# ITERATIVE METHODS FOR SINGULAR LINEAR EQUATIONS AND LEAST-SQUARES PROBLEMS

A DISSERTATION
SUBMITTED TO THE INSTITUTE FOR
COMPUTATIONAL AND MATHEMATICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Sou-Cheng (Terrya) Choi
December 2006

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Michael A. Saunders)
Principal Advisor

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Gene H. Golub)
Co-Advisor

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Rasmus M. Larsen)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Doron Levy)

Approved for the University Committee on Graduate Studies.

# Abstract

CG, MINRES, and SYMMLQ are Krylov subspace methods for solving large symmetric systems of linear equations. CG (the conjugate-gradient method) is reliable on positive-definite systems, while MINRES and SYMMLQ are designed for indefinite systems. When these methods are applied to an inconsistent system (that is, a singular symmetric least-squares problem), CG could break down and SYMMLQ's solution could explode, while MINRES would give a least-squares solution but not necessarily the minimum-length solution (often called the pseudoinverse solution). This understanding motivates us to design a MINRES-like algorithm to compute minimum-length solutions to singular symmetric systems.

MINRES uses QR factors of the tridiagonal matrix from the Lanczos process (where $R$ is upper-tridiagonal). Our algorithm uses a QLP decomposition (where rotations on the right reduce $R$ to lower-tridiagonal form), and so we call it MINRES-QLP. On singular or nonsingular systems, MINRES-QLP can give more accurate solutions than MINRES or SYMMLQ. We derive preconditioned MINRES-QLP, new stopping rules, and better estimates of the solution and residual norms, the matrix norm and condition number.

For a singular matrix of arbitrary shape, we observe that null vectors can be obtained by solving least-squares problems involving the transpose of the matrix. For sparse rectangular matrices, this suggests an application of the iterative solver LSQR. In the square case, MINRES, MINRES-QLP, or LSQR are applicable. Results are given for solving homogeneous systems, computing the stationary probability vector for Markov Chain models, and finding null vectors for sparse systems arising in helioseismology.

# Acknowledgments

First and foremost, I owe an enormous debt of gratitude to my advisor Professor Michael Saunders for his tireless support throughout my graduate education in Stanford. Michael is the best mentor a research student could possibly hope for. He is of course an amazing academic going by his first-rate scholarly abilities, unparalleled mastery of his specialty, and profound insights on matters algorithmic and numerical (not surprising considering that he is one of most highly cited computer scientists in the world today). But above and beyond all these, Michael is a most wonderful gentleman with great human qualities—he is modest, compassionate, understanding, accommodating, and possesses a witty sense of humor. I am very fortunate, very proud, and very honored to be Michael's student. This thesis certainly would not have been completed without Michael's most meticulous and thorough revision.

Professor Gene Golub is a demigod in our field and a driving force behind the computational mathematics community at Stanford. Incidentally, Gene is also Michael's advisor many years ago. I am also very grateful to Gene for his generosity and encouragement. He is the only professor I know who gives students 24-hour access to his large collection of books in his office. His stature and renown for hospitality attract visiting researchers from all over the world and create a most lively and dynamic environment at Stanford. This contributed greatly to my academic development. Like me, Gene came from a working class family—a rarity in a place like Stanford where many students are of the well-heeled gentry. He has often reminded me that a humble background is no obstacle to success. I am also very fortunate, very proud, and very honored to have Gene as my co-advisor.

Special thanks are due to Professor Chris Paige of McGill University for generously sharing his ideas and insights. He spent many precious hours with me over emails and long discussions during his two visits to Stanford in the past year. Chris is a giant in the field and it is a great honor to fill a gap in one of the famous works of Chris and Michael started long ago, and to have their help in doing so.

I thank my reading committee members: Dr. Rasmus Larsen and Professor Doron Levy. Their helpful suggestions have improved this thesis enormously. My thanks also to Professor Jerome Friedman for chairing my oral defense despite already having retired a few months earlier.

