Matrices, Spectra and Filtering*, SIAM, Philadelphia, PA, 2006.

[16] M. Hohn, G. Tang, G. Goodyear, P. R. Baldwin, Z. Huang, P. A. Penczek, C. Yang, R. M. Glaeser, P. D. Adams, and S. J. Ludtke, *SPARX, a new environment for Cryo-EM image processing*, J. Structural Biology, 157 (2007), pp. 47–55.

[17] M. G. Kang and S. Chaudhuri, *Super-resolution image reconstruction*, IEEE Signal Processing Magazine, 20 (2003), pp. 19–20.

[18] L. Kaufman, *A variable projection method for solving separable nonlinear least squares problems*, BIT, 15 (1975), pp. 49–57.

[19] L. Kaufman and V. Pereyra, *A method for separable nonlinear least squares problems with separable equality constraints*, SIAM J. Numer. Anal., 15 (1978), pp. 12–20.

[20] C. T. Kelley, *Iterative Methods for Optimization*, SIAM, Philadelphia, 1999.

[21] ———, *Implicit Filtering*, SIAM, Philadelphia, 2011. Software and manual can be found at http://www4.ncsu.edu/~ctk/imfil.html.

[22] R. L. Lagendijk and J. Biemond, *Iterative Identification and Restoration of Images*, Kluwer Academic Publishers, Boston/Dordrecht/London, 1991.
[23] R. Marabini, G. T. Herman, and J. M. Carazo, *3D reconstruction in electron microscopy using ART with smooth spherically symmetric volume elements (blobs)*, Ultramicroscopy, 72 (1998), pp. 53–65.
[24] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, New York, 1999.
[25] M. R. Osborne, *Separable least squares, variable projection, and the Gauss-Newton algorithm*, Elec. Trans. Numer. Anal., 28 (2007), pp. 1–15.
[26] P. A. Penczek, M. Radermacher, and J. Frank, *Three-dimensional reconstruction of single particles embedded in ice*, Ultramicroscopy, 40 (1992), pp. 33–53.
[27] A. Pruessner and D. P. O'Leary, *Blind deconvolution using a regularized structured total least norm algorithm*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 1018–1037.
[28] A. Ruhe and P. Wedin, *Algorithms for separable nonlinear least squares problems*, SIAM Review, 22 (1980), pp. 318–337.
[29] S. D. Saban, M. Silvestry, G. R. Nemerow, and P. L. Stewart, *Visualization of α-helices in a 6-Ångstrom resolution cryoelectron microscopy structure of adenovirus allows refinement of capsid protein assignments*, J. Virol, 80 (2006), pp. 49–59.
[30] D. M. Sima and S. V. Huffel, *Separable nonlinear least squares fitting with linear bound constraints and its application in magnetic resonance spectroscopy data quantification*, J. Comput. Appl. Math., 203 (2007), pp. 264–278.
[31] D. L. Snyder, A. M. Hammoud, and R. L. White, *Image recovery from data acquired with a charge-coupled-device camera*, J. Opt. Soc. Am. A, 10 (1993), pp. 1014–1023.
[32] D. L. Snyder, C. W. Helstrom, A. D. Lanterman, M. Faisal, and R. L. White, *Compensation for readout noise in ccd images*, J. Opt. Soc. Am. A, 12 (1995), pp. 272–283.