BIT Numerical Mathematics (2005) 45: 329–340
DOI: 10.1007/s10543-005-7122-y

MINIMIZATION OF LINEAR FUNCTIONALS DEFINED ON SOLUTIONS OF LARGE-SCALE DISCRETE ILL-POSED PROBLEMS ⋆

L. ELDÉN[1], P. C. HANSEN[2] and M. ROJAS[3,⋆⋆]

[1] *Department of Mathematics, Linköping University, S-581 83 Linköping, Sweden. email: laeld@math.liu.se*

[2] *Informatics and Mathematical Modelling, Technical University of Denmark, Building 321, DK-2800 Kgs. Lyngby, Denmark. email: pch@imm.dtu.dk*

[3] *Department of Mathematics, Wake Forest University, P.O. Box 7388, Winston-Salem, NC 27109, USA. email: mrojas@mthcsc.wfu.edu*

Abstract.

The minimization of linear functionals defined on the solutions of discrete ill-posed problems arises, e.g., in the computation of confidence intervals for these solutions. In 1990, Eldén proposed an algorithm for this minimization problem based on a parametric programming reformulation involving the solution of a sequence of trust-region problems, and using matrix factorizations. In this paper, we describe MLFIP, a large-scale version of this algorithm where a limited-memory trust-region solver is used on the subproblems. We illustrate the use of our algorithm in connection with an inverse heat conduction problem.

AMS subject classification (2000): 65F22.

Key words: discrete ill-posed problems, confidence intervals, large-scale algorithms, trust regions.

1 Introduction.

This work is concerned with the minimization of a linear functional defined on the solution of a discrete ill-posed problem. The mathematical formulation of this problem takes the following form: let A be an ill-conditioned coefficient matrix of dimensions $m \times n$, let b be the corresponding right-hand side, and let w be an arbitrary vector of unit length ($\|w\|_2 = 1$), then compute the solution x_w to the problem

$$\min w^{\mathrm{T}} x \quad \text{subject to} \quad \|Ax - b\|_2 \le \varepsilon, \quad \|x - d\|_2 \le \delta. \tag{1.1}$$

Here ε and δ are two positive constants that are used to stabilize the computation of the solution vector x. The vector d contains a priori knowledge of the

⋆ Received October 2004. Accepted January 2005. Communicated by Lothar Reichel.
⋆⋆ This work was partially supported by grant no. 9901581 from the Danish Natural Science Research Foundation, and by the Science Research Fund of Wake Forest University.

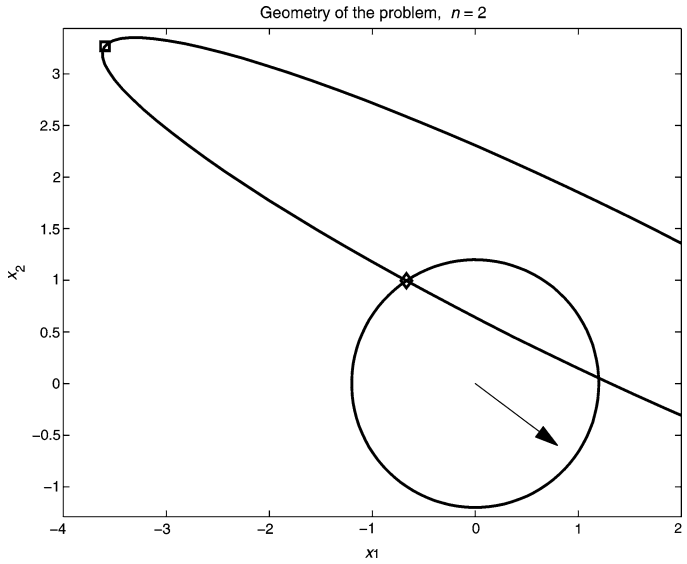

Figure 1.1: The circle and the ellipsoid are defined by $\|x - d\|_2 = \delta$ and $\|Ax - b\|_2 = \varepsilon$, respectively; here $d = 0$ and the ellipsoid is centered at $(1, 1)^{\mathrm{T}}$. The arrow illustrates the vector $w = (0.8, -0.6)^{\mathrm{T}}$. The solution to the problem $\min w^{\mathrm{T}}x$ s.t. $\|Ax - b\|_2 \leq \varepsilon$ is located at the square – far from the origin – while the solution x_w to (1.1) is located at the diamond.

solution, and may be set to the zero vector. Problems of the form (1.1) arise in the numerical treatment of inverse problems. One important application is the computation of confidence intervals for the solution of inverse problems [12]. Other applications include the computation of windowed or cumulative profiles of deconvolved data [13].

