DELFT UNIVERSITY OF TECHNOLOGY

REPORT 10-10

EFFICIENT NUMERICAL METHODS FOR LEAST-NORM REGULARIZATION

DANNY C. SORENSEN AND MARIELBA ROJAS

ISSN 1389-6520

Reports of the Department of Applied Mathematical Analysis

Delft 2010

Efficient Numerical Methods for Least-Norm Regularization

D.C. SORENSEN* M. ROJAS†

March 18, 2010

Abstract

The problem

$$\min_{\mathbf{x}} \|\mathbf{x}\|, \;\; s.t. \;\; \|\mathbf{b} - \mathbf{A}\mathbf{x}\| \leq \epsilon$$

arises in the regularization of discrete forms of ill-posed problems when an estimate of the noise level in the data is available. After deriving necessary and sufficient optimality conditions for this problem, we propose two different classes of algorithms: a factorization-based algorithm for small to medium problems, and matrix-free iterations for the large-scale case. Numerical results illustrating the performance of the methods demonstrate that both classes of algorithms are efficient, robust, and accurate. An interesting feature of our formulation is that there is no situation corresponding to the so-called hard case that occurs in the standard trust-region subproblem. Neither small singular values nor vanishing coefficients present any difficulty to solving the relevant secular equations.

1 Introduction

Consider the problem

$$\min_{\mathbf{x}} \|\mathbf{x}\|, \;\; s.t. \;\; \|\mathbf{b} - \mathbf{A}\mathbf{x}\| \leq \epsilon, \tag{1.1}$$

where $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, and $\mathbf{x} \in \mathbb{R}^n$. Note that the matrix $\mathbf{A}$ is any rectangular matrix, ie. both $m \leq n$ *and* $m \geq n$ are allowed. The Euclidean norm is used throughout the paper. Assume that the matrix $\mathbf{A}$ comes from the discretization of an operator in an ill-posed problem, that the vector $\mathbf{b}$ contains data contaminated by noise, and that $\epsilon \geq 0$ is an estimate of the noise level in the data. Then, solving Problem (1.1) yields an approximate (regularized) solution for the noise-free problem [6]. In this regularization approach, the parameter ϵ is used to control the effect of the noise. Note that the objective function in (1.1) could also take the form $\|\mathbf{C}\mathbf{x}\|$. When $\mathbf{C}$ is nonsingular or full-rank, it is possible to convert the problem to the form (1.1). Here, we shall assume $\mathbf{C} = \mathbf{I}$.

We shall assume that all of the error is measurement error residing in the right-hand side $\mathbf{b}$. Thus, the underlying problem is

$$\min_{\mathbf{x}} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|$$

where we assume that $\mathbf{b}$ is a perturbation of exact data, i.e. that

$$\mathbf{b} = \mathbf{b}_o + \mathbf{n} \;\text{ with }\; \mathbf{A}\mathbf{x}_o = \mathbf{b}_o, \tag{1.2}$$

$$\epsilon \geq \|\mathbf{n}\|. \tag{1.3}$$

*Department of Computational and Applied Mathematics, MS 134, Rice University, Houston, Texas 77251-1892 (sorensen@rice.edu). The work of this author was supported in part by NSF grants CCR-9988393 and ACI-0082645, and by AFOSR grant FA9550-09-1-0225.

†Delft Institute of Applied Mathematics, Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands (marielba.rojas@tudelft.nl).

Thus, $\mathbf{b}$ is a perturbation of a vector $\mathbf{b}_o \in Range(\mathbf{A})$ and ϵ is an overestimate of the perturbation error. Among other things, these assumptions assure that the true solution $\mathbf{x}_o$ is a feasible point for Problem (1.1). Additional consequences of these assumptions will emerge during the derivation of algorithms.

Problem (1.1) is a special case of quadratically-constrained quadratic problems studied in [2, 4]. Morozov developed a Newton iteration for small to medium dense problems in [13] and an SVD-based implementation of that method is available from [7, routine **discrep**]. One of the methods presented here for small dense problems is essentially the same as this one with the exception of some implementation and initialization details.

Solution methods that are suitable for large-scale instances of (1.1) have been proposed in [10, 19]. Both methods use Krylov subspace techniques to solve linear systems derived from the optimality conditions.

In this work, we present two classes of methods: a factorization-based Newton iteration for solving the relevant secular equation for small to medium problems, and two matrix-free nonlinear-Lanczos-based methods for solving relevant perturbed nonlinear eigenvalue problems in the large-scale case. Both classes of algorithms are efficient, robust, and accurate. Moreover, neither the presence of small singular values in the coefficient matrix nor vanishing coefficients in the secular equations pose any computational challenge for the methods.

The organization of the paper is as follows. In Section 2, we derive necessary and sufficient optimality conditions for a solution of Problem (1.1). In Section 3, we derive a method for small to medium problems. Section 4 contains suitable reformulations of the optimality conditions for the large-scale case. In Sections 5 and 6, we present matrix-free residual-space and solution-space iterations, respectively, for solving large-scale instances of Problem (1.1). Section 7 contains an analysis of the matrix-free methods. In Section 8, we present numerical results to illustrate the performance of the methods on test problems including large-scale image deblurring. Concluding remarks are presented in Section 9.

2 Necessary and Sufficient Conditions for Minimization

In this section, Lagrange multiplier theory is used to derive necessary (KKT) conditions for Problem (1.1). These results are a special case of the more general derivation of Gander [4]. We use essentially the same device here to derive necessary conditions for this special case. Due to the convexity of the problem, these conditions are also sufficient for optimality (cf. [1, 3]) and this may be used to great advantage in the large-scale setting to be discussed later. Moreover, since the objective function is strictly convex, Problem (1.1) has a unique solution.

Consider the Lagrangian for Problem (1.1):

$$\mathcal{L} := \|\mathbf{x}\|^2 + \lambda(\|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2 - \epsilon^2).$$

The KKT conditions for Problem (1.1) which derive from this Lagrangian are:

$$\mathbf{x} + \lambda \mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{b}) = \mathbf{0}, \;\; \lambda(\|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2 - \epsilon^2) = 0, \;\; \lambda \geq 0.$$

Now, observe that:

- $\|\mathbf{b}\| \leq \epsilon \Leftrightarrow \mathbf{x} = \mathbf{0}$ is a solution,
- $\lambda = 0 \Rightarrow \mathbf{x} = \mathbf{0}$,
- $\lambda > 0 \Leftrightarrow \mathbf{x} \neq \mathbf{0}$ and $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2 = \epsilon^2$.

Thus, the KKT conditions for Problem (1.1) may be restated with a positive λ as follows :

$$\mathbf{x} + \lambda \mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{b}) = \mathbf{0}, \;\; \|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2 = \epsilon^2, \;\; \lambda > 0. \tag{2.1}$$

We now show that the KKT necessary conditions (2.1) are also sufficient.

Lemma 2.1 *If the pair $(\mathbf{x}, \lambda)$ satisfies the KKT conditions (2.1) then $\mathbf{x}$ is the unique solution to Problem (1.1).*

Proof: Put $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$ in the KKT conditions (2.1) to get

$$\mathbf{x} = \lambda \mathbf{A}^T \mathbf{r} \text{ with } \|\mathbf{r}\| = \epsilon, \quad \lambda > 0.$$

Suppose $\hat{\mathbf{x}}$ is any vector satisfying $\|\hat{\mathbf{x}}\| \leq \|\mathbf{x}\|$ with $\|\mathbf{b} - \mathbf{A}\hat{\mathbf{x}}\| \leq \epsilon = \|\mathbf{r}\|$. Put $\hat{\mathbf{x}} = \mathbf{x} + \mathbf{p}$ and note that

$$\|\mathbf{x}\|^2 \geq \|\hat{\mathbf{x}}\|^2 = \|\mathbf{x}\|^2 + 2\mathbf{p}^T\mathbf{x} + \|\mathbf{p}\|^2.$$

Thus, $2\mathbf{p}^T\mathbf{x} \leq -\|\mathbf{p}\|^2$ which gives

$$2\lambda(\mathbf{A}\mathbf{p})^T\mathbf{r} \leq -\|\mathbf{p}\|^2.$$

Note also that $\mathbf{b} - \mathbf{A}\hat{\mathbf{x}} = \mathbf{r} - \mathbf{A}\mathbf{p}$ satisfies $\|\mathbf{r} - \mathbf{A}\mathbf{p}\|^2 = \|\mathbf{b} - \mathbf{A}\hat{\mathbf{x}}\|^2 \leq \epsilon^2 = \|\mathbf{r}\|^2$ implying

$$\|\mathbf{r}\|^2 - 2(\mathbf{A}\mathbf{p})^T\mathbf{r} + \|\mathbf{A}\mathbf{p}\|^2 \leq \|\mathbf{r}\|^2.$$

Hence,

$$0 \leq \lambda\|\mathbf{A}\mathbf{p}\|^2 \leq 2\lambda(\mathbf{A}\mathbf{p})^T\mathbf{r} \leq -\|\mathbf{p}\|^2 \leq 0,$$

which in turn implies that $\mathbf{p} = 0$.

This argument shows that $\|\mathbf{x}\| \leq \|\hat{\mathbf{x}}\|$ for any vector $\hat{\mathbf{x}}$ such that $\|\mathbf{b} - \mathbf{A}\hat{\mathbf{x}}\| \leq \epsilon$. Hence, this $\mathbf{x}$ solves Problem (1.1). Moreover, this solution must be unique. □

In order to derive practical algorithms for solving Problem (1.1) based on conditions (2.1), suitable manipulations are needed. In the rest of this section, we present two versions of the optimality conditions based, respectively, on the Singular Value Decomposition (SVD) and on the QR decomposition of the matrix $\mathbf{A}$.

2.1 Optimality Conditions: SVD version

Let $p = \min\{m, n\}$ and let $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ with $\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{I}_p$ and $\mathbf{S} = diag(\sigma_1, \sigma_2, \ldots, \sigma_p)$ be the (short-form) SVD of $\mathbf{A}$. Let $\mathbf{b} = \mathbf{U}\mathbf{b}_1 + \mathbf{b}_2$ with $\mathbf{U}^T\mathbf{b}_2 = \mathbf{0}$. Then

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2 = \|\mathbf{U}(\mathbf{b}_1 - \mathbf{S}\mathbf{V}^T\mathbf{x})\|^2 + \|\mathbf{b}_2\|^2$$

and thus

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2 \leq \epsilon^2 \iff \|\mathbf{b}_1 - \mathbf{S}\mathbf{V}^T\mathbf{x}\|^2 \leq \epsilon^2 - \|\mathbf{b}_2\|^2 =: \delta^2, \tag{2.2}$$

where we have assumed that $\epsilon^2 \geq \|\mathbf{b}_2\|^2$, since $\|\mathbf{b}_2\| > \epsilon$ would imply that there is no feasible point. Therefore, we must assume $\mathbf{b} = \mathbf{U}\mathbf{b}_1 + \mathbf{b}_2$ with $\|\mathbf{b}_2\| \leq \epsilon$.