I am very grateful to my professors from the National University of Singapore (NUS), who have instilled and inspired in me interests in computational mathematics since I was an undergraduate: Dr. Lawrence K. H. Ma, Professors Choy-Heng Lai, Jiang-Sheng Wang, Zuowei Shen, Gongyun Zhou, Kim-Chuan Toh, Belal Baaquie, Kan Chen, and last but not least Prabir Burman (UC Davis).

The work in this thesis was generously supported by research grants of Professors Michael Saunders, Gene Golub, and David Donoho. Thanks are also due to the C. Gary & Virginia Skartvedt Endowed Engineering Fund for a Stanford School-of-Engineering Fellowship, and to the Silicon Valley Engineering Council for an SVEC Scholarship.

MATLAB has been an indispensable tool—without which, none of the numerical experiments

could have been performed with such ease and efficiency. I am proud to say that I learned MATLAB first-hand from the person who created it—Professor Cleve Moler. I thank Cleve for selecting me as his teaching assistant for the course on which his very enjoyable book [71] is based (and for kindly recommending me as teaching assistant to his daughter Professor Kathryn Moler, who taught the course in the subsequent year). The book is filled with illuminating examples and this thesis has borrowed a most fascinating one (cf. Chapter 1).

I thank Michael Friedlander for the elegant thesis template that he generously shares with the Stanford public.

I have been fortunate to intern at both Google and IBM Almaden Labs, during which periods I benefited from working with Doctors John Tomlin, Andrew Tomkins, and Tom Truong.

Specifically I want to thank Dr. Xiaoye Sherry Li and Professor Amy Langville for inviting me to speak about applications motivated by this thesis in Lawrence Berkeley Lab and the SIAM Annual Meeting 2004 respectively. Thanks also to Professor Beresford Parlett and Professor Inderjit Dhillon for the opportunities to speak in their seminars in UC Berkeley and UT Austin respectively.

I also want to take the opportunity to thank each administrator and staff member of Stanford and NUS who have gone beyond their call of duty: Professors Walter Murray and Peter Glynn, Indira Choudhury, Lilian Lao, Evelyn Boughton, Lorrie Papadakis, Tim Keely, Seth Tornborg, Suzanne Bigas, Connie Chan, Christine Fiksdal, Dana Halpin, Jam Kiattinant, Nikkie Salgado, Claire Stager, Deborah Michael, Lori Cottle, Pat Shallenberger, Helen Tombropoulos, Sharon Bergman, Lee Kuen Chee, and Kowk Te Ang.

I am indebted to the following friends and colleagues for their friendship and encouragement that made my Stanford years so much more enjoyable: Michael's family Prue, Tania, and Emily; David, Ha, and baby Mike Saunders; Holly Jin, Neil, Danlin, and Hansen Lillemark; Lilian Lao, Michael and Victor Dang; Justin Wing Lok Wan and Winnie Wan Chu; Dulce Ponceleón, Walter, Emma and Sofia Murray; Von Bing Yap and Anne Suet Lin Chong; Pei Yee Woo and Kenneth Wee; Larry and Mary Wong. I thank for their friendship and wisdom: Monica Johnston, Wanchi So, Regina Ip-Lau, Stephen Ng, Wah Tung Lau, Chris Ng, Jonathan Choi, Xiaoqing Zhu, Sorav Bansal, Jeonghee Yi, Mike Ching, Cindy Law, Doris Wong, Jasmine Wong, Sandi Suardi, Sharon Wong, Popoh Low, Grace Ng, Roland Law, Ricky Ip, Fanny Lau, Stephen Yeung, Kenneth (D&G) Wong Chok Hang Yeung, Carrie Teng, Grace Hui, Anthony So, Samuel Ieong, Kenneth Tam, Yee Wai Chong, Anthony Fai Tong Chung, Winnie Wing Yin Choi, Victor Lee, William Yu Cheong Chan, Dik Kin Wong, Collin Kwok-Leung Mui, Rosanna Man, Michael Friedlander, Kaustuv, Zheng Su, Yen Lin Chia, Hanh Huynh, Wanjun Mi, Linzhong Deng, Ofer Levi, James Lambers, Paul Tupper, Melissa Aczon, Paul Lim, Steve Bryson, Oren Livne, Valentin Spitkovsky, Cindy Mason, Morten Mørup, Anil Gaba, Donald van Deventer, Kenji Imai, Chong-Peng Toh, Frederick Willeboordse, Yuan Ping Feng, Alex Ling, Roland Su, Helen Lau, and Suzanne Woo.