A related problem is that of computing stabilized/regularized solutions to discrete ill-posed problems, e.g., by solving the problem

$$\min \|Ax - b\|_2 \quad \text{subject to} \quad \|x - d\|_2 \leq \delta. \tag{1.2}$$

Here, δ acts as a regularization parameter that controls the "size" of the solution. The solution to (1.2) is identical to the Tikhonov solution, formally given by

$$x_\lambda = (A^{\mathrm{T}}A + \lambda^2 I)^{-1}(A^{\mathrm{T}}b + \lambda^2 d), \tag{1.3}$$

for a particular value of the Lagrange parameter λ.

Looking at the two problems (1.1) and (1.2), it may appear at first that one constraint – namely $\|Ax - b\|_2 \leq \varepsilon$ – would be enough to stabilize the first problem. However, this is not so; without the constraint $\|x-d\|_2 \leq \delta$ the vector x can have a large norm, and this happens when w points in a direction for which the ellipsoid $\|Ax - b\|_2 = \varepsilon$ is very elongated. The purpose of the constraint $\|x - d\|_2 \leq \delta$ is thus to ensure that this does not happen – at the expense of the possible non-existence of a solution (depending on the choice of δ and d). Figure 1.1 illustrates these points.

Some algorithms for solving (1.1) are presented in [4, 12, 13]. Our work builds on an algorithm developed by Eldén [4], based on the reformulation of (1.1) as a parametric programming problem involving the solution of a sequence of quadratically constrained least squares (trust-region) problems.

We have developed a framework where the core (trust-region) problems can be solved, in principle, by any trust-region method which is used in a "black-box" fashion. However, as we are considering large-scale problems where matrix factorizations are prohibitive, we are particularly interested in "matrix-free" trust-region solvers that only require the action of the matrices A and A^{T} on a vector, given in the form of subroutines for matrix–vector multiplication. Matrix-free methods for the large-scale trust-region subproblem include [6–8, 14, 15, 19]. All of these methods, except [6] and [7], have fixed storage requirements and [7, 8, 14, 15] take into account a special case usually called the hard case.

In the current implementation of the method MLFIP, we used a MATLAB version [16] of the limited-memory algorithm LSTRS by Rojas et al. [15] to solve the trust-region subproblems. Besides the features already mentioned, this method can efficiently handle the high-degree singularities arising in discrete ill-posed problems [17]. Results in [16] showed some advantages of using LSTRS versus [7, 8, 14] for regularization problems. Other methods that have been reported to perform well on discrete ill-posed problems and that could be used in MLFIP include [6] and the modification in [1], both based on Lanczos bidiagonalization and Gauss quadrature. Note, however, that these methods are not limited-memory techniques.

Another important issue in our work is the design of a robust implementation of the algorithm, suited for general use (e.g., in connection with the Regularization Tools package [9]). We want to be able to check the existence of a solution to a given problem, and we want to guarantee that our implementation always computes a solution if it exists. The resulting method is the algorithm MLFIP described in this work.

This paper is organized as follows. In Section 2 we briefly study the properties of the solution of (1.1) from the regularization point of view. In Section 3 we summarize the basic algorithm from [4]. The core of this algorithm is the computation of the smallest root of a special nonlinear function, and in Section 4 we describe our algorithm for this problem. In Section 5 we discuss some implementation issues of the algorithm. Finally, in Section 6 we demonstrate the use of MLFIP in connection with the sideways heat equation problem.

2 Properties of the solution.

In many applications where problem (1.1) is encountered – e.g., when computing confidence intervals for regularized solutions – the main goal is to compute the quantity $w^{\mathrm{T}}x$. Still, x is always computed as a by-product, and the purpose of this section is to discuss certain properties of this x.

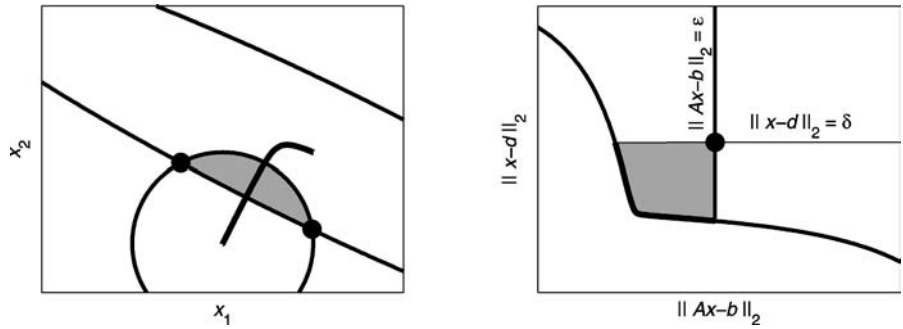

Figure 2.1: The gray region and the black circles represent the sets $\mathcal{S}$ and $\mathcal{S}^*$, respectively, while the thick line represent all the Tikhonov solutions.