Using (2.2), the KKT conditions (2.1) become:

$$\mathbf{x} + \lambda\mathbf{V}\mathbf{S}(\mathbf{S}\mathbf{V}^T\mathbf{x} - \mathbf{b}_1) = \mathbf{0}, \ \|\mathbf{b}_1 - \mathbf{S}\mathbf{V}^T\mathbf{x}\|^2 = \delta^2, \ \lambda > 0. \tag{2.3}$$

We now manipulate these equations into a more useful form. Note that multiplying the first equation in (2.3) on the left by $\mathbf{S}\mathbf{V}^T$ and subtracting $\mathbf{b}_1$ from both sides gives:

$$\mathbf{S}\mathbf{V}^T\mathbf{x} - \mathbf{b}_1 + \lambda\mathbf{S}^2(\mathbf{S}\mathbf{V}^T\mathbf{x} - \mathbf{b}_1) = -\mathbf{b}_1.$$

Hence, the KKT conditions become:

$$(\mathbf{I} + \lambda\mathbf{S}^2)(\mathbf{b}_1 - \mathbf{S}\mathbf{V}^T\mathbf{x}) = \mathbf{b}_1, \|\mathbf{b}_1 - \mathbf{S}\mathbf{V}^T\mathbf{x}\|^2 \leq \delta^2, \ \lambda > 0. \tag{2.4}$$

As a final transformation, we define $\mathbf{z} := \mathbf{b}_1 - \mathbf{S}\mathbf{V}^T\mathbf{x}$ to put this problem in the form:

$$(\mathbf{I} + \lambda\mathbf{S}^2)\mathbf{z} = \mathbf{b}_1, \ \|\mathbf{z}\|^2 \leq \delta^2, \ \lambda > 0. \tag{2.5}$$

Now, recalling that $\mathbf{x} + \lambda\mathbf{V}\mathbf{S}(\mathbf{S}\mathbf{V}^T\mathbf{x} - \mathbf{b}_1) = \mathbf{0}$, we have:

$$\mathbf{x} = \lambda\mathbf{V}\mathbf{S}\mathbf{z}.$$

2.2 Optimality Conditions: QR version

Following a similar approach to the one in Section 2.1, let $\mathbf{A} = \mathbf{QR}$ with $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$ be the (short-form) QR factorization, and let $\mathbf{b} = \mathbf{Q}\mathbf{b}_1 + \mathbf{b}_2$, with $\mathbf{Q}^T\mathbf{b}_2 = 0$. Hence, conditions (2.1) become:

$$(\mathbf{I} + \lambda\mathbf{R}\mathbf{R}^T)\mathbf{z} = \mathbf{b}_1,\ \|\mathbf{z}\|^2 = \delta^2,\ \lambda > 0, \tag{2.6}$$

where $\mathbf{z} = \mathbf{b}_1 - \mathbf{R}\mathbf{x}$, $\delta^2 = \epsilon^2 - \|\mathbf{b}_2\|^2$, and the solution vector satisfies $\mathbf{x} = \lambda\mathbf{R}^T\mathbf{z}$.

Note that to ensure feasibility, we must again assume $\|\mathbf{b}_2\| \le \epsilon$.

3 A Newton Iteration for Dense Problems

In this section, we derive an algorithm suitable for dense problems. That is, we assume the problem size is such that a direct computation of the short-form SVD of $\mathbf{A}$ is affordable. The derivation of the previous section indicates that we must perform the following steps:

- Compute $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$;
- Put $\mathbf{b}_1 = \mathbf{U}^T\mathbf{b}$;
- Set $\delta^2 = \epsilon^2 - \|\mathbf{b} - \mathbf{U}\mathbf{b}_1\|^2$;
- Compute $\lambda \ge 0$ and $\mathbf{z}$ s.t. $(\mathbf{I} + \lambda\mathbf{S}^2)\mathbf{z} = \mathbf{b}_1,\ \ \|\mathbf{z}\|^2 \le \delta^2$;
- Put $\mathbf{x} = \lambda\mathbf{V}\mathbf{S}\mathbf{z}$.

The choice of algorithm for computing λ is guided by previous algorithms developed for various secular equations, for example [5, 8, 11, 12, 17]. The simplest choice is to apply a safeguarded Newton's method to find a positive zero of the function

$$\psi(\lambda) := \frac{1}{\|\mathbf{z}_\lambda\|} - \frac{1}{\delta},$$

where

$$\mathbf{z}_\lambda := (\mathbf{I} + \lambda\mathbf{S}^2)^{-1}\mathbf{b}_1.$$

In the following discussion, we shall see that ψ is monotone increasing and concave on $[0, \infty)$. Hence, whenever $\|\mathbf{b}_2\| < \epsilon$, there is a unique $\lambda > 0$ such that $\|\mathbf{z}_\lambda\| = \delta$. To this end, suppose that $rank(\mathbf{A}) = r$ so that the singular values of $\mathbf{A}$ are $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_r > \sigma_{r+1} = \cdots = \sigma_n = 0$. Denote the jth component of $\mathbf{b}_1$ by β_j for $1 \le j \le n$, and consider the rational function

$$\phi(\lambda) := \mathbf{z}_\lambda^T\mathbf{z}_\lambda = \sum_{j=1}^{r} \frac{\beta_j^2}{(1 + \lambda\sigma_j^2)^2} + \beta_o^2,$$

where $\beta_o^2 := \sum_{j=r+1}^{n} \beta_j^2$. With this formulation, the key rational function ϕ only has poles at the points $-\frac{1}{\sigma_j^2}$ for $1 \le j \le r$. Therefore, the function $\phi(\lambda)$ is always well defined on the interval $[0, \infty)$ and there is no difficulty associated with small singular values or vanishing coefficients β_j. Note also that the assumption $\|\mathbf{b}_2\|^2 \le \epsilon^2$ implies $\|\mathbf{b}_1\| > \delta$. It follows that $\phi(0) = \|\mathbf{b}_1\|^2 > \delta^2$. Moreover, $\lim_{\lambda\to\infty} \phi(\lambda) = \beta_o^2 < \delta^2$. We now observe that

$$\phi'(\lambda) = -2\sum_{j=1}^{r} \frac{\sigma_j^2\beta_j^2}{(1 + \lambda\sigma_j^2)^3} = -2{\mathbf{b}_1}^T\mathbf{S}(\mathbf{I} + \lambda\mathbf{S}^2)^{-3}\mathbf{S}\mathbf{b}_1 < 0,$$

for all $\lambda \in (0, \infty)$. Thus ϕ is strictly monotone decreasing on $(0, \infty)$ and there must be a unique solution $\lambda = \lambda_o \in (0, \infty)$ to the equation $\phi(\lambda) = \delta^2$. Finally, it is straightforward to see that $\phi''(\lambda) > 0$ for all $\lambda \in (0, \infty)$.

Since $\phi(\lambda)$ is positive, strictly monotone decreasing, and strictly convex on $(0, \infty)$, it follows that $\psi'(\lambda) > 0$ and $\psi''(\lambda) < 0$ for all $\lambda \in (0, \infty)$. Therefore, we have:

1. $\psi(\lambda)$ is concave and monotone increasing for $\lambda \in (0, \infty)$,
2. $\psi(\lambda) = 0$ has a unique root at $\lambda = \lambda_o > 0$.

Given these properties, it is well known that Newton's method applied to finding a root of ψ will converge monotonically and quadratically to λ_o from any starting point in the interval $[0, \lambda_o]$ without safeguarding. Nevertheless, some safeguarding mechanism should be included. An initial guess in the interval $[0, \lambda_o)$ is available due to the following analysis. It is easily seen that:

$$\frac{\|\mathbf{b}_1\|^2}{(1+\lambda_o\sigma_1^2)^2} \le \phi(\lambda_o) = \delta^2.$$

Hence, an initial guess $\lambda_1 \in [0, \lambda_o)$ can be constructed as:

$$\lambda_1 := \frac{\|\mathbf{b}_1\| - \delta}{\delta\sigma_1^2} < \lambda_o. \tag{3.1}$$

The Newton iteration for finding a root of ψ (solving the secular equation $\psi(\lambda) = 0$) is presented in Figure 1 as function LNR_SecEqnSol. The complete algorithm for Least-Norm Regularization is presented in Figure 2 as function LNR.

```
function [$\mathbf{z}$,$\lambda$] = LNR_SecEqnSol($\mathbf{S}$, $\mathbf{b}_1$, $\delta$, $\tau$)

        Put $\mathbf{D} = \mathbf{S}^2$;  $\psi = 1$;
        $\lambda = \frac{\|\mathbf{b}_1\|-\delta}{\delta\sigma_1^2}$;
        while ($|\psi| > \tau$),
            $\mathbf{z} = (\mathbf{I} + \lambda\mathbf{D})^{-1}\mathbf{b}_1$;
            $\mathbf{w} = \mathbf{S}\mathbf{z}$ :
            $\phi = \|\mathbf{z}\|$;
            $d\phi = \frac{z' * z}{\mathbf{w}^T(\mathbf{I}+\mathbf{D})^{-1}\mathbf{w}}$
            $\psi = 1 - \phi/\delta$ ;
            $\lambda \leftarrow \lambda - \psi * d\phi$;
        end
```

Figure 1: Algorithm LNR_SecEqnSol: Newton's method on the secular equation $\psi(\lambda) = 0$.

As mentioned during the introduction, Morozov proposed the same basic Newton iteration in [13] and an SVD-based implementation of that method is available from [7, routine **discrep**]. However, we have introduced some important performance and robustness additions. We have introduced an initial guess, Equation (3.1), which can significantly reduce the number of Newton steps required to solve the secular equation. We also provide the observation that ϵ must be replaced by δ, where $\delta^2 = \epsilon^2 - \|\mathbf{b} - \mathbf{U}\mathbf{b}_1\|^2$ and discuss the implications on feasibility when this difference is negative. Our implementation incorporates ideas pioneered in Hebden [8] although our test examples did not reveal any significant performance differences between the implementation from [7, routine **discrep**] and our method.

```
function [$\mathbf{x}$,$\lambda$] = LNR($\mathbf{A}$, $\mathbf{b}$, $\epsilon$, $\tau$)

        if $\|b\| \leq \epsilon$
            $\mathbf{x} = 0$; $\lambda = 0$;
        else
            Compute $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$ (SVD);
            Put $\mathbf{b}_1 = \mathbf{U}^T\mathbf{b}$;
            Put $\mathbf{b}_2 = \mathbf{b} - \mathbf{U}\mathbf{b}_1$;
            if $\|\mathbf{b}_2\| > \epsilon$
                Problem is infeasible
            else
                Set $\delta = \sqrt{\epsilon^2 - \|\mathbf{b}_2\|^2}$;
                $[\mathbf{z}, \lambda] = \mathrm{LNR\_SecEqnSol}(\mathbf{S}, \mathbf{b}_1, \delta, \tau)$;
                Put $\mathbf{x} = \lambda\mathbf{V}\mathbf{S}\mathbf{z}$;
            end
        end
```

Figure 2: Algorithm LNR for Least-Norm Regularization.