I have been infinitely lucky to have met Lek-Heng Lim when we were both undergraduates in NUS. As I made further acquaintance with Lek-Heng, I found him to be the most thoughtful, encouraging, and inspiring friend and colleague I could wish to have. Without his encouragement, I would not have started this long journey, let alone finished.

Last but not least, I thank my parents and grandma for years of toiling and putting up with my "life-long" studies. I am equally indebted to my siblings Dawn and Stephen and brother-in-law Jack Cheng for their love and constant support.

# Contents

# Tables and Figures

## Tables

# Figures

# Chapter 1

# Introduction

## 1.1 The Motivating Problem

In 1998 when the Google PageRank algorithm was first described [16], the World Wide Web contained about 150 million web pages and the classical power method appeared to be effective for computing the relevant matrix eigenvector. By 2003, the number of web pages had grown to 2 billion, and the power method was still being used (monthly) to compute an up-to-date ranking vector. Given some initial eigenvector estimate $v_0$, the power method involves the iteration

$$x_k = Av_{k-1}, \qquad v_k = x_k/\|x_k\|, \qquad k = 1, \ldots, k_P, \tag{1.1}$$

where $A$ is a square matrix with rows and columns corresponding to web pages, and $A_{ij} \neq 0$ if there is a link from page $j$ to page $i$. Each column of $A$ sums to 1 and thus $A$ is called a column-stochastic matrix. Moreover, if its underlying graph is strongly connected, then by the Perron-Frobenius theorem, $A$ would have a simple dominant eigenvalue of 1 and thus the power method is applicable. In practice, the convergence of (1.1) appeared to be remarkably good. The required number of iterations $k_P$ was at most a few hundred.

Much analysis has since been done (e.g., [31, 64]), but at this stage, there was still room for optimistic researchers [18, 42, 46] to believe that Krylov subspace methods might prove useful in place of the power method. Since the related eigenvalue is known to be 1, the method of *inverse iteration* [50, p. 362], [87] could be used. This involves a sequence of linear systems in the following iteration:

$$(A - I)x_k = v_{k-1}, \qquad v_k = x_k/\|x_k\|, \qquad k = 1, \ldots, k_I, \tag{1.2}$$

where the number of iterations $k_I$ would be only 1 or 2. The matrix $A-I$ is intentionally singular, and the computed solutions $x_k$ are expected to grow extremely large ($\|x_k\| \approx 1/\varepsilon$, where $\varepsilon$ is the machine precision), so that the normalized vectors $v_k$ would satisfy $(A - I)v_k \approx 0$ and hence $Av_k \approx v_k$ as required.

Of course, Krylov subspace methods involve many matrix-vector products $Av$ (as in the power method) and additional storage in the form of some very large work vectors.

### 1.1.1 Null Vectors

The Google matrix $A$ is square but unsymmetric. With the PageRank computation in mind, we were motivated to investigate the use of LSQR [82, 83] on singular least-squares systems

$$\min_x \|Ax - b\|_2, \qquad A \in \mathbb{R}^{m\times n}, \qquad \text{rank}(A) < n \tag{1.3}$$

in order to compute *null vectors* $v$ satisfying $Av \approx 0$. (We have now replaced $A - I$ by $A$, and $A$ may be rectangular.) For almost any nonzero vector $b$, the computed solution $x$ should be extremely large in norm, and the normalized vector $v = x/\|x\|$ will be a null vector of $A$.