Consider first the left plot in Figure 2.1, which shows the geometry of the problem for $n = 2$. The gray region is the set $\mathcal{S}$ of "feasible points" given by

$$\mathcal{S} = \{x \mid \|Ax - b\|_2 \leq \varepsilon \wedge \|x - d\|_2 \leq \delta\} \tag{2.1}$$

while the two black circles represent the set

$$\mathcal{S}^* = \{x \mid \|Ax - b\|_2 = \varepsilon \wedge \|x - d\|_2 = \delta\}. \tag{2.2}$$

Moreover, the thick solid line represents all the Tikhonov solutions x_λ given by (1.3) for $0 \leq \lambda < \infty$. Any x that solves (1.1) lies on the boundary of $\mathcal{S}$, and coincides with a Tikhonov solution only for two particular instances of the vector w. For all other w, the corresponding x differs from a Tikhonov solution – hence it is not "optimal" in the sense of balancing the norms $\|x - d\|_2$ and $\|Ax - b\|_2$, and it is not necessarily smooth in the sense of being dominated by the principal singular vectors.

Next consider the right plot in Figure 2.1 which shows a generic L-curve, i.e., a log–log plot of the norm of the Tikhonov solution $\|x_\lambda\|_2$ versus the corresponding residual norm $\|Ax_\lambda - b\|_2$. It is well known (see, e.g., Section 4.6 in [10]) that for an arbitrary $x \in \mathbb{R}^n$, the point $(\|Ax - b\|_2, \|x\|_2)$ must lie on or above the L-curve. The thin vertical and horizontal lines represent vectors x for which $\|Ax - b\|_2 = \varepsilon$ and $\|x - d\|_2 = \delta$, respectively.

Moreover, the gray region represents all the solutions in the set $\mathcal{S}$, while all the points that represent the solutions in the set $\mathcal{S}^*$ coincide in a single point (the black circle). We note that all solutions x to (1.1) lie on the upper or right boundary of the gray region that represents the set $\mathcal{S}$. Again, this illustrates the fact that the solutions x to (1.1) differ from the Tikhonov solutions x_λ.

To illustrate this point further, we consider the test problem shaw from [9]. Let x^{exact} denote the exact solution, and let $b = Ax^{\text{exact}} + e$ where the elements of the perturbation e are from a Gaussian distribution with zero mean and standard deviation such that $\|e\|_2/\|b\|_2 = 10^{-2}$. We choose $d = 0$, $\delta = \|x^{\text{exact}}\|_2$ and $\varepsilon = \|e\|_2$. The vector w is the normalized ith basis vector ψ_i in the discrete cosine transform for $i = 4$, 8, 12 and 16. The corresponding four solutions are shown

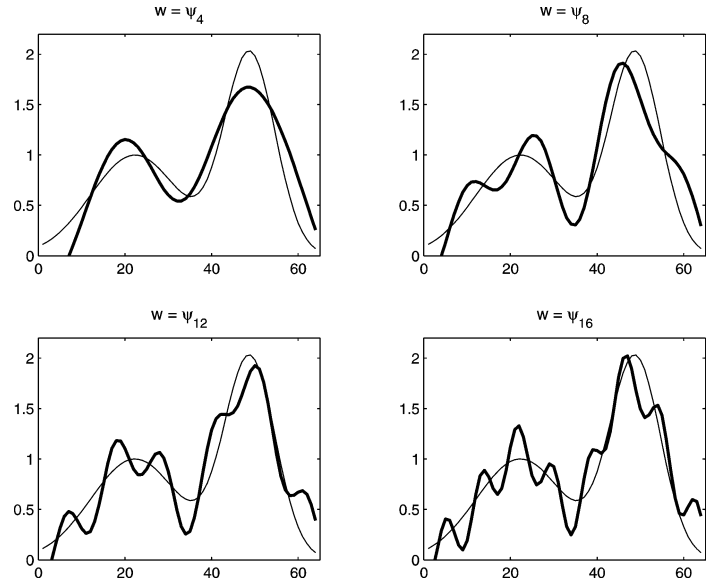

Figure 2.2: The exact solution (thin line) and the solution x_w to (1.1) for the shaw test problem and with four different choices of w.

in Figure 2.2. Clearly, x can have a highly oscillatory component, depending on the choice of w.