4 The Large-Scale Setting

In the large-scale case, some suitable modifications to the conditions derived in Section 3 are required. We return to the KKT conditions (2.1). Multiplying the first equation of (2.1) through by $\mathbf{A}$ and adding $-\mathbf{b}$ to both sides gives:

$$\mathbf{Ax} - \mathbf{b} + \lambda \mathbf{A}\mathbf{A}^T(\mathbf{Ax} - \mathbf{b}) = -\mathbf{b}.$$

Put $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ to obtain the equation:

$$(\mathbf{I} + \lambda \mathbf{A}\mathbf{A}^T)\mathbf{r} = \mathbf{b} \tag{4.1}$$

and set $\mathbf{x}_\lambda = \lambda \mathbf{A}^T\mathbf{r}$. Note that the equation for $\mathbf{r}$ should be reasonably well conditioned for positive λ.

The zero-finding algorithm must be cast in terms of adjusting λ so that:

$$\|\mathbf{r}_\lambda\| = \epsilon. \tag{4.2}$$

A variety of possibilities exist for iteratively solving Equation (4.2) and a very promising one is derived in the following section. However, this form may be inconvenient for an $m \times n$ matrix $\mathbf{A}$ with $m >> n$ because it generally involves vectors of length m rather than n. Working with the KKT conditions (2.1) directly will address the case $m >> n$.

The first equation in (2.1) can be written as:

$$(\mathbf{I} + \lambda \mathbf{A}^T\mathbf{A})\mathbf{x} = \lambda \mathbf{A}^T\mathbf{b}. \tag{4.3}$$

Given an $n \times k$ orthogonal matrix $\mathbf{V}$, let $\mathbf{QR} = \mathbf{AV}$ be the short-form QR-factorization with $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}_k$ and $\mathbf{R}$ upper triangular. To obtain a projected problem, put $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$ and obtain the projected equation:

$$(\mathbf{I} + \lambda(\mathbf{AV})^T(\mathbf{AV}))\hat{\mathbf{x}} = \lambda(\mathbf{AV})^T\mathbf{b}$$

by multiplying through with $\mathbf{V}^T$ from the left. Substituting $\mathbf{QR}$ in place of $\mathbf{AV}$ gives:

$$(\mathbf{I} + \lambda \mathbf{R}^T\mathbf{R})\hat{\mathbf{x}} = \lambda \mathbf{R}^T\mathbf{Q}^T\mathbf{b}.$$

Assuming $\mathbf{R}^T$ is nonsingular gives:

$$(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)\mathbf{R}^{-T}\hat{\mathbf{x}} = \lambda\hat{\mathbf{b}} \text{ with } \hat{\mathbf{b}} \equiv \mathbf{Q}^T\mathbf{b}.$$

Thus $\hat{\mathbf{x}} = \lambda \mathbf{R}^T(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)^{-1}\hat{\mathbf{b}}$ and

$$\mathbf{b} - \mathbf{Ax} = \mathbf{b} - (\mathbf{AV})\hat{\mathbf{x}} = \mathbf{b} - \lambda \mathbf{Q}(\mathbf{R}\mathbf{R}^T)(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)^{-1}\hat{\mathbf{b}}.$$

Put $\mathbf{f} = \mathbf{b} - \mathbf{Q}\hat{\mathbf{b}}$, substitute and manipulate to obtain:

$$\mathbf{b} - \mathbf{Ax} = \mathbf{f} + \mathbf{Q}[\mathbf{I} - \lambda(\mathbf{R}\mathbf{R}^T)(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)^{-1}]\hat{\mathbf{b}} = \mathbf{f} + \mathbf{Q}(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)^{-1}\hat{\mathbf{b}}.$$

Now, observe that

$$\|\mathbf{b} - \mathbf{Ax}\|^2 = \|\mathbf{f}\|^2 + \|(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)^{-1}\hat{\mathbf{b}}\|^2$$

since $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{Q}^T\mathbf{f} = \mathbf{0}$. Thus, $\|\mathbf{b} - \mathbf{Ax}\| = \epsilon$ is satisfied when

$$\|(\mathbf{I} + \lambda \mathbf{R}\mathbf{R}^T)^{-1}\hat{\mathbf{b}}\|^2 = \epsilon^2 - \|\mathbf{f}\|^2. \tag{4.4}$$

If m is not huge, it may be worthwhile to actually use the QR-factorization of $\mathbf{AV}$. Column updating can be utilized to update this factorization as $\mathbf{V}$ expands column by column. When m is huge, one might get $\mathbf{R}$ from a

Cholesky factorization of $(\mathbf{AV})^T(\mathbf{AV})$ but this will require the storage (or re-computation at each step) of $\mathbf{AV}$ as $\mathbf{V}$ expands.

Obtaining Equation (4.4) for the residual will enable the derivation of an algorithm using Algorithm LNR_SecEqnSol shown in Figure 1 to solve the secular equation as before. This can be done either using an SVD of $\mathbf{R}$ or deriving a version of LNR that will use $\mathbf{R}$ instead. Details will be given when we derive the corresponding algorithm.

Matrix-free algorithms must be constructed when $\mathbf{A}$ is either not explicitly available or too large to make the computation of an SVD affordable. Large-scale instances of the related, standard trust-region subproblem

$$\min \|\mathbf{b} - \mathbf{Ax}\| \ \text{s.t.} \ \|\mathbf{x}\| \leq \Delta,$$

can be solved with the iterative algorithm LSTRS [15, 16, 17]. Recently [9], the performance of the LSTRS algorithm was greatly improved by incorporating the nonlinear Lanczos iteration proposed by Voss [20].

The Voss nonlinear Lanczos approach worked so well in the standard setting that it seemed natural to apply a variant of it for solving the constrained least-norm problem (1.1). However, to obtain a suitable formulation for the least norm regularization problem, the parametric eigenvalue problem at the heart of LSTRS must be replaced by an appropriate nonlinear function, the secular equation.

We shall present two nonlinear Lanczos iterations for the least-norm regularization problem. One of these will be in the *residual* $\mathbf{r}$*-space* and the other will be in the *solution or* $\mathbf{x}$*-space*. At each step of these iterations an existing basis V is expanded. A linear Lanczos method would expand this space with a vector obtained by the 3-term Lanczos recurrence and this would amount to a standard Krylov update. Our iteration is nonlinear because it obtains the new expansion direction $\mathbf{v}$ from a nonlinear process, the solution of the secular equation . In Section 7, we show that the adjoined vector $\mathbf{v}$ is in the direction of a projected gradient (or a Newton-Like step if a pre-conditioner is used) using a direction obtained from the solution of the secular equation. In many ways, this is analogous to the expansion step in a Jacobi-Davidson iteration [18].

5 Nonlinear Lanczos $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ Iteration

As we mentioned above, in the large-scale setting, it may be intractable to work with the SVD as we did in Section 3. It is quite natural however to work with Equation (4.1) to approximately solve Equation (4.2) with an iterative projection approach. Thus, the parametric eigenvalue problem in LSTRS is replaced by the nonlinear equation $\|(\mathbf{I} + \lambda\mathbf{AA}^T)^{-1}\mathbf{b})\| = \epsilon$.

An orthogonal matrix $\mathbf{V} \in \mathbb{R}^{m\times k}$ is used to construct a projection of Equation (4.1) as follows:

- Put $\mathbf{r} = \mathbf{V}\hat{\mathbf{r}}$ and express $\mathbf{b} = \mathbf{V}\hat{\mathbf{b}} + \mathbf{f}$ with $\mathbf{V}^T\mathbf{f} = 0$.
- Then the projected problem based upon Equation (4.1) becomes:
$$\mathbf{V}^T(\mathbf{I} + \lambda\mathbf{AA}^T)\mathbf{V}\hat{\mathbf{r}} = \hat{\mathbf{b}}. \tag{5.1}$$
- If $\mathbf{WSU}^T = \mathbf{V}^T\mathbf{A}$ is the (short-form) SVD, then the projected form of Equation (4.2) becomes:
$$\|(\mathbf{I} + \lambda\mathbf{S}^2)^{-1}\mathbf{b}_1\| = \epsilon \ \text{with} \ \mathbf{b}_1 = \mathbf{W}^T\hat{\mathbf{b}}. \tag{5.2}$$
- The secular equation (5.2) is solved to obtain λ using Algorithm LNR_SecEqnSol in Figure 1 with $\delta = \epsilon$.
- We then put $\mathbf{x}_\lambda = \lambda\mathbf{A}^T\mathbf{V}\hat{\mathbf{r}} = \lambda\mathbf{A}^T\mathbf{V}(\mathbf{Wz}) = \mathbf{USz}\lambda$, where $\mathbf{z} = (\mathbf{I} + \lambda\mathbf{S}^2)^{-1}\mathbf{b}_1$.
- *Nonlinear Lanczos Step*: Now, compute $\mathbf{r} = \mathbf{b} - \mathbf{Ax}_\lambda$ and obtain a new search direction $\mathbf{v} = (\mathbf{I} - \mathbf{VV}^T)\mathbf{r}$ and set $\mathbf{v} \leftarrow \mathbf{v}/\|\mathbf{v}\|$. This vector is adjoined to $\mathbf{V}$ to update the basis $\mathbf{V} \leftarrow [\mathbf{V}, \mathbf{v}]$ and then the process is repeated until convergence.

This iteration must terminate at the nonlinear Lanczos step if $(\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{r} = \mathbf{v} = \mathbf{0}$. However, this would be a fortunate event since it would imply that the problem has been solved. That is,

$$(\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{r} = \mathbf{0} \;\Rightarrow\; (\mathbf{I} + \lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} = \mathbf{b} \;\text{ and }\; \|\mathbf{r}\| = \epsilon.$$

In other words, this $\mathbf{r}$ satisfies the KKT conditions (2.1). To see this, note that in such a case $\mathbf{r} = \mathbf{V}\mathbf{s}$ must hold for some $\mathbf{s}$. Thus,

$$\mathbf{V}\mathbf{s} = \mathbf{b} - \mathbf{A}\mathbf{x}_\lambda = \mathbf{b} - \lambda\mathbf{A}\mathbf{A}^T\mathbf{V}\hat{\mathbf{r}}.$$

Hence,

$$\mathbf{s} = \hat{\mathbf{b}} - \lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V}\hat{\mathbf{r}} = (\mathbf{I} + \lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V})\hat{\mathbf{r}} - \lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V}\hat{\mathbf{r}} = \hat{\mathbf{r}}$$

This gives

$$\mathbf{r} = \mathbf{V}\hat{\mathbf{r}} = \mathbf{b} - \lambda\mathbf{A}\mathbf{A}^T\mathbf{V}\hat{\mathbf{r}},$$

and thus

$$(\mathbf{I} + \lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} = \mathbf{b}.$$

Since equation (5.2) implies $\mathbf{r} = \mathbf{V}\hat{\mathbf{r}}$ and $\|\mathbf{r}\| = \|\hat{\mathbf{r}}\| = \epsilon$ this $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}_\lambda$ satisfies the KKT conditions. An additional consequence is that this nonlinear Lanczos iteration is finitely terminating since $\mathbf{V}\mathbf{V}^T = \mathbf{I}$ whenever $k = p = \min\{m, n\}$.