Our first test matrix $A$ was derived from $A_H$, the $500 \times 500$ unsymmetric Harvard matrix called `harvard500` assembled by Cleve Moler [71] to simulate the PageRank problem. With normal stopping tolerances in place, we found that LSQR converged to a least-squares solution that did *not* have large norm (and was not a null vector of $A$). Only after disabling all stopping conditions were we able to force LSQR to continue iterating until the solution norm finally increased toward $1/\varepsilon$, giving a null vector $v = x/\|x\|$ as required.

### 1.1.2 A Revelation

The question arose: Which solution $x$ was LSQR converging to with the normal stopping rules when $A$ was singular? Probably it was the *minimum-length solution* in which $\|x\|_2$ is minimized among the (infinitely many) solutions that minimize $\|Ax - b\|_2$. In any case, the associated residual vector $r = b - Ax$ was satisfying $A^T r = 0$ because LSQR's stopping rules require $\|A^T r\|/(\|A\|\|r\|)$ to be small when $\|r\| \neq 0$. Suddenly we realized that we were computing a null vector for the *transpose* matrix $A^T$. This implied that to obtain a null vector for the singular matrix $A$ in (1.3), we could solve the least-squares problem

$$\min_y \|A^T y - c\|_2, \qquad A \in \mathbb{R}^{m\times n}, \qquad \operatorname{rank}(A) < n \tag{1.4}$$

with some rather arbitrary vector $c$. The optimal residual $s = c - A^T y$ would satisfy $As = 0$, and the required null vector would be $v = s/\|s\|$. Furthermore, LSQR should converge sooner on (1.4) than if we force it to compute a very large $x$ for (1.3).

Figure 1.1 shows LSQR converging twice as quickly on (1.4) compared to (1.3).

### 1.1.3 Symmetric Systems

At some point another question arose: What would happen in the symmetric case? Both the systems (1.3) and (1.4) take the form

$$\min_x \|Ax - b\|_2, \qquad A \in \mathbb{R}^{n\times n} \text{ and symmetric}, \qquad \operatorname{rank}(A) < n. \tag{1.5}$$

For general symmetric $A$ (not necessarily positive definite), the natural Krylov subspace methods are SYMMLQ and MINRES [81]. When $A$ is singular, MINRES is the logical choice because it allows the residual $r = b - Ax$ to be nonzero. In all cases, the optimal residual satisfies $Ar = 0$. If $b$ happens to lie in the range of $A$, the optimal residual is $r = 0$, but otherwise—for example, if $b$ is a random vector—we can expect $r \neq 0$, so that $v = r/\|r\|$ will be a null vector, and again it will be obtained sooner than if we force iterations to continue until $\|x\|$ is extremely large. In a least-squares problem, $\|r\| > 0$ and thus MINRES would need new stopping conditions to detect if $\|Ax\|/\|x\|$ or $\|Ar\|/\|r\|$ were small enough. We derive recurrence relations for $\|Ax\|$ and $\|Ar\|$ that give us accurate estimates without extra matrix-vector multiplications.

We created our second test matrix from `harvard500` by defining $\widetilde{A} = A_H + A_H^T$ and constructing a diagonal matrix $D$ with diagonal elements $d(i) = 1/\sqrt{\|\widetilde{A}(i,:)\|_1}$, which is well-defined

