ACCELERATING THE LSTRS ALGORITHM

J. LAMPE *, M. ROJAS †, D.C. SORENSEN ‡, AND H. VOSS §

Abstract. In a recent paper [Rojas, Santos, Sorensen: ACM ToMS 34 (2008), Article 11] an efficient method for solving the Large-Scale Trust-Region Subproblem was suggested which is based on recasting it in terms of a parameter dependent eigenvalue problem and adjusting the parameter iteratively. The essential work at each iteration is the solution of an eigenvalue problem for the smallest eigenvalue of the Hessian matrix (or two smallest eigenvalues in the potential hard case) and associated eigenvector(s). Replacing the implicitly restarted Lanczos method in the original paper with the Nonlinear Arnoldi method makes it possible to recycle most of the work from previous iterations which can substantially accelerate LSTRS.

Key words. constrained quadratic optimization, regularization, trust-region, ARPACK, Nonlinear Arnoldi method

AMS subject classification. 65F15, 65F22, 65F30

1. Introduction.

We consider the Large-Scale Trust-Region Subproblem (LSTRS problem) to minimize a quadratic function subject to a spherical constraint

$$\min_x \psi(x) := \frac{1}{2}x^T Hx + g^T x \quad \text{subject to } \|x\| \leq \Delta \tag{1.1}$$

where $H = H^T \in \mathbb{R}^{n\times n}$, $g \in \mathbb{R}^n$ and $\Delta > 0$ are given.

Problem (1.1) arises for example in connection with the trust region globalization strategy in optimization. A special case of (1.1) is the least squares problem with a norm constraint which is equivalent to Tikhonov regularization for discrete forms of ill-posed problems.

If it is possible to compute the Cholesky factorization of matrices of the form $H - \lambda I$ the method of choice is the one proposed by Moré and Sorensen [10] which uses Newton's method to find a root of a scalar function. This is reasonable if the matrix H is not too large and given explicitly.

Hager in [2] introduced the sequential subspace method (SSM). A sequence of four dimensional subspaces $\mathcal{S}_k$ is generated that act as additional constraint $x \in \mathcal{S}_k$ for (1.1). (Here the inequality has to be replaced by the equality constraint $\|x\| = \Delta$). The ingredients of the subspaces are the current iterate x_k, the gradient of $\psi(x)$ at x_k, an estimate of the smallest eigenvector of H and an approximation of the SQP iterate (i.e. Newton's method applied to the first-order optimality system). The first three vectors are sufficient for linear global convergence, proven by Hager and Park in [3]. By inserting the SQP iterate into the search space the convergence is locally quadratic. The calculation of an SQP iterate exhibits a strong connection to the Jacobi-Davidson algorithm of Sleijpen and Van der Vorst [18]. The main costs

*Institute of Numerical Simulation, Hamburg University of Technology, D-21071 Hamburg, Germany (joerg.lampe@tu-harburg.de). The work was supported by Philips Research Germany and the German Federal Ministry of Education and Research (BMBF) under grant number 13N9079

†Department of Informatics and Mathematical Modelling, Technical University of Denmark, DK-2800 Kongens Lyngby (mr@imm.dtu.dk)

‡Dept. of Computational & Applied Mathematics, Rice University, Houston, TX 77005-1892, USA (sorensen@caam.rice.edu). The work was supported in part by NSF grants CCR-9988393 and ACI-0082645, and by AFOSR grant FA9550-09-1-0225.

§Institute of Numerical Simulation, Hamburg University of Technology, D-21071 Hamburg, Germany (voss@tu-harburg.de)

of SSM are the approximate solutions of a sequence of linear systems. These linear systems are treated independently by variants of MINRES, i.e. there is no reuse of information from previous iteration steps except for the current iterate x_k itself.

The approach of Sorensen in [21] involved the solution of a related parametric eigenvalue problem. The parameter is adjusted within a sequence of eigenproblems since the solution does fulfill the bound condition. At the same time Rendl and Wolkowicz [12] came out with a very similar idea using the same parametric eigenvalue problems. The resulting algorithms are different in spirit: While Sorensen attempts to satisfy the optimality condition, Rendl and Wolkowicz try to reduce the duality gap in a primal-dual semidefinite programming framework.

In a nice survey of Fortin and Wolkowicz [1], the Generalized Lanczos Trust Region Algorithm was compared to the above mentioned algorithm in [10] and an improved version of the algorithm in [12].

The drawback of switching the algorithms in the hard case in [21] has been analyzed and removed by Rojas, Santos and Sorensen and they suggested in a recent paper [15] the LSTRS algorithm.