The conclusion is that the solution x to (1.1), albeit regularized, cannot be expected to be smooth in the same manner as the Tikhonov solutions.

3 The basic algorithm.

As shown in [4], problem (1.1) can be reformulated in such a way that the core computational problem is a parametric programming problem, which is easier to handle numerically than the original problem. According to Theorem 2.5 in [4], the minimizer $\phi = w^{\mathrm{T}}x$ to (1.1) is the *smallest* solution of the nonlinear equation

$$L(\phi) = \varepsilon^2, \tag{3.1}$$

where the function $L(\phi)$ is defined as

$$L(\phi) = \min_{x \in \mathcal{T}(\phi)} \|Ax - b\|_2, \qquad \mathcal{T}(\phi) = \{x \mid \|x - d\|_2 \le \delta,\ w^{\mathrm{T}}x = \phi\}. \tag{3.2}$$

Hence, in order to solve the original problem (1.1) we use a root finder to compute the smallest root of $L(\phi) - \varepsilon^2$. Note that the computation of the function values of $L(\phi)$ requires the solution of the parametric programming problem in (3.2).

This problem, in turn, is solved by first removing the linear constraint $w^{\mathrm{T}}x = \phi$ and then solving the resulting quadratically constrained least squares problem. Following [4], we introduce the orthogonal matrix V whose first column is w (recall that w has unit length):

$$V = (w, \overline{V})$$

and we notice that V can be chosen as a single Householder transformation. Introducing the quantities

$$\begin{aligned} \overline{A} &= A\overline{V}, \\ \overline{b} &= b - \phi A w, \\ \bar{d} &= \overline{V}^{\mathrm{T}} d, \\ \overline{\delta}^2 &= \delta^2 - (\phi - w^{\mathrm{T}} d)^2, \end{aligned}$$

the solution to the problem in (3.2) is given by

$$x = V \begin{pmatrix} \phi \\ \bar{y} \end{pmatrix} = \phi w + \overline{V}\bar{y},$$

where the vector $\bar{y}$ solves the following least squares problem with a quadratic constraint:

$$\min \|\overline{A}\bar{y} - \overline{b}\|_2 \quad \text{subject to} \quad \|\bar{y} - \bar{d}\|_2 \leq \overline{\delta}. \tag{3.3}$$

Therefore, the evaluation of $L(\phi)$ is equivalent to solving the trust-region subproblem (3.3).

4 Computing the smallest root.

Assuming that we can evaluate the function $L(\phi)$ accurately, we need a robust algorithm for computing the smallest root of the nonlinear equation (3.1), including a reliable test for the existence of such a root. The following theorem is central.

THEOREM 4.1. *The function $L(\phi)$ given by* (3.2) *is defined for $\phi_* \leq \phi \leq \phi^*$, where*

$$\phi_* = w^{\mathrm{T}} d - \delta, \qquad \phi^* = w^{\mathrm{T}} d + \delta.$$

Moreover, $L(\phi)$ is continuous and convex in this interval.

PROOF. The function L is well defined as long as $\delta^2 - (\phi - w^{\mathrm{T}} d)^2$ is nonnegative. It is straightforward to see that this holds only for $\phi \in [w^{\mathrm{T}} d - \delta, w^{\mathrm{T}} d + \delta]$. The continuity of L was proved in [4]. We will now show that this function is convex. First, assume that A has full column rank so that the ellipsoid $\{x \mid \|Ax - b\|_2 \leq \varepsilon\}$ is strictly convex. Take two points, ϕ_1 and ϕ_2, in the interval $[\phi_*, \phi^*]$, and let the corresponding, unique minimizers of $L(\phi)$ be denoted by x_1 and x_2. Define

$$\phi_\tau = (1 - \tau)\phi_1 + \tau\phi_2,$$

for some $\tau \in [0, 1]$. Then

$$\bar{x} := (1 - \tau)x_1 + \tau x_2 \in \mathcal{T}(\phi_\tau),$$

since the sphere is convex and $w^{\mathrm{T}}x$ is linear in x. It follows that

$$\begin{aligned} L(\phi_\tau) &= \min_{x\in\mathcal{T}(\phi_\tau)} \|Ax-b\|_2 \\ &\le \|A\bar{x}-b\|_2 \\ &\le (1-\tau)\|Ax_1-b\|_2+\tau\|Ax_2-b\|_2 \\ &= (1-\tau)L(\phi_1)+\tau L(\phi_2). \end{aligned}$$

If A is rank deficient and w is not orthogonal to the null-space of A, then the minimizers are unique and the above argument still holds. Finally, if w is orthogonal to the null-space, then the minimizers are not unique. However, the above proof can be applied to any pair of minimizers. Thus we have proved that $L(\phi)$ is convex. □

It follows from this theorem that, depending on ε, there is either zero, one or two solutions to the equation $L(\phi)=\varepsilon^2$. Geometrically, these three situations correspond to the ball and the ellipsoid not intersecting each other, being tangent to each other at a single point, and intersecting each other, respectively. The case where the ball lies exclusively within the ellipsoid can be taken care of initially, as discussed in Section 5. Hence, we are guaranteed that $L(\phi_*)>\varepsilon^2$.