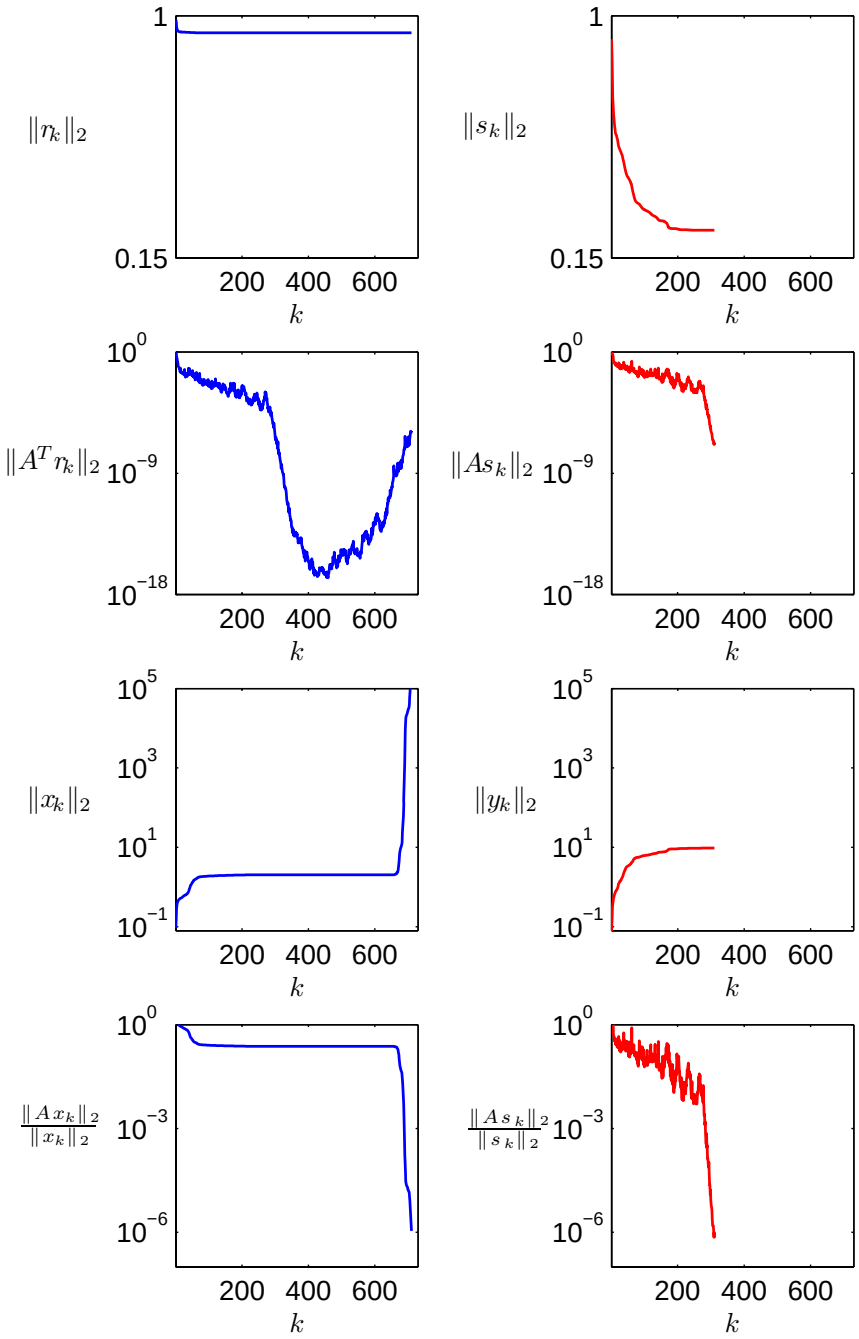


FIGURE 1.1 *Solving* $\min \|Ax-b\|$ *(1.3) and* $\min \|A^Ty-c\|$ *(1.4) with* $A = A_H - I$, *b random,* $\|b\|_2 = 1$, *and* $c = b$, *where* $A_H$ *is the* $500\times500$ *Harvard matrix of Moler [71]. The matrix is unsymmetric with rank* 499. *Both solves compute the null vector of A.* ***Left:*** *With the normal stopping rules disabled, LSQR on* $\min \|Ax-b\|$ *(1.3) takes 711 iterations to give an exploding solution* $x_k$ *such that* $\|Ax_k\|/\|x_k\| \approx 1\times10^{-6}$, *where k is the LSQR iteration number.* ***Right:*** *In contrast, LSQR on* $\min \|A^Ty-c\|$ *(1.4) takes only 311 iterations to give* $s_k = c - A^Ty_k$ *such that* $\|As_k\|/\|s_k\| \approx 7\times10^{-7}$. *To reproduce this figure, run* `testNull3([3,4])`.