The LSTRS method is based on a reformulation of (1.1) as a parameter dependent eigenvalue problem

$$B_\alpha y := \begin{pmatrix} \alpha & g^T \\ g & H \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix} = \lambda \begin{pmatrix} 1 \\ x \end{pmatrix} \tag{1.2}$$

where the real parameter α has to be adjusted such that the solution of (1.1) can be read off from the smallest eigenvalue of (1.2) and the (appropriately normalized) corresponding eigenvector. To this end one has to solve a sequence of eigenvalue problems $B_{\alpha_k} y_k = \lambda_k y_k$ for the smallest eigenpair (λ_k, y_k) (and for a second eigenpair in the potential hard case) which is the essential cost of every step of the LSTRS method.

In [15] the eigenvalue problems are solved by the implicitly restarted Lanczos method [19] implemented in ARPACK [9] and included into MATLAB as function `eigs`. As the sequence of parameters $\{\alpha_k\}$ produced by the LSTRS algorithm converges to some α^*, the matrices B_{α_k} converge as well. Therefore, it should be beneficial to reuse information from previous steps when solving $B_{\alpha_k} y = \lambda y$ in the current step. Recycling of prior information can be accomplished in ARPACK by choosing the eigenvector of the last step corresponding to the smallest eigenvalue as initial vector in the next step. However, this usually does not significantly speed convergence.

An eigensolver that is able to use all information acquired in previous iteration steps is the the Nonlinear Arnoldi method which was introduced in [22] for solving nonlinear eigenvalue problems. As an iterative projection method it determines an approximation to an eigenpair from the projection $V^T B_{\alpha_k} V z = \lambda z$ of the eigenproblem to a subspace V of small dimension, and it expands V if the approximation does not meet a specified accuracy. It is obvious that these projected problems can be updated and reused without further cost when changing the parameter α. In this paper we discuss this modification of the LSTRS algorithm and we demonstrate its capability to accelerate the LSTRS approach with several examples. Note that the algorithms of [12] and the improved version in [1] could be accelerated in exactly the same way as presented here for LSTRS.

The paper is organized as follows. Section 2 briefly sketches the relation between the LSTRS problem (1.1) and the parametrized eigenvalue problem (1.2) and the approach of the LSTRS algorithm. In Section 3 we describe the Nonlinear Arnoldi

method and how it is included into the LSTRS algorithm, and Section 4 demonstrates the improvement of LSTRS by recycling of prior information with several examples.

2. The LSTRS method. In this section we briefly describe the LSTRS method. Its theoretical foundation and a discussion of its convergence properties is contained in [13, 14, 16], and a detailed description of the LSTRS algorithm with special emphasis on computational aspects is presented in [15].

The trust-region subproblem (1.1) always has a solution. The following characterization was proved in [20].

LEMMA 2.1. *A feasible vector x^* is a solution to* (1.1) *with corresponding Lagrange multiplier λ^* if and only if x^* and λ^* satisfy $(H-\lambda^* I)x^* = -g$ with $H-\lambda^* I$ positive semidefinite, $\lambda^* \leq 0$, and $\lambda^*(\Delta - \|x^*\|) = 0$.*

Lemma 2.1 implies that all solutions of the trust-region subproblem are of the form $x^* = -(H-\lambda^* I)^\dagger g + z$, for some $z \in \mathcal{N}(H-\lambda^* I)$ where $A^\dagger$ denotes the pseudo inverse of a matrix A and $\mathcal{N}(A)$ the null space of A. If the Hessian matrix H is positive definite and if it holds that $\|H^{-1}g\| < \Delta$, problem (1.1) has a unique interior solution given by $x^* = -H^{-1}g$, with Lagrange multiplier $\lambda^* = 0$. If H is positive semidefinite or indefinite, there exist boundary solutions satisfying $\|x^*\| = \Delta$ and $\lambda^* \leq \delta_1 \leq 0$ where δ_1 denotes the smallest eigenvalue of H. In this paper, we only consider the case that there exists a boundary solution.

Lemma 2.1 reveals the relationship between the trust-region subproblem (1.1) and the eigenvalue problem (1.2): For real α let $\lambda_1(\alpha)$ be the smallest eigenvalue of B_α and y be a corresponding eigenvector.

We first consider the case that the first component of y is different from 0, and can be scaled to be equal to one. For such an eigenvector $y = (1, x^T)^T$ we have

$$\alpha - \lambda_1(\alpha) = -g^T x, \tag{2.1}$$

and

$$(H - \lambda_1(\alpha) I)x = -g. \tag{2.2}$$

Equation (2.2) demonstrates that two of the conditions of Lemma 2.1 are automatically satisfied: $(H-\lambda_1(\alpha)I)x = -g$, and since the eigenvalues of H interlace those of B_α, $H-\lambda_1(\alpha)I$ must be positive semidefinite. Therefore, the parameter α simply must be adjusted to satisfy the two remaining properties $\lambda_1(\alpha) \leq 0$ and $\lambda_1(\alpha)(\Delta - \|x\|) = 0$.