Assuming that there are one or two solutions, we can always approach the smallest root by starting the secant method at ϕ_* because the function $L(\phi)$ is

```
Secant initialization: φ^(−1) = φ_* and φ^(0) = φ_* + 0.1δ.
If L(φ^(0)) < ε^2
    φ = robust_solver(L(φ) − ε^2, φ_*, φ^(0)); return
For k = 1, 2, ...
    compute new secant iterate φ^(k);
    if L(φ^(k)) < ε^2
        φ = robust_solver(L(φ) − ε^2, φ^(k), φ^(k−1)); return;
    if φ^(k) < φ^(k−1) or φ^(k) > φ^* or L(φ^(k)) > L(φ^(k−1))
        compute φ_0 = argmin L(φ);
        if L(φ_0) > ε^2
            return with confidence that no solution exists;
        else
            φ = robust_solver(L(φ) − ε^2, φ_*, φ^(k−1)); return;
    else
        secant update: φ^(k−1) = φ^(k−2).
    if |φ^(k) − φ^(k−1)| < τ|φ^(k)|
        φ = φ^(k); return.
```

Figure 4.1: Safeguarded secant algorithm for computing the smallest root of $L(\phi)-\varepsilon^2$. Here, τ is a user-specified threshold.

convex. If the approach fails to find a root, it is likely that no root exists. We choose the starting iterates for the secant method to be ϕ_* and $\phi_* + 0.1\delta$. If $L(\phi_* + 0.1\delta) < \varepsilon^2$ then we have a bracket for the smallest root, and we can use this bracket to compute the root; otherwise we start the secant method, and stop when the step is small enough.

During the iterations we may encounter an iterate ϕ such that $L(\phi) < \varepsilon^2$; then again we have a bracket for the smallest root.

If the new iterate is outside the interval $[\phi_*, \phi^*]$, if the secant step is backwards, or if we encounter an increasing value of $L(\phi)$, then it is likely that no root exists. In this case we take the effort to compute the minimum of $L(\phi)$; if this minimum is greater than ε^2 then we are guaranteed that no solution exists, otherwise we once again have a bracket for the smallest root.

We summarize our *safeguarded* algorithm for computing the smallest root of $L(\phi) - \varepsilon^2$ in Figure 4.1, where "robust_solver$(L(\phi) - \varepsilon^2, a, b)$" means that we call a robust solver to find the root in the bracket determined by a and b.

5 Implementation issues.

In this section, we discuss some issues concerning the implementation of the algorithm MLFIP, described in Sections 3 and 4. First of all, we mention that the special case where x satisfies $\|Ax - b\|_2 < \varepsilon$ and $\|x - d\|_2 = \delta$ can be checked stably and efficiently. The solution to the problem $\min w^{\mathrm{T}} x$ s.t. $\|x - d\|_2 \leq \delta$ occurs at the boundary, with $x = d - \delta w$; if this x satisfies $\|Ax - b\|_2 < \varepsilon$ then we are done. Hence, our algorithm initially makes this inexpensive check.

A second issue is how to solve the trust-region subproblems required to evaluate the function $L(\phi)$ in MLFIP. In our implementation, the subproblems are solved by means of the algorithm LSTRS [15, 16]. This algorithm only requires multiplications with the matrix $\overline{A} = A\overline{V}$ and its transpose. Note that matrix–vector products with $\overline{V}$ require only $\mathcal{O}(n)$ flops, since $V = (w, \overline{V})$ is a single Householder transformation. Moreover, LSTRS has fixed storage requirements and is able to handle the singularities of ill-posed problems [17].

Our initial experiments showed that the evaluation of $L(\phi)$ in MLFIP requires a *boundary* solution for the trust-region subproblems. Therefore, when calling LSTRS, it is necessary to give priority to boundary solutions over other kinds of solutions. Boundary solutions are favored by a proper choice of the tolerances in the stopping criteria, i.e., tighter tolerances for the undesired kind of solutions and looser for boundary solutions. The LSTRS Matlab software provides a straightforward way of specifying these tolerances.