This discussion leads to the nonlinear Lanczos iteration in Figure 3 for satisfying Equations (4.1) and (4.2).

6 Nonlinear Lanczos $\mathbf{x}$ Iteration

Virtually the same procedure as that of the previous section can be used to derive a nonlinear Lanczos iteration in the solution space. In this case we shall work with Equation (4.3) to approximately solve Equation (4.4) with an iterative projection approach.

Now, an $n \times k$ orthogonal matrix $\mathbf{V}$ shall be used to construct a projection of Equation (4.3) as follows:

- Put $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$ and compute $\mathbf{Q}\mathbf{R} = \mathbf{A}\mathbf{V}$, the short form QR-factorization. Express $\mathbf{b} = \mathbf{Q}\hat{\mathbf{b}} + \mathbf{f}$ with $\mathbf{Q}^T\mathbf{f} = 0$. Set $\delta = \sqrt{\epsilon^2 - \mathbf{f}^T\mathbf{f}}$. If δ is not real, $\mathbf{V}$ must be expanded (via standard Lanczos steps applied to $\mathcal{K}_k(\mathbf{A}^T\mathbf{A}, \mathbf{A}^T\mathbf{b})$) until it becomes real.

- The projection procedure described in Section 4 is followed to arrive at the problem of solving

$$\|(\mathbf{I} + \lambda\mathbf{R}\mathbf{R}^T)^{-1}\hat{\mathbf{b}}\|^2 = \delta^2,$$

  which replaces LSTRS's parametric eigenvalue problem in this approach.

- If $\mathbf{W}\mathbf{S}\mathbf{U}^T = \mathbf{R}$ is the (short form) SVD, then the projected form of Equation (4.4) becomes

$$\|(\mathbf{I} + \lambda\mathbf{S}^2)^{-1}\mathbf{b}_1\| = \delta \quad \text{with} \quad \mathbf{b}_1 = \mathbf{W}^T\hat{\mathbf{b}}. \tag{6.1}$$

- The secular equation (6.1) is solved to obtain λ using Algorithm LNR_SecEqnSol in Figure 1 with δ as specified above.

- We then put $\mathbf{x}_\lambda = \lambda\mathbf{V}(\mathbf{U}\mathbf{S}\mathbf{z})$ where $\mathbf{z} = (\mathbf{I} + \lambda\mathbf{S}^2)^{-1}\mathbf{b}_1$.

- *Nonlinear Lanczos Step*: Now, compute $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}_\lambda$ and obtain a new search direction $\mathbf{v} = (\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r})$. The normalized vector $\mathbf{v} \leftarrow \mathbf{v}/\|\mathbf{v}\|$ is adjoined to $\mathbf{V}$ to update the basis $\mathbf{V} \leftarrow [\mathbf{V}, \mathbf{v}]$ and then the process is repeated until convergence.

```
function [$\mathbf{x}, \mathbf{r}, \lambda$] = LNR_NLLr($\mathbf{A}, \mathbf{b}, \epsilon, \tau$)

Input:   $\mathbf{A}$   an $m$ by $n$ matrix
         $\mathbf{b}$   a vector of length $m$
         $\epsilon$   a positive scalar with $\epsilon \geq \|\mathbf{b} - \mathbf{b}_o\|$
         $\tau$   tolerance on norm constraint relative accuracy

Output:  $\mathbf{x}$   solution vector of length $n$
         $\mathbf{r}$   the linear system residual $\mathbf{b} - \mathbf{Ax}$
         $\lambda$   a parameter such that $(\mathbf{I} + \lambda\mathbf{AA}^T)\mathbf{r} = \mathbf{b}$
             This solves $\min \|\mathbf{x}\|$ s.t. $\|\mathbf{b} - \mathbf{Ax}\| = \epsilon$

         Compute an initial basis $\mathbf{V}$ for $\mathcal{K}_k(\mathbf{AA}^T, \mathbf{b})$ via Lanczos
             (with $\mathbf{Ve}_1 = \mathbf{b}/\|\mathbf{b}\|$)
         Set $\mathbf{r} = \mathbf{b}$;  $\mathbf{x} = \mathbf{0}$;  $\lambda = 0$;

         while (| $\|\mathbf{r}\| - \epsilon$ | $> \epsilon \cdot \tau$),
             Compute $\mathbf{V}^T\mathbf{A} = \mathbf{WSU}^T$ (SVD);
             Set $\mathbf{b}_1 = \mathbf{W}^T(\mathbf{V}^T\mathbf{b})$;
             $[\mathbf{z}, \lambda] =$ LNR_SecEqnSol$(\mathbf{S}, \mathbf{b}_1, \epsilon, \tau)$;
             $\mathbf{x} = \mathbf{U}(\mathbf{Sz}\lambda)$;
             $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$;
             $\mathbf{v} \leftarrow \mathbf{r} - \mathbf{V}(\mathbf{V}^T\mathbf{r})$;  $\mathbf{v} \leftarrow \mathbf{v}/\|\mathbf{v}\|$;
             $\mathbf{V} \leftarrow [\mathbf{V}, \mathbf{v}]$;
         end
```

Figure 3: Algorithm LNR_NLLr: Nonlinear Lanczos Residual-Space Iteration for LNR.

This iteration must terminate at the nonlinear Lanczos step if $(\mathbf{I}-\mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}) = \mathbf{v} = \mathbf{0}$. However, this would again be a fortunate event since it would imply that the problem has been solved. In other words,

$$(\mathbf{I}-\mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}) = \mathbf{0} \;\Rightarrow\; \mathbf{x}+\lambda\mathbf{A}^T(\mathbf{A}\mathbf{x}-\mathbf{b}) = \mathbf{0},\;\; \|\mathbf{b}-\mathbf{A}\mathbf{x}\| = \epsilon,\;\; \lambda > 0.$$

To see this, note $\mathbf{v} = 0$ implies $\lambda\mathbf{A}^T\mathbf{r} = \mathbf{V}\mathbf{s}$, and therefore:

$$\mathbf{V}\mathbf{s} = \lambda\mathbf{A}^T(\mathbf{b}-\mathbf{A}\mathbf{x}) = \lambda\mathbf{A}^T\mathbf{b} - \lambda\mathbf{A}^T\mathbf{A}\mathbf{V}[\mathbf{I}+\lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})]^{-1}\lambda(\mathbf{A}\mathbf{V})^T\mathbf{b}.$$

Thus,

$$\begin{aligned}
\mathbf{s} &= \lambda(\mathbf{A}\mathbf{V})^T\mathbf{b} - \lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})[\mathbf{I}+\lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})]^{-1}\lambda(\mathbf{A}\mathbf{V})^T\mathbf{b} \\
&= [\mathbf{I} - \lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})[\mathbf{I}+\lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})]^{-1}]\lambda(\mathbf{A}\mathbf{V})^T\mathbf{b} \\
&= [\mathbf{I}+\lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})]^{-1}\lambda(\mathbf{A}\mathbf{V})^T\mathbf{b} \\
&= \hat{\mathbf{x}}.
\end{aligned}$$

Hence, $\lambda\mathbf{A}^T\mathbf{r} = \mathbf{V}\mathbf{s} = \mathbf{V}\hat{\mathbf{x}} = \mathbf{x}$ and it follows that

$$\mathbf{0} = \mathbf{x} - \lambda\mathbf{A}^T\mathbf{r} = \mathbf{x} + \lambda\mathbf{A}^T(\mathbf{A}\mathbf{x}-\mathbf{b}),$$

with $\|\mathbf{b}-\mathbf{A}\mathbf{x}\| = \|\mathbf{r}\| = \epsilon$, and $\lambda > 0$.

Thus, the KKT conditions (2.1) are satisfied and $\mathbf{x}$ is the unique solution to Problem (1.1). Again, an additional consequence is that this nonlinear Lanczos iteration is finitely terminating since $\mathbf{V}\mathbf{V}^T = \mathbf{I}$ whenever $k = p = \min\{m, n\}$.

This discussion leads to the nonlinear Lanczos iteration in Figure 4 for satisfying Equations (4.3) and (4.4).

7 Analysis

7.1 The residual space iteration

We have demonstrated that the KKT conditions are equivalent to finding $\mathbf{r}, \lambda > 0$ such that

$$(\mathbf{I}+\lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} = \mathbf{b} \;\text{ with }\; \|\mathbf{r}\| = \epsilon.$$

The solution $\mathbf{x} = \lambda\mathbf{A}^T\mathbf{r}$ is a by-product. For a given λ, the first equation amounts to solving the problem

$$\min_{\mathbf{r}}\{\frac{1}{2}\mathbf{r}^T(\mathbf{I}+\lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} - \mathbf{b}^T\mathbf{r}\} \equiv \min_{\mathbf{r}}\varphi(\mathbf{r},\lambda),$$

since $(\mathbf{I}+\lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} - \mathbf{b} = \nabla_{\mathbf{r}}\varphi(\mathbf{r},\lambda)$. At a point $\mathbf{r} = \mathbf{V}\hat{\mathbf{r}}$, the steepest descent direction is given by

$$\mathbf{s} = -[(\mathbf{I}+\lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} - \mathbf{b}] = (\mathbf{b}-\mathbf{A}\mathbf{x}-\mathbf{r}).$$

This direction may be decomposed into orthogonal components

$$\mathbf{s} = \mathbf{V}\mathbf{V}^T\mathbf{s} + (\mathbf{I}-\mathbf{V}\mathbf{V}^T)\mathbf{s} = \mathbf{V}[\hat{\mathbf{b}} - (\mathbf{I}+\lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V})\hat{\mathbf{r}}] + (\mathbf{I}-\mathbf{V}\mathbf{V}^T)(\mathbf{b}-\mathbf{A}\mathbf{x}-\mathbf{r}) = (\mathbf{I}-\mathbf{V}\mathbf{V}^T)(\mathbf{b}-\mathbf{A}\mathbf{x}),$$

since

$$(\mathbf{I}+\lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V})\hat{\mathbf{r}} = \hat{\mathbf{b}} \;\text{ and }\; (\mathbf{I}-\mathbf{V}\mathbf{V}^T)\mathbf{r} = \mathbf{0}.$$