Taking advantage of (2.1) α can be updated according to

$$\alpha_+ = \lambda_1(\alpha) - g^T x = \lambda_1(\alpha) + g^T (H - \lambda_1(\alpha) I)^\dagger g =: \lambda_1(\alpha) + \phi(\lambda_1(\alpha)).$$

$\phi(\lambda) = g^T (H - \lambda I)^\dagger g$ is a rational function with possible poles at the eigenvalues of H which is too expensive to evaluate. LSTRS therefore employs a rational interpolate $\hat{\phi}(\lambda)$ of $\phi(\lambda)$ based on the Hermitian data $\phi(\lambda) = -g^T x$ and $\phi'(\lambda) = g^T((H-\lambda I)^\dagger)^2 g = x^T x$. The new value α_+ is then computed as $\alpha_+ = \hat{\lambda} + \hat{\phi}(\hat{\lambda})$ where $\hat{\lambda}$ is determined such that $\hat{\phi}'(\hat{\lambda}) = \Delta^2$.

The previous approach assumes that there exists an eigenvector corresponding to the smallest eigenvalue of B_α the first component of which can be scaled to one. It breaks down if all eigenvectors associated with $\lambda_1(\alpha)$ have first component zero. This can only happen when g is orthogonal to the eigenspace $\mathcal{S}_1$ of H corresponding to δ_1. If $g \perp \mathcal{S}_1$ there is the possibility for the occurrence of the hard case (cf. [10]), and therefore this situation is called potential hard case [14].

It was shown in [14] that in a potential hard case for all values of α greater than a certain critical $\tilde{\alpha}$ all eigenvectors associated with $\lambda_1(\alpha)$ have first component zero. But for any α there is a well defined eigenvector of B_α depending continuously on α that can safely be normalized to have first component one. If $g \not\perp \mathcal{S}_1$ or $g \perp \mathcal{S}_1$ and $\alpha \leq \tilde{\alpha}$ then this eigenvector corresponds to the smallest eigenvalue $\lambda_1(\alpha)$, and if $g \perp \mathcal{S}_1$ and α exceeds $\tilde{\alpha}$ by a small amount it is associated with the second smallest eigenvalue. It is this eigenpair which is used in LSTRS to construct the rational Hermitian interpolation $\hat{\phi}$ mentioned earlier.

Above we sketched the essential ingredients of the LSTRS method demonstrating that the main cost in every iteration step is the solution of the eigenvalue problem (1.2) with fixed α for the smallest eigenvalue or the two smallest eigenvalues in the potential hard case and a corresponding eigenvector. The real software uses a safeguarding strategy to ensure the global convergence of $\{\alpha_k\}$ to its optimal value, and it employs the eigenpairs $(\lambda_1(\alpha_k), x_k)$ and $(\lambda_1(\alpha_{k-1}), x_{k-1})$ from the last two iteration steps when constructing the rational interpolation $\hat{\phi}$ to generate the next parameter $\alpha_{k+1} = \hat{\lambda} + \hat{\phi}(\hat{\lambda})$. It was shown in [14], Theorem 5.1 that the resulting algorithm is superlinearly convergent.

3. Nonlinear Arnoldi. Solving a large-scale symmetric eigenvalue problem for the smallest or two smallest eigenvalues and corresponding eigenvectors the method of choice is the implicitly restarted Lanczos method. However, in LSTRS a sequence of eigenproblems depending continuously on a convergent parameter has to be solved, and one should take advantage of information gained in previous iteration steps. The only freedom of Krylov subspace methods like Lanczos is the choice of the initial vector, and consequently in the original LSTRS method the kth iteration step is initialized by the eigenvector of the $(k-1)$th step.

An eigensolver that is able to make use of all information from previous iteration steps is the Nonlinear Arnoldi method, which was introduced in [22] for solving nonlinear eigenvalue problems. As an iterative projection method it computes an approximation to an eigenpair from a projection to a subspace of small dimension, and it expands the subspace if the approximation does not meet a given accuracy requirement. These projections can be easily reused when changing the parameter α_k.

Although the Nonlinear Arnoldi method is usually more expensive than the implicitly restarted Lanczos method when solving a single eigenproblem the heavy reuse of information gained from previous steps leads to a significant speed-up in LSTRS, since almost all necessary information for solving $B_{\alpha_{k+1}} y = \lambda y$ is already contained in the subspace, that has been built up for solving $B_{\alpha_k} y = \lambda y$.