At each iteration of LSTRS, the solution of a large eigenvalue problem is required. In [16], the eigenproblems are solved by the Implicitly Restarted Lanczos Method (IRLM) [18] through the routine eigs which is the Matlab interface to ARPACK [11]. For ill-posed problems, most of the matrices arising in LSTRS will have a large cluster of their smallest eigenvalues very close to zero. Thus, matrix–vector products with such matrices will annihilate components in the direction of eigenvector associated with those small eigenvalues and special care must be taken in this case. This is a concern since LSTRS requires precisely some of the

smallest eigenvalues and since the IRLM relies on matrix–vector multiplications. To overcome this situation, a Tchebyshev Spectral Transformation is used to compute the small eigenvalues. Such transformation has been successfully used in other contexts. We refer the reader to [17] for more details.

Through the use of the safeguarded root finder outlined in the previous section, our implementation is *robust* in the sense that it is able to check for the existence of a solution. If the minimum of the function $L(\phi)$ is positive then we are certain that there is no solution to problem (1.1). For the robust solver, we use Matlab's fzero function whenever a bracket for the desired root is available, and we use Matlab's fminbnd function to compute the minimum of $L(\phi)$, when required.

6 Numerical examples.

We first illustrate the use of our algorithm by applying it to a problem in inverse heat conduction from [5]. The sideways heat equation in a quarter plane geometry can be formulated as a Volterra integral equation of the first kind [2],

$$\int_0^t k_\kappa(t-\tau)f(\tau)\mathrm{d}\tau = g(t), \quad 0 \le t \le 1,$$

where the functions $f(t)$ and $g(t)$ represent the temperature as functions of time on an inaccessible and an accessible side of an object, respectively. The kernel function is given by

$$k_\kappa(t) = \frac{1}{2t^{3/2}\sqrt{\pi\kappa}} \exp\left[\frac{1}{4\kappa t}\right].$$

The function k_κ is a Green's function for the heat conduction problem, and the parameter κ is the thermal diffusivity of the object.

The data[⋆] are measurements of the temperature g in a cooling experiment for a particle board. The coefficient κ is equal to 9.55. The integral equation was discretized using the rectangle rule, which yielded a Toeplitz matrix. Thus, matrix–vector multiplications with the matrix A can be computed in $O(n \log n)$ operations using the FFT.

Our computations were done in Matlab 6.5. The problem dimensions were $m = n = 256$, and we first computed the Tikhonov solution x_λ (1.3) using $\lambda = 0.1$; this solution is shown as the solid line in Figure 6.1. The value of λ was chosen as small as possible such that the solution remained visually monotonically decreasing in the interval $[0, 0.9]$ (recall that this is a cooling experiment). The behavior of the solution near $t = 1$ is nonphysical: the solution should decrease monotonically in the whole interval. However, using data from the interval $[0, 1]$ it is impossible to compute any reasonable approximation of the solution for t close to 1, see [3]; therefore, this behavior is expected.

The problem of computing confidence intervals for the Tikhonov solution is described in [4, 12]. The key idea is to choose the vector $w(t_0)$ as samples of an approximate delta function located at some point t_0, and then – ideally – solve the problem (1.1) with $w = \pm w(t_0)$ for $d = 0$, δ equal to the norm

[⋆] The data can be obtained from the authors.

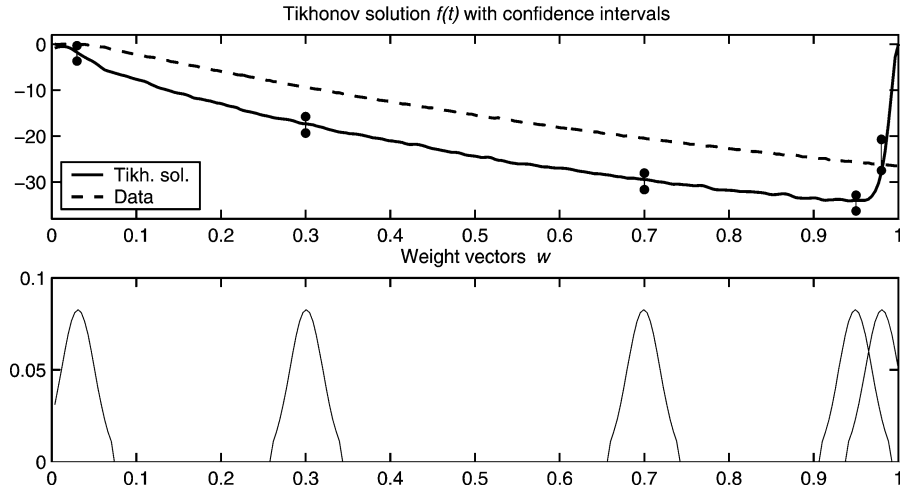

Figure 6.1: The Tikhonov solution x_λ (solid line), the measured data (dashed line) and five confidence intervals computed with $d = 0$, $\delta = 389$, and $\varepsilon = 3.2$. The weight vectors w are plotted in the lower graph.