Hence, the step $\mathbf{s}$ used in Algorithm LNR_NLLr is in fact the steepest descent direction for φ at $\mathbf{r} = \mathbf{V}\hat{\mathbf{r}}$ and λ. The vector $\mathbf{v} = \mathbf{s}/\|\mathbf{s}\|$ is adjoined to the search space $\mathcal{S} = \mathrm{Range}([\mathbf{V}, \mathbf{v}])$ and this space contains all points of the form $\mathbf{r}_\alpha = \mathbf{r} + \mathbf{v}\alpha$ and hence contains the minimum of φ along the steepest descent direction.

```
function $[\mathbf{x}, \mathbf{r}, \lambda]$ = LNR_NLLx($\mathbf{A}, \mathbf{b}, \epsilon, \tau$)

Input:   $\mathbf{A}$   an $m$ by $n$ matrix
         $\mathbf{b}$   a vector of length $m$
         $\epsilon$   a positive scalar with $\epsilon \geq \|\mathbf{b} - \mathbf{b}_o\|$
         $\tau$   tolerance on norm constraint relative accuracy

Output:  $\mathbf{x}$   solution vector of length $n$
         $\mathbf{r}$   the linear system residual $\mathbf{b} - \mathbf{A}\mathbf{x}$
         $\lambda$   a parameter such that $(\mathbf{I} + \lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} = \mathbf{b}$
             This solves $\min \|\mathbf{x}\|$ s.t. $\|\mathbf{b} - \mathbf{A}\mathbf{x}\| = \epsilon$

         Compute an initial basis $\mathbf{V}$ for $\mathcal{K}_k(\mathbf{A}^T\mathbf{A}, \mathbf{A}^T\mathbf{b})$ via Lanczos
         (with $\mathbf{V}\mathbf{e}_1 = \mathbf{A}^T\mathbf{b}/\|\mathbf{A}^T\mathbf{b}\|$), performing enough steps to
         ensure that $\epsilon \geq \|\mathbf{f}\|$.
         Set $\mathbf{r} = \mathbf{b}$;  $\mathbf{x} = \mathbf{0}$;  $\lambda = 0$;

         while ($|\ \|\mathbf{r}\| - \epsilon\ | > \epsilon \cdot \tau$),
             Compute $\mathbf{A}\mathbf{V} = \mathbf{Q}\mathbf{R}$;
             Compute $\mathbf{R} = \mathbf{W}\mathbf{S}\mathbf{U}^T$ (SVD);
             Set $\mathbf{h} = \mathbf{Q}^T\mathbf{b}$;
             Set $\mathbf{b}_1 = \mathbf{W}^T\mathbf{h}$;
             Set $\mathbf{f} = \mathbf{b} - \mathbf{Q}\mathbf{h}$ ;  $\delta = \sqrt{\epsilon^2 - \mathbf{f}^T\mathbf{f}}$ ;
             $[\mathbf{z}, \lambda] = \mathrm{LNR\_SecEqnSol}(\mathbf{S}, \mathbf{b}_1, \delta, \tau)$;
             $\mathbf{x} = \lambda\mathbf{V}(\mathbf{U}\mathbf{S}\mathbf{z})$;
             $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$;
             $\mathbf{v} = \lambda\mathbf{A}^T\mathbf{r}$;
             $\mathbf{v} \leftarrow \mathbf{v} - \mathbf{V}(\mathbf{V}^T\mathbf{v})$;  $\mathbf{v} \leftarrow \mathbf{v}/\|\mathbf{v}\|$;
             $\mathbf{V} \leftarrow [\mathbf{V}, \mathbf{v}]$;
         end
```

Figure 4: Algorithm LNR_NLLx: Nonlinear Lanczos Solution-Space Iteration for LNR.

The new $\mathbf{r}_+, \lambda_+$ selected by Algorithm LNR_NLLr from this space must give descent from the point $\mathbf{r}, \lambda$ since $\mathbf{r} \in \mathcal{S}$. The new function value must satisfy $\varphi(\mathbf{r}_+, \lambda_+) < \varphi(\mathbf{r}, \lambda)$, since

$$\varphi(\mathbf{r}_+, \lambda_+) = \min \varphi(\mathbf{V}_+\hat{\mathbf{r}}_+, \lambda_+) < \min \varphi(\mathbf{V}\hat{\mathbf{r}}, \lambda) = \varphi(\mathbf{r}, \lambda).$$

Thus the algorithm is globally convergent in an iterative sense (it also has finite convergence) and can only halt with the exact solution (otherwise there is a nonzero step $\mathbf{s}$).

More generally, at a point $\mathbf{r} = \mathbf{V}\hat{\mathbf{r}}$, a descent direction is given by

$$\mathbf{s} = -\mathbf{M}[(\mathbf{I} + \lambda\mathbf{A}\mathbf{A}^T)\mathbf{r} - \mathbf{b}] = \mathbf{M}(\mathbf{b} - \mathbf{A}\mathbf{x} - \mathbf{r}),$$

where $\mathbf{M}$ is any symmetric positive definite matrix and $\mathbf{x} = \lambda\mathbf{A}^T\mathbf{r}$. Using the orthogonal decomposition given above, we have

$$\mathbf{s} = \mathbf{M}(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\mathbf{b} - \mathbf{A}\mathbf{x}).$$

To get $\mathbf{v}$, the direction $\mathbf{s}$ is orthogonalized against $\mathrm{Range}(\mathbf{V})$ and then normalized to have unit length. Thus $\mathbf{v}$ is in the direction

$$\hat{\mathbf{v}} = (\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{M}(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\mathbf{b} - \mathbf{A}\mathbf{x}).$$

This will give a pre-conditioned or Newton-like iteration. Again, when $\mathbf{v}$ is adjoined to the search space, the next iterate will give a decrease in the objective function that is at least as good as the Newton-like step since that step is contained in the search space.

Should $\hat{\mathbf{v}} = 0$ occur, the iteration will halt with the solution as before since this would imply

$$0 = (\mathbf{b} - \mathbf{A}\mathbf{x})^T\hat{\mathbf{v}} = (\mathbf{b} - \mathbf{A}\mathbf{x})^T(\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{M}(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\mathbf{b} - \mathbf{A}\mathbf{x}).$$

Thus $\mathbf{0} = (\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\mathbf{b} - \mathbf{A}\mathbf{x})$ which implies

$$\mathbf{b} - \mathbf{A}\mathbf{x} = \mathbf{V}\mathbf{z} \tag{7.1}$$

for some $\mathbf{z}$. Hence,

$$\mathbf{z} = \mathbf{V}^T(\mathbf{b} - \mathbf{A}\mathbf{x}) = \hat{\mathbf{b}} - \lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{r} = (\mathbf{I} + \lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V})\hat{\mathbf{r}} - \lambda\mathbf{V}^T\mathbf{A}\mathbf{A}^T\mathbf{V}\hat{\mathbf{r}} = \hat{\mathbf{r}}.$$

Substituting $\mathbf{z} = \hat{\mathbf{r}}$ into Equation (7.1) gives $\mathbf{b} - \mathbf{A}\mathbf{x} = \mathbf{r}$. Since $\|\mathbf{r}\| = \epsilon$ by construction, this implies that the KKT conditions will be satisfied indicating $\mathbf{x} = \lambda\mathbf{A}^T\mathbf{r}$ is the solution.

7.2 The solution space iteration

The analysis is similar to the one just given for the residual space iteration. We have demonstrated that the KKT conditions are equivalent to finding $\mathbf{x}, \lambda > 0$ such that

$$(\mathbf{I} + \lambda\mathbf{A}^T\mathbf{A})\mathbf{x} = \lambda\mathbf{A}^T\mathbf{b} \ \text{ with } \ \|\mathbf{r}\| = \epsilon \ \text{ and } \ \mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}.$$

For a given λ, the first equation amounts to solving the problem

$$\min_{\mathbf{x}}\{\frac{1}{2}\mathbf{x}^T(\mathbf{I} + \lambda\mathbf{A}^T\mathbf{A})\mathbf{x} - \lambda(\mathbf{A}^T\mathbf{b})^T\mathbf{x}\} \equiv \min_{\mathbf{x}} \varphi(\mathbf{x}, \lambda),$$

since $(\mathbf{I} + \lambda\mathbf{A}^T\mathbf{A})\mathbf{x} - \lambda\mathbf{A}^T\mathbf{b} = \nabla_{\mathbf{x}}\varphi(\mathbf{x}, \lambda)$. At a point $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$, the steepest descent direction is given by

$$\mathbf{s} = -[(\mathbf{I} + \lambda\mathbf{A}^T\mathbf{A})\mathbf{x} - \lambda\mathbf{A}^T\mathbf{b}] = (\lambda\mathbf{A}^T\mathbf{r} - \mathbf{x}).$$

This direction may be decomposed into orthogonal components

$$\mathbf{s} = \mathbf{V}\mathbf{V}^T\mathbf{s} + (\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{s} = \mathbf{V}[\lambda(\mathbf{A}\mathbf{V})^T\mathbf{b} - [\mathbf{I} + \lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})]\hat{\mathbf{x}}] + (\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}) = (\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}),$$

since

$$[\mathbf{I} + \lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})]\hat{\mathbf{x}} = \lambda(\mathbf{A}\mathbf{V})^T\mathbf{b}.$$

Hence, as before, we have that the step $\mathbf{s}$ used in Algorithm LNR_NLLx is in fact the steepest descent direction for φ at $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$ and λ. The vector $\mathbf{v} = \mathbf{s}/\|\mathbf{s}\|$ is adjoined to the search space $\mathcal{S} = \mathrm{Range}([\mathbf{V}, \mathbf{v}])$ and this space contains all points of the form $\mathbf{x}_\alpha = \mathbf{x} + \mathbf{x}\alpha$ and hence contains the minimum of φ along the steepest descent direction.

The new $\mathbf{x}_+, \lambda_+$ selected by Algorithm LNR_NLLx from this space must give descent from the point $\mathbf{x}, \lambda$ since $\mathbf{x} \in \mathcal{S}$. The new function value must satisfy $\varphi(\mathbf{x}_+, \lambda_+) < \varphi(\mathbf{x}, \lambda)$, since

$$\varphi(\mathbf{x}_+, \lambda_+) = \min \varphi(\mathbf{V}_+\hat{\mathbf{x}}_+, \lambda_+) < \min \varphi(\mathbf{V}\hat{\mathbf{x}}, \lambda) = \varphi(\mathbf{x}, \lambda).$$

Thus the algorithm is globally convergent in an iterative sense (it also has finite convergence) and can only halt with the exact solution (otherwise there is a nonzero step $\mathbf{s}$).

More generally, at a point $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$, a descent direction is given by

$$\mathbf{s} = -\mathbf{M}[(\mathbf{I} + \lambda\mathbf{A}^T\mathbf{A})\mathbf{x} - \lambda\mathbf{A}^T\mathbf{b}] = \mathbf{M}(\lambda\mathbf{A}^T\mathbf{r} - \mathbf{x}),$$

where $\mathbf{M}$ is any symmetric positive definite matrix. Using the orthogonal decomposition given above, we have

$$\mathbf{s} = \mathbf{M}(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}).$$

To get $\mathbf{v}$, the direction $\mathbf{s}$ is orthogonalized against $\mathrm{Range}(\mathbf{V})$ and then normalized to have unit length. Thus $\mathbf{v}$ is in the direction

$$\hat{\mathbf{v}} = (\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{M}(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}).$$

Again, this provides a pre-conditioned or Newton-Like step.