In regularized total least squares (RTLS) a similar technique has been successfully applied in [5, 7] for accelerating the RTLS solver in [17] which is based on a sequence of quadratic eigenvalue problems. Another method for RTLS presented in [11] which is based on a sequence of linear eigenproblems has also been accelerated substantially in [6, 8].

The following algorithm is used in LSTRS for solving

$$T_k(\lambda)y = (B_0 + \alpha_k N - \lambda I)y = \left(\begin{pmatrix} 0 & g^T \\ g & H \end{pmatrix} + \alpha_k e_1 e_1^T - \lambda I \right) y = 0$$

Algorithm 1 Nonlinear Arnoldi

1: Start with initial basis V, $V^TV = I$
2: For fixed α_k find smallest eigenvalue μ of $V^T T_k(\mu)Vz = 0$ and corresponding eigenvector z
3: Determine preconditioner $PC \approx T_k^{-1}(\mu)$
4: Set $u = Vz$, $r = T_k(\mu)u$
5: **while** $\|r\|/\|u\| > \epsilon$ **do**
6: $v = PCr$
7: $v = v - VV^Tv$
8: $\tilde{v} = v/\|v\|$, $V = [V, \tilde{v}]$
9: Find smallest eigenvalue μ of $V^T T_k(\mu)Vz = 0$ and corresponding eigenvector z
10: Set $u = Vz$, $r = T_k(\mu)u$
11: **end while**

The Nonlinear Arnoldi method allows thick starts in line 1, i.e. when solving $T_k(\lambda)y = 0$ in step k of the LSTRS method, algorithm 1 is started with the orthonormal basis V that was used in the preceding iteration step when determining the solution $y_{k-1} = Vz$ of $V^T T_{k-1}(\lambda)Vz = 0$. So all search spaces of previous problems are kept.

The projected problem in the kth iteration step

$$V^T T_k(\mu)Vz = \left(V^T B_0 V + \alpha_k (e_1^T V)^T (e_1^T V) - \mu I\right) z = 0 \tag{3.1}$$

can be achieved immediately from the previous step since the matrices V, V^TB_0V and $v_1 = V(1,:)$ are known. Within the iteration these matrices are obtained on-the-fly by appending one column to V and one column and row to V^TB_0V, respectively. This update relies on matrix-vector products only, and does not require the matrix B_0 explicitly.

At Step 7, numerical orthogonality of v with respect to V is enforced by a reorthogonalization step if necessary. If $v = 0$ after Step 7 is completed, the iteration must halt. However, this is a fortunate event as it would imply $r = 0$. This is because $v = 0$ would imply $Vs = PCr$ for some s. Hence, $r^TPCr = r^TVs = 0$ by virtue of Step 9 which in turn implies $r = 0$ since PC is positive definite. Therefore, the test at Step 5 would have already halted the iteration with a numerical solution.

The LSTRS method with Nonlinear Arnoldi can be executed with low and fixed storage requirements. For memory allocation purposes a maximal dimension $p \ll n$ of the search space span(V) can be set in advance, and if in the course of the LSTRS method the dimension of V reaches p, then the Nonlinear Arnoldi method can be restarted with a subspace spanned by a small number q of eigenvectors corresponding to the smallest eigenvalues of $T_k(\lambda)$.

LSTRS requires an eigenvector associated with the smallest eigenvalue of $T_k(\lambda)$. An additional eigenvector is needed only if the first eigenvector cannot be safely scaled to have one as a first component The implicitly restarted Lanczos method approximates eigenvectors corresponding to extreme eigenvalues simultaneously, and therefore in the original version of LSTRS two eigenvectors are returned by `eigs` in every iteration step. The Nonlinear Arnoldi method aims at one eigenpair at a time. In algorithm 1 this is the smallest one. If a second eigenvector is needed the search space V can be further extended now aiming at the second smallest eigenvalue μ in statement 9.

A few further comments are in order:

— To solve the very first eigenproblem with the Nonlinear Arnoldi it is recommended to put some useful information in the starting basis V. An orthonormal basis of the Krylov subspace $\mathcal{K}_\ell(B_0, e_1)$ with a small value of $\ell \approx 5$ is a suitable choice.
— The projected eigenproblems in line 2 and 9 can be solved by a dense solver for all eigenvalues; in the examples MATLABs `eig` has been used.
— For general nonlinear eigenproblems the nonlinear Arnoldi usually requires a preconditioner. For this application, PC would need to be positive definite if it is to approximate the inverse of a positive definite matrix, $PC \approx T_k^{-1}(\mu)$. For the test examples in Section 4 the method always worked fine without it, i.e. $PC = I$.
— For a least squares problem with norm constraint the explicit form of the matrix $H = A^T A$ is not needed to determine the projected matrix $V^T B_0 V$, but it can be updated according to $V^T B_0 V = ([-b, A]V)^T([-b, A]V) - \|b\|^2 v_1^T v_1$ (recalling that v_1 is the first row of V).