of the exact solution, and ε equal to the approximate error level in the data. With assumptions on the statistical distribution of the measurement errors it is possible to quantify the level of confidence of the intervals, see [12, Theorem 1]. In the example we chose $w(t_0)$ as samples of a Gaussian function with standard deviation equal to 0.5. Strictly speaking, we only compute confidence intervals for the functionals defined by these vectors. However, if the solution x of the ill-posed problem varies smoothly, then the upper and lower values of the functionals are realistic estimates of upper and lower values of x itself.

Since the exact solution was not available in our experiment, we used $\delta = \|x_\lambda\|_2$ (in this case approximately 389). The error level of the data was estimated as follows: A visual examination of the data showed that the errors are below 0.2 (degrees Celsius). As we wanted to allow the residual to be large enough to accommodate errors of this magnitude, we chose $\varepsilon = \sqrt{256 \cdot 0.2^2} = 3.2$. We then used the MLFIP algorithm to compute confidence intervals at $t_0 = 0.03$, 0.3, 0.7, 0.95 and 0.98 as shown if Figure 6.1. We see that for the first four values of t_0, the confidence intervals are quite small, while near $t = 1$ the confidence interval becomes larger reflecting the nonphysical nature of the solution there, and indicating that one should not trust the solution for values of t close to 1.

The tolerance in the secant method (cf. Figure 4.1) was $\tau = 0.01$, and each call to MLFIP required an average of 12 secant iterations. Each secant iteration involves one evaluation of the function $L(\phi)$ via a call to LSTRS. We used the default storage parameters, for which LSTRS's storage requirement is fixed to 9 vectors of length n. Also, using the default convergence parameters, except `epsilon.Delta` $= 10^{-3}$ and `epsilon.Int` $= 0$ (in order to enforce a boundary solution, cf. [16]) LSTRS needs, on average, 230 matrix–vector multiplications to solve the trust-region problem of computing $L(\phi)$. Many of these multiplications

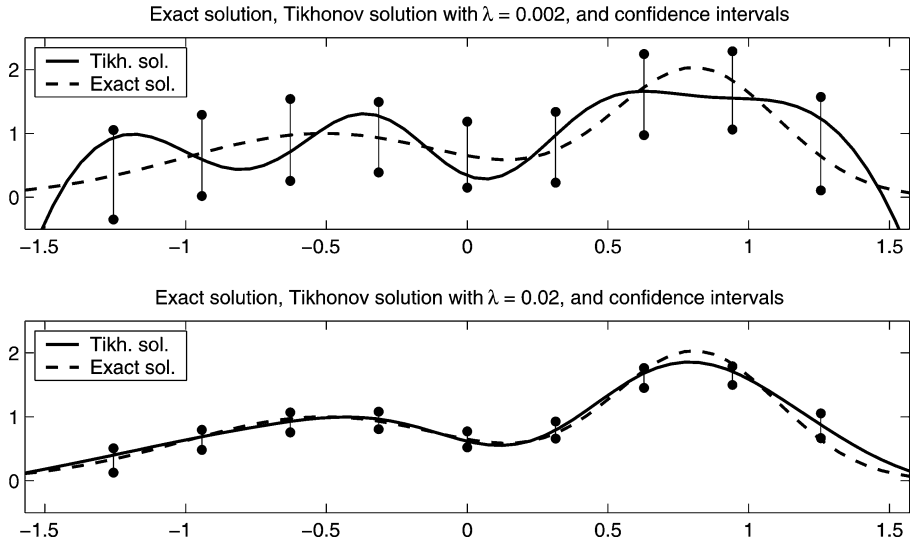

Figure 6.2: Tikhonov solution sx_λ (solid line) and confidence intervals computed with $d = 0$, $\delta = \|x_\lambda\|_2$, and $\varepsilon = \|Ax_\lambda - b\|_2$, for two choices of λ.

are spent in the Tchebyshev spectral transformation that is required for dealing with ill-posed problems.