Should $\hat{\mathbf{v}} = 0$ occur, the iteration will halt with the solution as before since this would imply

$$0 = \lambda(\mathbf{A}^T\mathbf{r})^T\hat{\mathbf{v}} = \lambda(\mathbf{A}^T\mathbf{r})^T(\mathbf{I} - \mathbf{V}\mathbf{V}^T)\mathbf{M}(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r}).$$

Thus $\mathbf{0} = (\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\lambda\mathbf{A}^T\mathbf{r})$ which implies

$$\lambda\mathbf{A}^T\mathbf{r} = \mathbf{V}\mathbf{z}$$

for some $\mathbf{z}$. Hence,

$$\mathbf{z} = \mathbf{V}^T(\lambda\mathbf{A}^T\mathbf{r}) = \lambda\mathbf{V}^T\mathbf{A}^T(\mathbf{b} - \mathbf{A}\mathbf{x}) = \lambda(\mathbf{A}\mathbf{V})^T\mathbf{b} - \lambda(\mathbf{A}\mathbf{V})^T(\mathbf{A}\mathbf{V})\hat{\mathbf{x}} = \hat{\mathbf{x}}.$$

Therefore, $\lambda\mathbf{A}^T\mathbf{r} = \mathbf{x}$, and since $\|\mathbf{r}\| = \epsilon$ by construction, this implies that the KKT conditions will be satisfied by λ and $\mathbf{x} = \lambda\mathbf{A}^T\mathbf{r}$.

8 Numerical Results

In this section, we present numerical results to illustrate the performance of the proposed methods on regularization problems. The experiments were carried out in MATLAB R2008b on a MacBookPro with a 2.66 GHz processor and 4 GB of RAM, running the Mac OS X version 10.6.2 (Snow Leopard) operating system. The floating-point arithmetic was IEEE standard double precision with machine precision $2^{-52} \approx 2.2204 \times 10^{-16}$. We used problems from a standard test set [7]. In Sections 8.1 through 8.3, the data vector $\mathbf{b}$ was constructed as $\mathbf{b} = \mathbf{b}_o + \mathbf{n}$, with $\mathbf{b}_o$ the noise-free data vector (available from the test set), and $\mathbf{n}$ a fixed noise vector with normally-distributed components and such that $||\mathbf{n}||/||\mathbf{b}|| = 10^{-5}$. In Section 8.4, $||\mathbf{n}||/||\mathbf{b}|| = 10^{-2}$ and 10^{-3} were used. The exact solution to the inverse problem was available for all problems and is denoted here by $\mathbf{x}_*$.

8.1 LNR: the Newton iteration

In this section, we report results for the Newton iteration, LNR. We performed two sets of experiments on $m \times n$ matrices with $m = n = 300$ and $m = n = 1024$. The stopping criteria for Algorithm LNR_SecEqnSol (Figure 1) was $|\psi| \leq 1.5 \times 10^{-8}$ (square root of machine precision) and a maximum of 20 iterations. In Tables 1 and 2, we report the number of iterations (Iter), the average time in seconds for ten runs (Time), and the relative error in the regularized solution $\mathbf{x}$ with respect to $\mathbf{x}_*$. For all but one problem (**i_laplace, ex. 1**), we obtained very accurate regularized solutions with respect to $\mathbf{x}_*$ both in the 2-norm and visually. For problem **i_laplace, ex. 1**, the LNR solutions and the best truncated-SVD solutions have similar accuracy.

Problem	Iter	Time	$\frac{\|\|\mathbf{x}-\mathbf{x}_*\|\|}{\|\|\mathbf{x}_*\|\|}$
baart	12	0.18	5.39e-02
deriv2, ex. 1	8	0.18	7.51e-02
deriv2, ex. 2	8	0.18	7.24e-02
foxgood	10	0.16	2.26e-03
i_laplace, ex. 1	11	0.11	1.30e-01
i_laplace, ex. 3	10	0.11	1.93e-03
heat, mild	3	0.19	1.43e-04
heat, severe	8	0.20	8.72e-03
phillips	8	0.16	1.19e-03
shaw	10	0.16	3.18e-02

Table 1: LNR on Regularization Tools problems, $m = n = 300$.

Problem	Iter	Time	$\frac{\|\|\mathbf{x}-\mathbf{x}_*\|\|}{\|\|\mathbf{x}_*\|\|}$
baart	12	57.04	5.33e-02
deriv2, ex. 1	9	57.18	6.90e-02
deriv2, ex. 2	9	57.93	6.59e-02
foxgood	11	59.91	1.96e-03
i_laplace, ex. 1	12	23.04	1.67e-01
i_laplace, ex. 3	11	22.88	1.96e-03
heat, mild	4	60.96	1.13e-03
heat, severe	9	40.27	6.95e-03
phillips	9	46.97	1.32e-03
shaw	11	57.25	3.14e-02

Table 2: LNR on Regularization Tools problems, $m = n = 1024$.

8.2 LNR_NLLr and LNR_NLLx: the matrix-free iterations

In this section, we report results for the residual-space and solution-space, matrix-free iterations LNR_NLLr (Figure 3) and LNR_NLLx (Figure 4), respectively. We performed two sets of experiments using the same data as in Section 8.1 in order to compare the computed solutions with those obtained with the Newton iteration. The stopping criterion was $|\ ||\mathbf{Ax} - \mathbf{b}|| - \epsilon\ |/\epsilon \leq \tau$. The value of τ was chosen such that the relative error in $\mathbf{x}$ with respect to $\mathbf{x}_*$ was of the same order as for the LNR solution. In this case, $\tau = 10^{-1}$ was sufficient to achieve that goal for all but one method on one problem. For LNR_NLLx on problem **heat, severe**, $\tau = 10^{-2}$ was used. The value of $\epsilon = \|\mathbf{b} - \mathbf{b}_o\|$ was used for all problems except for problem **heat, mild**, for which $\epsilon = 2\|\mathbf{b} - \mathbf{b}_o\|$ was used. This is a good example of how the upper bound on the noise level can be advantageously used in the least-norm regularization approach. The number of vectors in the initial basis for the Krylov subspace was 11 for LNR_NLLr, and 21 for LNR_NLLx.

The results for LNR_NLLr and LNR_NLLx for $m = n = 300$ are reported in Tables 3 and 4, respectively. For $m = n = 1024$, the results are reported in Tables 5 and 6. In these tables and throughout the discussion, $\mathbf{x}$ shall denote the solution computed by the matrix free method LNR_NLL (r or x), $\mathbf{x}_{LNR}$ shall denote the solution computed by the dense Newton method LNR, and $\mathbf{x}_*$ shall denote the true solution. We report the number of outer iterations (OuIt), the average number of inner iterations, ie. iterations required by LNR on the projected problems (InIt), the number of matrix-vector products (MVP), the average time in seconds for ten runs (Time), the number of vectors (Vec), the relative error in the computed solution $\mathbf{x}$ with respect to $\mathbf{x}_{LNR}$, and the relative error in $\mathbf{x}$ with respect to $\mathbf{x}_*$.

The results show a reasonable agreement between $\mathbf{x}$ and $\mathbf{x}_{LNR}$, and regularized solutions that have the same accuracy as $\mathbf{x}_{LNR}$, and that are obtained at a low computational cost. Timings and number of vectors required by

LNR_NLLr and LNR_NLLx to solve problems of dimension 1024 are plotted in Figure 5 (note that abbreviations have been used for problem's names). The results seem to indicate an advantage for LNR_NLLr over LNR_NLLx for most problems, in terms of both time and storage.

In order to illustrate the savings obtained with the matrix-free iterations, Figure 6 shows the measured time and number of vectors as a percentage of the time used by LNR and of the full storage $n = 1024$, respectively. We can see that the time required by the matrix-free algorithms is under 1% of the time required by the Newton iteration, and that the storage requirements are under 5% of the problem size, and for many problems, around 2%.

We note also that in our implementation of the matrix-free algorithms, the SVD (and the QR factorization in LNR_NLLx) of matrices that depend on $\mathbf{V}$ is calculated from scratch at every iteration. An efficient implementation should update the previous SVD (and QR factorization in LNR_NLLx) since only one column is added to $\mathbf{V}$ at each iteration. We expect further savings in time with such an efficient implementation. We believe that savings in storage are also possible by changing the size of the initial basis. These issues require further investigation.

Problem	OuIt	InIt	MVP	Time	Vec	$\frac{\|\|\mathbf{x}-\mathbf{x}_{LNR}\|\|}{\|\|\mathbf{x}_{LNR}\|\|}$	$\frac{\|\|\mathbf{x}-\mathbf{x}_{*}\|\|}{\|\|\mathbf{x}_{*}\|\|}$
baart	1	12.0	35	0.04	12	9.08e-12	5.39e-02
deriv2, ex. 1	29	3.2	91	0.09	40	5.42e-03	7.50e-02
deriv2, ex. 2	27	3.2	87	0.09	38	5.95e-03	7.25e-02
foxgood	1	10.0	35	0.04	12	2.40e-13	2.26e-03
i_laplace, ex. 1	4	4.3	41	0.04	15	1.84e-02	1.36e-01
i_laplace, ex. 3	2	6.0	37	0.04	13	2.20e-03	3.05e-03
heat, mild	23	2.0	79	0.08	34	3.03e-04	3.35e-04
heat, severe	28	3.0	89	0.09	39	1.50e-03	8.80e-03
phillips	5	3.8	43	0.04	16	8.08e-04	1.30e-03
shaw	1	10.0	35	0.04	12	8.62e-10	3.18e-02

Table 3: LNR_NLLr on Regularization Tools problems, $m = n = 300$.

Problem	OuIt	InIt	MVP	Time	Vec	$\frac{\|\|\mathbf{x}-\mathbf{x}_{LNR}\|\|}{\|\|\mathbf{x}_{LNR}\|\|}$	$\frac{\|\|\mathbf{x}-\mathbf{x}_{*}\|\|}{\|\|\mathbf{x}_{*}\|\|}$
baart	1	7.0	67	0.04	22	1.33e-11	5.39e-02
deriv2, ex. 1	13	3.5	103	0.06	34	1.45e-02	7.95e-02
deriv2, ex. 2	12	3.5	100	0.06	33	1.40e-02	7.58e-02
foxgood	1	6.0	67	0.04	22	1.40e-10	2.26e-03
i_laplace, ex. 1	1	7.0	67	0.05	22	3.00e-07	1.30e-01
i_laplace, ex. 3	1	6.0	67	0.05	22	7.40e-09	1.93e-03
heat, mild	18	2.0	118	0.08	39	1.78e-04	2.25e-04
heat, severe	14	3.1	106	0.07	35	2.56e-03	9.97e-03
phillips	1	5.0	67	0.04	22	1.67e-05	1.18e-03
shaw	1	7.0	67	0.04	22	1.01e-11	3.18e-02

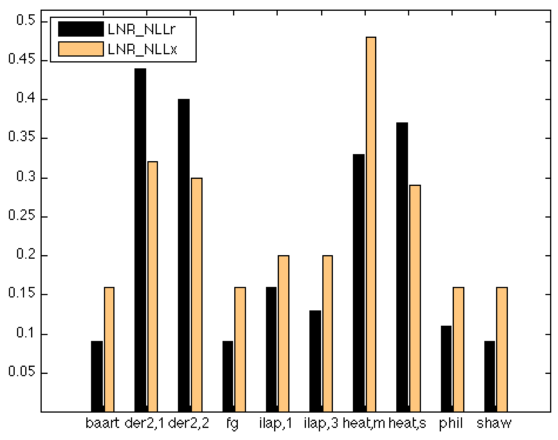
Table 4: LNR_NLLx on Regularization Tools problems, $m = n = 300$.