because there is no zero row in $A_H$, and then we apply diagonal scaling: $\hat{A} = D\widetilde{A}D$. Note that $\hat{A}$ is not a doubly stochastic matrix (which would have a trivial dominant eigenvector $e = [1, \ldots, 1]^T$), but it happens to have a simple dominant eigenvalue 1. We applied MINRES twice on (1.5) with the shifted matrix $A := \hat{A} - I$ and a randomly generated $b$: the first time with normal stopping conditions and a second time with all stopping conditions disabled except $\|Ax\|/\|x\| < \text{tol}$. The results are shown in Figure 1.2.

Given that singular least-squares problems have an infinite number of solutions, the same question arises: Which solution does MINRES produce on singular problems? As for LSQR, we surmised that it would be the minimum-length solution, and indeed this is true for MINRES when $b$ lies in the range of $A$. However, when the optimal $r = b - Ax$ in (1.5) is nonzero, we found experimentally (and later theoretically) that MINRES does *not* return the minimum-length solution.

Thus began the research that comprises most of this thesis. A new implementation called MINRES-QLP has been developed that has the desired property on singular systems (that of minimizing $\|x\|$). The implementation is substantially more complex, but as a bonus we expect MINRES-QLP to be more accurate than the original MINRES on nonsingular symmetric systems $Ax = b$.

For a preview of the performance of MINRES-QLP compared to MINRES with normal stopping conditions on symmetric problems, see Figures 1.3–1.6. On ill-conditioned nonsingular compatible systems, the solution quality of MINRES-QLP could be similar to that of MINRES, but the residuals are much more accurate (see Figures 1.5 and 1.6). There are applications, such as null-vector computations, that require accurate residuals. On singular systems, MINRES-QLP's solutions and residuals could be much more accurate than MINRES's (see Figures 1.3 and 1.4).

Ipsen and Meyer [60] state that in general, Krylov subspace methods such as GMRES on singular compatible systems yield only the Drazin inverse solution (see section 2.3.2 for more details). GMRES is actually mathematically equivalent to MINRES if $A$ is symmetric. In contrast, our work shows that both MINRES and MINRES-QLP could give us the minimum-length solution.

Ipsen and Meyer [60] also show that in general, Krylov subspace methods return no solution for inconsistent problems. However, we show that MINRES computes a least-squares solution (with minimum $\|r\|_2$) and our new Krylov subspace method MINRES-QLP gives the minimum-length solution to singular symmetric linear systems or least-squares problems.

In Chapter 2 we establish that for singular and incompatible Hermitian problems, existing iterative methods such as the conjugate-gradient method CG [57], SYMMLQ [81], MINRES [81], and SQMR [38] cannot minimize the solution norm and residual norm *simultaneously*.