The advantage of using the Nonlinear Arnoldi method in LSTRS over the implicitly restarted Lanczos method is due to the fact that thick starts are possible. This holds true also for other iterative projection approaches like the Jacobi–Davidson method where the search space $\mathrm{span}(V)$ is expanded by an approximate solution of the correction equation

$$\big(I - \frac{uu^T}{u^T u}\big) T_k(\mu) \big(I - \frac{uu^T}{u^T u}\big) v = T_k(\mu) u, \quad v \perp u. \tag{3.2}$$

However, solving (3.2) will usually be much more expensive than the Nonlinear Arnoldi expansion $v = PC \cdot T_k(\mu) u$. For really huge problems where storage is critical and only coarse preconditioners are available, Jacobi–Davidson could be beneficial because the correction equation can be solved by a Krylov solver with short recurrence.

4. Numerical examples. In order to evaluate the performance of LSTRS using Nonlinear Arnoldi as eigensolver, we used a modified version of LSTRS where the second eigenpair is calculated only when needed. In this case, the Nonlinear Arnoldi method is called again and a second eigenpair is computed.

Numerical experiments were performed on different problem classes. The first class consisted of regularization problems. For those problems, the matrix H is usually close to singular and the vector g is nearly orthogonal to $\mathcal{S}_1$. Therefore, a (near) potential hard case is always present and, depending on the value of Δ, the hard case is also present.

The second problem class consisted of shifted Laplacian problems with Hessian matrix $H = L - 5I$, where L is a discrete version of the 2-D Laplacian. The diagonal shift makes H indefinite. Both the easy and the hard case were considered.

In the third problem class, the matrix H was of the form $H = UDU^T$, with D a diagonal matrix and U a Householder matrix. Due to the complete knowledge of H, all cases can be very easily generated and investigated.

We studied the performance of LSTRS when different eigensolvers were used. For the regularization problems, we compared the following eigensolvers. IRLM+H: the IRLM combined with the heuristics described in [15]. IRLM+T: the IRLM combined with a Tchebyshev Spectral Transformation as described in [16]. NLArn: the Nonlinear Arnoldi method described in Section 3. For the other two classes of problems, we compared the IRLM with NLArn. The specific LSTRS settings are described under each class of problems.

The numerical tests were run under MATLAB R2008a on a PentiumR4 computer with 3.4 GHz and 8GB RAM, running the Linux operating system.

4.1. Regularization Problems. The regularization problems were taken from the Regularization Tools package [4]. Most of the problems in this package are discretizations of Fredholm integral equations of the first kind, which are typically very ill-conditioned.

Regularized solutions were computed by solving the following quadratically-constrained least squares problem:

$$\min_x \frac{1}{2}\|Ax - b\|^2 \quad \text{subject to} \quad \|x\| \leq \Delta, \tag{4.1}$$

where $A \in \mathbb{R}^{m\times n}$, $m \geq n$, and $b \in \mathbb{R}^m$. The matrix A comes from the discretization of an operator in an ill-posed problem and is typically very ill-conditioned. Problem (4.1) is equivalent to a trust-region problem of type (1.1) with $H = A^TA$ and $g = -A^Tb$.

The MATLAB routines `heat, ilaplace, parallax, phillips` and `shaw` from [4] provided the matrices A and the right-hand sides b. Except for `parallax`, the routines also provided the true solutions x_{true} and $\Delta = \|x_{true}\|$ was used for those problems. For problem *parallax*, $\Delta = 5$ was used.

No noise was added to the vector b since the absence of noise yields a more difficult trust-region problem for which the potential (near) hard case is present in a multiple instance (cf. [15, 16]). The parameters for LSTRS were chosen as in [15] and were as follows. The values `epsilon_HC = 1e-16` and `epsilon_Int = 0` were used to make a boundary solution more likely. The values `epsilon_Delta = 1e-2` and `max_eigentol = 0.4` were used in all experiments, except for the mildly ill-posed *heat* problem for which `epsilon_Delta = 1e-3` and `max_eigentol = 0.7`. The initial vector for the first call to the IRLM was $v_0 = (1, \ldots, 1)^T/\sqrt{n+1}$ and the total number of vectors was fixed at 8. For NLArn, the maximum dimension of the search space was $p = 40$, the dimension of the starting search space was $\ell = 5$, and we restarted with $q = 2$ eigenvectors corresponding to the two smallest eigenvalues of the projected problem. The results are shown in Table 4.1.