Problem	OuIt	InIt	MVP	Time	Vec	$\frac{\|\|\mathbf{x}-\mathbf{x}_{LNR}\|\|}{\|\|\mathbf{x}_{LNR}\|\|}$	$\frac{\|\|\mathbf{x}-\mathbf{x}_{*}\|\|}{\|\|\mathbf{x}_{*}\|\|}$
baart	1	12.0	35	0.09	12	1.51e-11	5.33e-02
deriv2, ex. 1	33	3.4	99	0.44	44	7.83e-03	6.96e-02
deriv2, ex. 2	31	3.4	95	0.40	42	8.16e-03	6.55e-02
foxgood	1	11.0	35	0.09	12	5.29e-13	1.96e-03
i_laplace, ex. 1	7	4.0	47	0.16	18	2.19e-02	1.72e-01
i_laplace, ex. 3	4	4.0	41	0.13	15	2.61e-03	3.16e-03
heat, mild	25	2.0	83	0.33	36	1.21e-03	5.50e-04
heat, severe	29	3.1	91	0.37	40	2.27e-03	7.50e-03
phillips	5	4.2	43	0.11	16	9.41e-04	1.41e-03
shaw	1	11.0	35	0.09	12	2.29e-09	3.14e-02

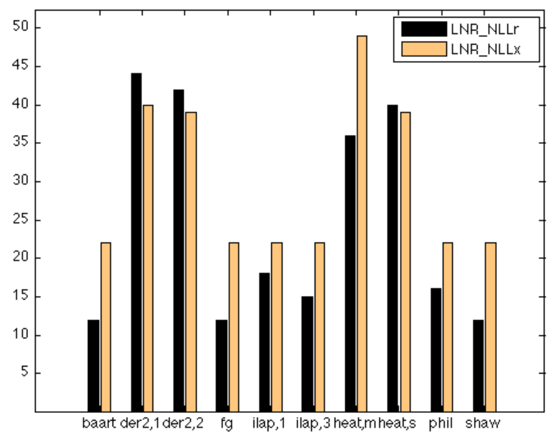
Table 5: LNR_NLLr on Regularization Tools problems, $m = n = 1024$.

Problem	OuIt	InIt	MVP	Time	Vec	$\frac{\|\|\mathbf{x}-\mathbf{x}_{LNR}\|\|}{\|\|\mathbf{x}_{LNR}\|\|}$	$\frac{\|\|\mathbf{x}-\mathbf{x}_{*}\|\|}{\|\|\mathbf{x}_{*}\|\|}$
baart	1	7.0	67	0.16	22	4.17e-11	5.33e-02
deriv2, ex. 1	17	3.3	115	0.32	38	1.58e-02	7.45e-02
deriv2, ex. 2	16	3.3	112	0.30	37	1.48e-02	7.08e-02
foxgood	1	6.0	67	0.16	22	3.67e-10	1.96e-03
i_laplace, ex. 1	1	7.0	67	0.20	22	1.66e-06	1.67e-01
i_laplace, ex. 3	1	6.0	67	0.20	22	2.68e-08	1.96e-03
heat, mild	27	2.0	145	0.48	48	1.14e-03	4.07e-04
heat, severe	15	3.1	109	0.29	36	3.49e-03	9.13e-03
phillips	1	5.0	67	0.16	22	2.89e-05	1.32e-03
shaw	1	7.0	67	0.16	22	1.65e-12	3.14e-02

Table 6: LNR_NLLx on Regularization Tools problems, $m = n = 1024$.

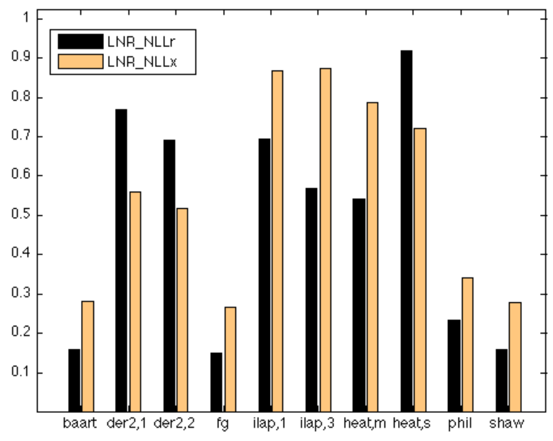


(a)

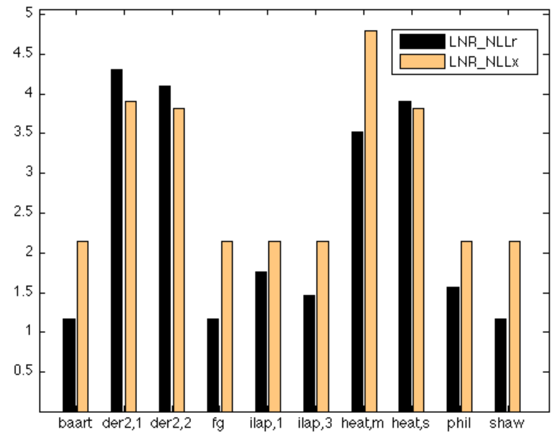


(b)

Figure 5: Time (a) and number of vectors (b) required by LNR_NLLr (dark) and LNR_NLLx (clear), $m = n = 1024$.



(a)



(b)

Figure 6: Percentage of LNR time (a) and of full storage $n = 1024$ (b) required by LNR_NLLr (dark) and LNR_NLLx (clear).

8.3 Convergence

In this section, we illustrate the convergence behavior of LNR, LNR_NLLr, and LNR_NLLx. We also use the opportunity to present results on problems with non-square matrices. The test problems were generated with the routine **heat** from [7] modified to construct non-square matrices. The mildly ill-conditioned problem was used ($\kappa = 5$ in **heat**). We generated two problems with $m \times n$ matrices with $m = 1024$, $n = 300$, and $m = 300$, $n = 1024$, respectively. In all tests, $\tau = 10^{-1}$ and $\epsilon = \|\mathbf{b} - \mathbf{b}_o\|$. The number of vectors in the initial basis for the Krylov subspace was 11 for LNR_NLLr, and 21 for LNR_NLLx.

The convergence plots are presented in Figures 7 and 8. Plots (a) in both figures illustrate the quadratic convergence of LNR, the Newton iteration. Plots (b) in both figures show that the LNR_NLLr iteration converges non-monotonically. The behavior of LNR_NLLx is shown in plots (c). For our two test problems, the convergence was monotonic. However, we have observed during our experiments that for lower tolerances (which require more iterations), roundoff error and noise will usually cause non-monotonic behavior in the last iterations.

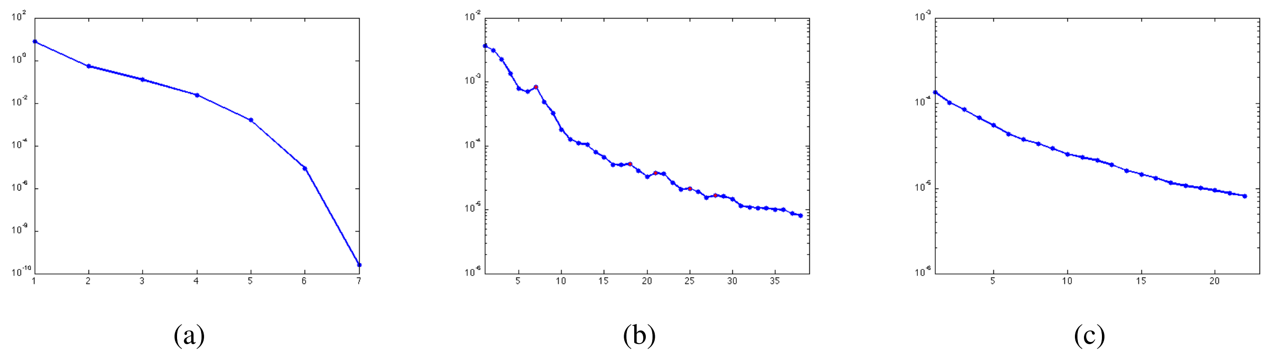


Figure 7: Convergence behavior for LNR (a), LNR_NLLr (b), and LNR_NLLx (c). Problem **heat, mild**, $m = 1024$, $n = 300$, x-axis: iteration index k, y-axis: $|\psi(\lambda_k)|$ (a) and $\|\mathbf{Ax}_k - \mathbf{b}\|$ (b) and (c).

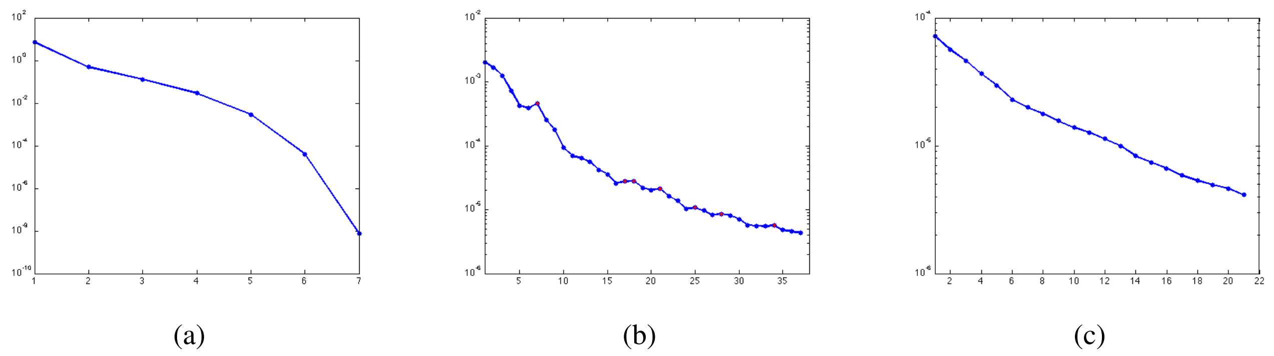


Figure 8: Convergence behavior for LNR (a), LNR_NLLr (b), and LNR_NLLx (c). Problem **heat, mild**, $m = 300$, $n = 1024$, x-axis: iteration index k, y-axis: $|\psi(\lambda_k)|$ (a) and $\|\mathbf{Ax}_k - \mathbf{b}\|$ (b) and (c).