We remark that we can replace the constraint $\|x\|_2 \leq \delta$ with a constraint $\|Lx\|_2 \leq \delta_L$ involving a semi-norm in which L represents the first derivative operator, as done in [5]. This can, e.g., be implemented via a standard-form transformation [10]. The use of the constraint $\|x\|_2 \leq \delta$ here allows us to better illustrate the usefulness of the confidence intervals.

In our second example, we use MLFIP to compute confidence intervals for Tikhonov solutions to the shaw test problem from [9]. We compute two Tikhonov solutions x_λ; one with a good choice of λ and one whose value is too small. For each x_λ we then compute confidence intervals, using weight vectors generated as above, and with parameters $\varepsilon = \|Ax_\lambda - b\|_2$ and $\delta = \|x_\lambda\|_2$. The results are shown in Figure 6.2, and we see that an undersmoothed solution goes hand in hand with large confidence intervals.

7 Conclusion.

We described MLFIP, a large-scale robust algorithm for the minimization of linear functionals defined on solutions of discrete ill-posed problems. MLFIP is based on an algorithm from [4] and is robust in the sense that it checks for the existence of a solution. The method requires the solution of a sequence of large-scale trust-region subproblems, which are solved by means of the limited-memory method LSTRS [15]. We illustrated the effectiveness of MLFIP on the problem of computing confidence intervals for the solution of an inverse heat conduction problem with real data. Our limited numerical experience indicates

that MLFIP is a valuable new tool for the numerical treatment of large-scale inverse problems.

REFERENCES

1. D. Calvetti and L. Reichel, *Tikhonov regularization with a solution constraint*, SIAM J. Sci. Comput., 26 (2004), pp. 224–239.
2. A. S. Carasso, *Determining surface temperatures from interior observations*, SIAM J. Appl. Math., 42 (1982), pp. 558–574.
3. L. Eldén, *The numerical solution of a non-characteristic Cauchy problem for a parabolic equation*, in P. Deuflhard and E. Hairer (eds), Numerical Treatment of Inverse Problems in Differential and Integral Equations, Proceedings of an International Workshop, Heidelberg, 1982, Birkhäuser, Boston, 1983, pp. 246–268.
4. L. Eldén, *Algorithms for the computation of functionals defined on the solution of discrete ill-posed problems*, BIT, 30 (1990), pp. 466–483.
5. L. Eldén, F. Berntsson, and T. Reginska, *Wavelet and Fourier methods for solving the sideways heat equation*, SIAM J. Sci. Comput., 21 (2001), pp. 2187–2205.
6. G. H. Golub and U. von Matt, *Quadratically constrained least squares and quadratic problems*, Numer. Math., 59 (1991), pp. 561–580.
7. N. Gould, S. Lucidi, M. Roma, and Ph. L. Toint, *Solving the trust–region subproblem using the Lanczos method*, SIAM J. Optim., 9 (1999), pp. 504–525.
8. W. W. Hager, *Minimizing a quadratic over a sphere*, SIAM J. Optim., 12 (2001), pp. 188–208.
9. P. C. Hansen, *Regularization Tools: A Matlab package for analysis and solution of discrete ill-posed problems*, Numer. Algo., 6 (1994), pp. 1–35.
10. P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*, SIAM, Philadelphia, 1998.
11. R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK User's Guide: Solution of Large Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods*, SIAM, Philadelphia, 1998.
12. D. P. O'Leary and B. W. Rust, *Confidence intervals for inequality-constrained least squares problems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 473–489.
13. P. Paateri, *Extreme value estimation, a method for regularizing ill-posed inversion problems*, in A. N. Tikhonov (ed.), *Ill-Posed Problems in Natural Sciences*, VSP, Utrecht, 1992, pp. 118–133.
14. F. Rendl and H. Wolkowicz, *A semidefinite framework for trust region subproblems with applications to large scale minimization*, Math. Prog., 77 (1997), pp. 273–299.
15. M. Rojas, S. A. Santos, and D. C. Sorensen, *A new matrix-free algorithm for the large-scale trust-region subproblem*, SIAM J. Optim., 11 (2000), pp. 611–646.
16. M. Rojas, S. A. Santos, and D. C. Sorensen, *LSTRS: Matlab software for large-scale trust-region subproblems and regularization*, Technical Report 2003-4, Department of Mathematics, Wake Forest University, October 2003, revised October 2004. `http://web.math.wfu.edu/~mrojas/software.html`
17. M. Rojas and D. C. Sorensen, *A trust-region approach to the regularization of large-scale discrete forms of ill-posed problems*, SIAM J. Sci. Comp., 26 (2002), pp. 1843–1861.
18. D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.
19. T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.