In Table 4.1, the size of the problem (n) is given in parentheses and 'MatVecs' denotes the number of matrix-vector products required. Here, $A \in \mathbb{R}^{n\times n}$, $x, b \in \mathbb{R}^n$. The only exception is problem *parallax*, for which $A \in \mathbb{R}^{26\times n}$, $b \in \mathbb{R}^{26}$. For problem *heat*, a parameter κ controls the degree of ill-posedness. The value $\kappa = 5$ yields a mildly ill-posed problem whereas $\kappa = 1$ generates a severely ill-posed one.

The maximum dimension of the search space in the Nonlinear Arnoldi method ($p = 40$) was reached only when solving the mildly ill-posed *heat* problem. In this case, one restart was necessary. In all the regularization tests, the Nonlinear Arnoldi method outperformed the Implicitly Restarted Lanczos Method in terms of matrix-vector products and optimality of the trust-region solution measured by the quantity $\frac{\|H-\lambda I+g\|}{\|g\|}$. Roughly speaking, the effort was reduced by a factor of 10 with respect to the IRLM+H while the optimality level was higher. The IRLM+T needed even more MatVecs than the IRLM+H to compute solutions with similar optimality levels.

4.2. Laplacian Problems. In these problems, the Hessian matrix was $H = L - 5I$, with L the standard 2-D discrete Laplacian on the unit square based upon a 5-point stencil with equally-spaced mesh points. We studied problems of size 324 and 1024. The vector g was randomly generated with entries uniformly distributed on $(0, 1)$. The trust-region radius was fixed at $\Delta = 100$. Problems with and without

TABLE 4.1
Problems from Regularization Tools

Problem(Size)	Eigensolver	$\frac{\|x-x_{true}\|}{\|x_{true}\|}$	$\frac{\|H-\lambda I+g\|}{\|g\|}$	$\frac{\|\|x\|-\Delta\|}{\Delta}$	MatVecs
heat(300),$\kappa = 1$	IRLM+H	5.0e-02	5.3e-06	3.6e-03	554
	IRLM+T	5.0e-02	3.1e-04	1.2e-03	869
	NLArn	5.1e-02	3.1e-11	9.5e-03	33
heat(300),$\kappa = 5$	IRLM+H	3.3e-05	1.2e-07	1.2e-07	330
	IRLM+T	3.3e-02	1.5e-12	2.9e-02	2231
	NLArn	1.3e-03	3.8e-06	3.3e-06	64
heat(1000),$\kappa = 1$	IRLM+H	5.5e-02	7.0e-06	4.3e-03	552
	IRLM+T	7.0e-02	7.3e-05	4.8e-03	1029
	NLArn	1.2e-02	1.9e-09	3.0e-04	37
ilaplace(195)	IRLM+H	2.8e-01	2.4e-04	3.2e-16	304
	IRLM+T	1.5e-01	4.1e-08	1.4e-02	1909
	NLArn	1.2e-01	1.1e-14	8.7e-03	22
parallax(300)	IRLM+H	-	1.4e-06	2.5e-03	335
	IRLM+T	-	8.4e-05	9.1e-03	953
	NLArn	-	3.4e-15	8.7e-03	23
phillips(300)	IRLM+H	3.4e-02	1.2e-05	1.6e-04	213
	IRLM+T	1.3e-02	2.2e-06	8.9e-04	393
	NLArn	1.0e-02	1.6e-13	5.2e-04	26
phillips(1000)	IRLM+H	1.0e-02	1.4e-06	4.5e-04	252
	IRLM+T	1.4e-02	3.5e-06	9.6e-04	393
	NLArn	1.0e-02	2.1e-13	5.6e-04	26
shaw(300)	IRLM+H	5.8e-02	7.2e-09	5.6e-03	247
	IRLM+T	5.8e-02	7.2e-09	5.6e-03	873
	NLArn	5.8e-02	5.3e-14	5.7e-03	17
shaw(1000)	IRLM+H	8.4e-02	2.3e-09	9.9e-03	223
	IRLM+T	8.4e-02	2.3e-09	9.9e-03	792
	NLArn	5.9e-02	2.2e-13	5.8e-03	17

hard case were studied. To generate the hard case, the vector g was orthogonalized against the eigenvector q_1 corresponding to the smallest eigenvalue of H. Hence, the easy case was present when g was not orthogonalized against q_1. A noise vector of norm 10^{-8} was added to g.