The performance results are shown in Table 7. We observe comparable performance (time and storage) for LNR_NLLr and LNR_NLLx when the matrices are 1024×300. For 300×1024 matrices, the storage is comparable but LNR_NLLx requires half the time as LNR_NLLr. The times reported correspond to one run of each method.

Method / $m \times n$	OutIt	InIt	MVP	Time	Vec	$\frac{\|\|\mathbf{x}-\mathbf{x}_*\|\|}{\|\|\mathbf{x}_*\|\|}$
LNR / 1024×300	7	–	–	0.29	300	5.21e-03
LNR_NLLr / 1024×300	38	2.8	109	0.22	49	5.22e-03
LNR_NLLx / 1024×300	22	3.1	130	0.21	43	5.99e-03
LNR / 300×1024	7	–	–	0.30	1024	5.18e-03
LNR_NLLr / 300×1024	37	2.8	107	0.35	48	5.11e-03
LNR_NLLx / 300×1024	21	3.1	127	0.17	42	5.76e-03

Table 7: Performance of LNR, LNR_NLLr, and LNR_NLLx on **heat, mild** with rectangular matrices.

8.4 An Image Restoration Problem

In this section, we present a large-scale example from image restoration where we want to recover an image from blurred and noisy data. The problem was constructed in the following way. A digital photograph of an art gallery in Paris was blurred with the routine **blur** from [7]. Then, random Gaussian noise was added to the blurred image. The $n \times n$ matrix $\mathbf{A}$ was the blurring operator returned by **blur**. The data vector $\mathbf{b}$ was the blurred and noisy image stored as a one-dimensional array. The original photograph consisted of 256×256 pixels. Thus, the problem dimension was $n = 65536$. The noise level in $\mathbf{b} = \mathbf{b}_o + \mathbf{n}$ was $\|\mathbf{n}\|/\|\mathbf{b}_o\| = 10^{-2}$ as in [16], and also 10^{-3}. Here, $\mathbf{b}_o$ was a column version of the original image.

The following settings were used in algorithms LNR_NLLr and LNR_NLLx. The values $\tau = 10^{-1}$ and $\epsilon = 2\|\mathbf{b} - \mathbf{b}_o\|$ were used in all experiments. The number of vectors in the initial basis for the Krylov subspace was 11 for LNR_NLLr, and 21 for LNR_NLLx. The results are shown in Figure 9. The *waves* or *ripples* observed in the restorations are due to the famous Gibbs phenomenon (cf. [14]) and are characteristic of least squares restorations.

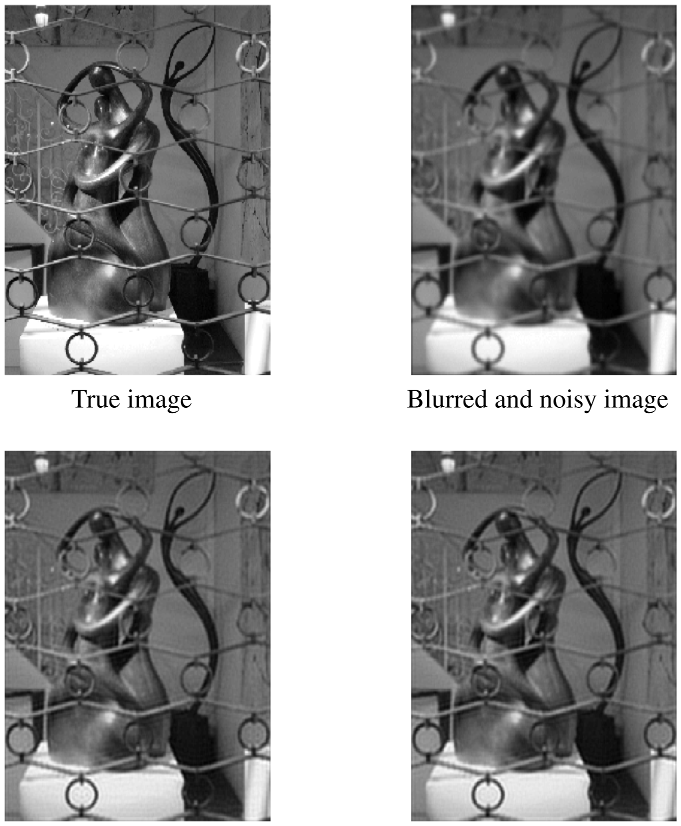
Table 8 shows the performance of LNR_NLLr and LNR_NLLx. For noise level 10^{-2}, both methods compute a solution with similar accuracy to the LSTRS solution (cf. [16]), using fewer matrix-vector products, and a moderate number of vectors. Table 8 also shows results for lower noise level. In this case, it is possible to obtain very good restorations. However, in general, low noise will make the regularization problem more difficult to solve (cf. [17]). In our tests, the new methods indeed computed better restorations both in terms of the 2-norm relative error and visually (not shown). The problem was considerably more challenging for LNR_NLLr than for LNR_NLLx. However, both methods were able to compute a solution with similar accuracy at a low cost.

Method / noise level	OutIt	InIt	MVP	Time	Vec	$\frac{\|\|\mathbf{x}-\mathbf{x}_*\|\|}{\|\|\mathbf{x}_*\|\|}$
LNR_NLLr / 10^{-2}	1	4.0	35	0.79	12	1.08e-01
LNR_NLLx / 10^{-2}	1	4.0	67	2.10	22	1.08e-01
LNR_NLLr / 10^{-3}	41	3.0	115	46.67	52	7.13e-02
LNR_NLLx / 10^{-3}	6	3.0	82	3.87	27	7.86e-02

Table 8: Performance of LNR_NLLr and LNR_NLLx on an Image Restoration Problem.

9 Concluding Remarks

In this work, we addressed the least-norm regularization problem, which can be seen as a perturbed least norm problem where the perturbation represents noise in the data. We have derived optimality (KKT) conditions for a solution which led to the associated secular equation. We then manipulated these conditions into various forms that provided a foundation for the development of two new iterative methods suitable for large-scale problems.



True image

Blurred and noisy image

LNR_NLLr restoration

LNR_NLLx restoration

Figure 9: Restoration of the photograph of an art gallery in Paris. Dimension: 65536, $\|\mathbf{b}\|/\|\mathbf{b}_o\| = 10^{-2}$.

We also showed that neither small singular values nor vanishing coefficients in the key rational function have any influence on the robustness or efficiency of our numerical solution of the secular equation. This is in significant contrast to the algorithmic difficulties in the standard trust-region subproblem associated with the so-called hard case (see [15, 16, 17]).

Our work has provided three algorithms: an SVD-based Newton iteration on the associated secular equation, well suited for small to medium scale dense problems, and two Nonlinear-Lanczos-based matrix-free iterations suited for large problems. The matrix-free techniques rely on the dense Newton iteration to solve a sequence of small projected problems. A brief analysis of these matrix-free iterations has shown that the basic iterative algorithms are closely related to a steepest descent iteration and that a pre-conditioner may easily be incorporated to provide a Newton-like iteration. The dense algorithm we present here was essentially given by Morozov [13] although we have made some improvements regarding implementation details. As far as we know, the large-scale iterative methods are new.

A limited set of numerical experiments were reported to illustrate the performance of the methods on a familiar test set of small-size problems taken from [7]. We also include computational results for a realistic large-scale image restoration problem. These results demonstrate that the new methods are robust, efficient, and accurate. Although more experimentation is needed, the preliminary results seem to indicate that the new techniques are efficient tools for the numerical treatment of discrete forms of ill-posed problems. The use of preconditioning in combination with the new methods is the subject of current research and shall be reported in future work.

Acknowledgment. M. Rojas would like to thank Professor Kees Roos of the Software Technology Department at the Delft University of Technology in The Netherlands for the hospitality during an extended visit during which most of this work was carried out. We would also like to thank Bill Symes for introducing us to the least norm regularization problem. Finally, we thank Heinrich Voss for providing the Nonlinear Lanczos/Arnoldi framework [20] that has turned out to be so effective for this problem as well as for the standard trust-region problem [9].

References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, 2004.

[2] L. Eldén. Algorithms for the regularization of ill-conditioned least squares problems. *BIT*, 17:134–145, 1977.

[3] R. Fletcher. *Practial Methods of Optimization*. John Wiley & Sons, Chichester, 1987.

[4] W. Gander. Least squares with a quadratic constraint. *Numer. Math.*, 36:291–307, 1981.

[5] G.H. Golub and U. von Matt. Quadratically constrained least squares and quadratic problems. *Numer. Math.*, 59:561–580, 1991.

[6] P.C. Hansen. *Rank-Deficient and Discrete Ill-Posed Problems, Numerical Aspects of Linear Inversion*. SIAM, Philadelphia, 1998.

[7] P.C. Hansen. Regularization Tools version 4.0 for Matlab 7.3. *Numer. Algo.*, 46:189–194, 2007.

[8] M.D. Hebden. An algorithm for minimization using exact second derivatives. Technical Report T.P. 515, Atomic Energy Research Establishment, Harwell, England, 1973.

[9] J. Lampe, M. Rojas, D.C. Sorensen, and H. Voss. Accelerating the LSTRS Algorithm. Technical Report 138, Institute of Numerical Simulation, Hamburg University of Technology, Hamburg, July 2009.

[10] G. Landi. The Lagrange method for the regularization of discrete ill-posed problems. *Comput. Optim. Appl.*, 39:347–368, 2008.

[11] J.J. Moré. The Levenberg-Marquardt algorithm: implementation and theory. In G. A. Watson, editor, *Numerical Analysis Proceedings Dundee 1977, Lecture Notes in Mathematics 630*, pages 144–157. Springer-Verlag, Berlin, 1977.

[12] J.J. Moré and D.C. Sorensen. Computing a trust region step. *SIAM J. Sci. Stat.Comput.*, 4(3):553–572, 1983.

[13] V.A. Morozov. *Methods for solving incorrectly posed problems*. Springer-Verlag, New York, 1984.

[14] T.W. Parks and C.S. Burrus. *Digital Filter Design*. John Wiley & Sons, New York, 1987.

[15] M. Rojas. A large-scale trust-region approach to the regularization of discrete ill-posed problems. Ph.D. thesis, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, Technical Report TR98-19, May 1998.

[16] M. Rojas, S.A. Santos, and D.C. Sorensen. Algorithm 873: LSTRS: MATLAB software for large-scale trust-region subproblems and regularization. *ACM Trans. Math. Software*, 34(2):11, 2008.

[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. Comput.*, 23(6):1843–1861, 2002.

[18] Gerard L. G. Sleijpen and Henk A. Van der Vorst. A jacobi–davidson iteration method for linear eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications*, 17(2):401–425, 1996.

[19] W.W. Symes. Extremal regularization. Technical Report TR99-07, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1999.

[20] H. Voss. An Arnoldi method for nonlinear eigenvalue problems. *BIT*, 44:387–401, 2004.