The LSTRS parameter `max_eigentol = 0.02` was used in all experiments. For $n = 324$, `epsilon_HC = 1e-5` and `epsilon_Delta = 1e-5` were used. For $n = 1024$, the parameters were `epsilon_Delta = 1e-5` and `epsilon_HC = 1e-11` in the easy case, and `epsilon_Delta = epsilon_HC = 1e-11` in the hard case, cf. choices in [15]. The initial vector for the first call to the IRLM was $v_0 = (1, \ldots, 1)^T/\sqrt{n+1}$ and the total number of vectors was fixed at 10. For NLArn, $p = 40, \ell = 5, q = 2$ were used in the easy case. In the hard case, $p = 40, \ell = 20, q = 11$ were used.

Average results for ten related problems, differing only in the vector g, are shown in Table 4.2. The quantity $|\lambda^* - d_1|$ is a measure of how exactly the hard case was hit. In the easy case, this value is not meaningful. In the hard case, $\lambda^* = \delta_1$ and therefore, the exact value should be zero.

In the easy case, Nonlinear Arnoldi required about one third of the average num-

TABLE 4.2
The 2-D discrete Laplacian

Problem(Size)	Eigensolver	$\frac{\|H-\lambda I+g\|}{\|g\|}$	$\frac{\|\|x\|-\Delta\|}{\Delta}$	$\|\lambda^* - d_1\|$	MatVecs
Easy case(324)	IRLM	3.9e-02	4.7e-16	-	140
	NLArn	6.7e-04	9.5e-07	-	38
Easy case(1024)	IRLM	2.3e-06	1.3e-06	-	127
	NLArn	3.7e-04	2.3e-06	-	36
Hard case(324)	IRLM	4.9e-02	5.4e-16	7.8e-02	161
	NLArn	5.4e-05	1.9e-07	4.6e-07	97
Hard case(1024)	IRLM	6.5e-06	7.7e-16	7.2e-03	256
	NLArn	2.3e-02	4.8e-16	2.7e-06	180

ber of matrix-vector products required by the IRLM. In the hard case, savings of roughly 30% in the number of matrix-vector products were obtained with Nonlinear Arnoldi. For NLArn, 2-4 restarts were necessary in the hard case and none in the easy case. Another interesting point is the accuracy of λ^* in the hard case. NLArn determines λ^* much more accurately than the IRLM.

4.3. UDU^T family. In these problems, the matrix H was of the form $H = UDU^T$ with D a diagonal matrix with elements $d_1, \ldots, d_n$, and $U = I - 2uu^T$ with $u^T u = 1$. The elements of D were randomly generated with a uniform distribution on $(-5, 5)$, then sorted in nondecreasing order and d_1 was set to -5. Both vectors u and g were randomly generated with entries selected from a uniform distribution on $(-0.5, 0.5)$. The vector u was normalized to have unit length. We studied problems of order 300 and 1000.

The eigenvectors of H are of the form $q_i = e_i - 2uu_i, i = 1, \ldots, n$, with e_i the ith canonical vector and u_i the ith component of the vector u. The vector g was orthogonalized against $q_1 = e_1 - 2uu_1$, and a noise vector was added to g. Finally, g was normalized to have unit norm. The noise vectors had norms 10^{-2} and 10^{-8} for the easy and hard case, respectively. We computed $x_{min} = \|(H - d_1 I)^\dagger g\|, \Delta_{min} = \|x_{min}\|$, and then set $\Delta = 0.1\Delta_{min}$ for the easy case and $\Delta = 5\Delta_{min}$ for the hard case. One fact makes this problem extremely difficult to solve: typically, x_{min} is almost orthogonal to q_1 but has huge components γ_i in the directions $q_i, i = 2, 3, 4, 5$ and only very small components in eigendirections corresponding to large eigenvalues of H. To construct an appropriate solution in the hard case, the vectors

$$\begin{pmatrix} 0 \\ q_1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 \\ x_{min} \end{pmatrix} \approx \begin{pmatrix} 1 \\ \sum_{i=2}^{5} \gamma_i q_i \end{pmatrix}$$

have to be properly separated. This is a highly demanding task since the vectors q_i correspond to the same cluster around $\delta_1 = -5$.

In all experiments, the parameters `epsilon_Delta = 1e-4`, `epsilon_HC = 1e-10` and the initial guess `delta_U = -4.5` were used. In the easy case, `max_eigentol = 0.2` and $\alpha^{(0)} = \delta_U$. In the hard case, `max_eigentol = 0.03` and $\alpha^{(0)} =$'min'. These choices are the same as in [15].

The initial vector for the first call to the IRLM was $v_0 = (1, \ldots, 1)^T/\sqrt{n+1}$. The total number of vectors was fixed at 10 in the easy case and at 24 in the hard case. For NLArn, $p = 40, \ell = 5, q = 2$ were used in the easy case. In the hard case, the number

of vectors kept in case of a restart was increased to $q = 20$ and the starting search space was improved by setting $\ell = 35$. The maximum dimension of the search space was $p = 40$. Average results for ten related problems, differing only in the vector g, are shown in Table 4.3.

TABLE 4.3
The UDU^T family

Problem(Size)	Eigensolver	$\frac{\|H-\lambda I+g\|}{\|g\|}$	$\frac{\|\|x\|-\Delta\|}{\Delta}$	$\|\lambda^* - d_1\|$	MatVecs
Easy case(300)	IRLM	5.6e-07	2.5e-05	-	103
	NLArn	2.8e-06	2.5e-05	-	39
Easy case(1000)	IRLM	3.0e-06	1.1e-05	-	90
	NLArn	1.8e-06	1.2e-05	-	38
Hard case(300)	IRLM	7.0e-06	8.7e-06	1.8e-03	755
	NLArn	2.2e-08	1.9e-05	1.8e-03	224
Hard case(1000)	IRLM	9.7e-06	2.1e-05	2.1e-04	954
	NLArn	4.7e-07	2.9e-05	2.4e-04	278

We can see in Table 4.3 that the IRLM required about three times the number of matrix-vector products required by NLArn.

5. Conclusions. A suitable algorithm for solving the Large-Scale Trust-Region Subproblem is the LSTRS method. The main computation in LSTRS is the solution of a sequence of eigenvalue problems. Since by construction of the algorithm this sequence is convergent, it is highly advantageous to use the information gathered while solving one eigenproblem in the solution of the next. The Nonlinear Arnoldi method can efficiently use all previously-obtained information. In all our experiments, using the Nonlinear Arnoldi method instead of the implicitly restarted Lanczos method significantly reduced the computational cost.

REFERENCES

[1] C. Fortin and H. Wolkowicz. the trust region subproblem and semidefinite programming. *Optimization Methods and Software*, 19:41 – 67, 2004.

[2] W.W. Hager. Minimizing a quadratic over a sphere. *SIAM J. Optim.*, 12:188 – 208, 2001.

[3] W.W. Hager and S. Park. Global convergence of ssm for minimizing a quadratic over a sphere. *Math. Comp.*, 74:1413 – 1423, 2005.

[4] P.C. Hansen. Regularization tools version 4.0 for Matlab 7.3. *Numer. Alg.*, 46:189 – 194, 2007.

[5] J. Lampe and H. Voss. On a quadratic eigenproblem occuring in regularized total least squares. *Comput. Stat. Data Anal.*, 52/2:1090 – 1102, 2007.

[6] J. Lampe and H. Voss. A fast algorithm for solving regularized total least squares problems. *Electr. Trans. Numer. Anal.*, 31:12 – 24, 2008.

[7] J. Lampe and H. Voss. Global convergence of RTLSQEP: a solver of regularized total least squares problems via quadratic eigenproblems. *Math. Modelling Anal.*, 13:55 – 66, 2008.

[8] J. Lampe and H. Voss. Solving regularized total least squares problems based on eigenproblems. Technical report, Institute of Numerical Simulation, Hamburg University of Technology, 2008. Submitted to Taiwanese Journal of Mathematics.

[9] R.B. Lehoucq, D.C. Sorensen, and C. Yang. *ARPACK Users' Guide. Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia, 1998.

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

[11] R.A. Renaut and H. Guo. Efficient algorithms for solution of regularized total least squares. *SIAM J. Matrix Anal. Appl.*, 26:457 – 476, 2005.

[12] F. Rendl and H. Wolkowicz. A semidefinite framework for trust region subproblems with applications to large scale minimization. *Math. Prog.*, 77:273 – 299, 1997.

[13] M. Rojas. *A large-scale trust-region approach to the regularization of discrete ill-posed problems*. PhD thesis, Rice University, Houston, Texas, 1998.

[14] M. Rojas, S.A. Santos, and D.C. Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM J. Optim.*, 11:611 – 646, 2000.

[15] 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.

[16] 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.*, 23:1843 – 1861, 2002.

[17] D.M. Sima, S. Van Huffel, and G.H. Golub. Regularized total least squares based on quadratic eigenvalue problem solvers. *BIT Numerical Mathematics*, 44:793 – 812, 2004.

[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] D. C. Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM Journal on Matrix Analysis and Applications*, 13(1):357–385, 1992.

[20] D.C. Sorensen. Newton's method with a model trust-region modification. *SIAM J. Numer. Anal.*, 19:409 – 426, 1982.

[21] D.C. Sorensen. Minimization of a large-scale quadratic function subject to a spherical constraint. *SIAM J. Optim.*, 7:141 – 161, 1997.

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