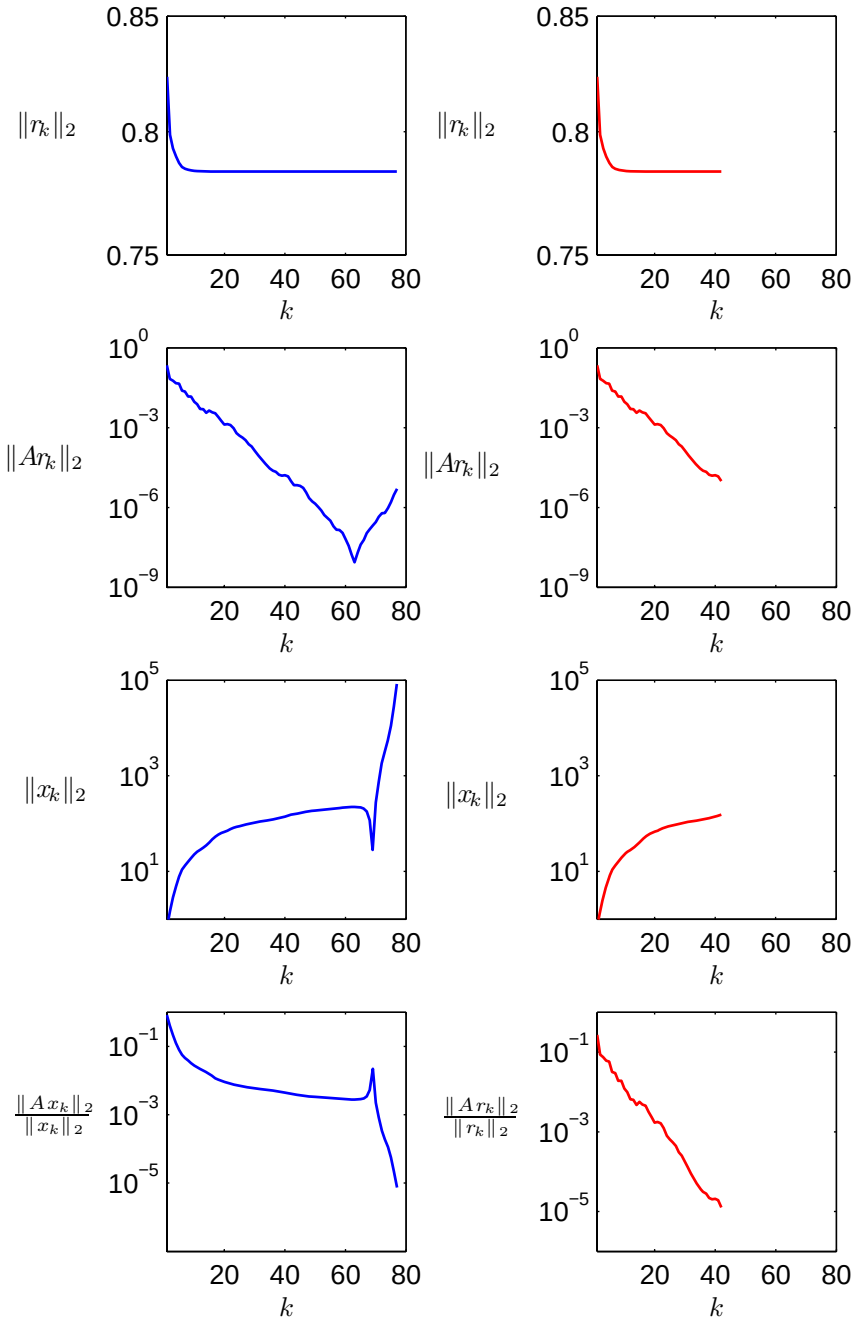


FIGURE 1.2 *Solving* $\min \|Ax - b\|$ *(1.5) with* $A := \hat{A} - I$*, where* $\hat{A}$ *is a symmetrized and scaled form of the* $500 \times 500$ *Harvard matrix* $A_H$ *(see text in section 1.1.3),* $b$ *random, and* $\|b\|_2 = 1$*. The matrix has rank 499.* ***Left:*** *With all stopping rules disabled except* $\|Ax_k\|/\|x_k\| < \text{tol} = 10^{-5}$*, MINRES takes 77 iterations to give an exploding solution* $x_k$ *such that* $\|x_k\| \approx 8 \times 10^4$ *and* $\|Ax_k\|/\|x_k\| \approx 7 \times 10^{-6}$*.*
***Right:*** *In contrast, MINRES with normal stopping conditions takes only about 42 iterations to give* $r_k$ *such that* $\|r_k\| \approx 0.78$ *and* $\|Ar_k\|/\|r_k\| \approx 1 \times 10^{-5}$*. Since the system is incompatible, MINRES needs new stopping conditions to detect if* $\|Ax_k\|/\|x_k\|$ *or* $\|Ar_k\|$ *is small enough. To reproduce this figure, run* `testNull3([5,6])`*.*

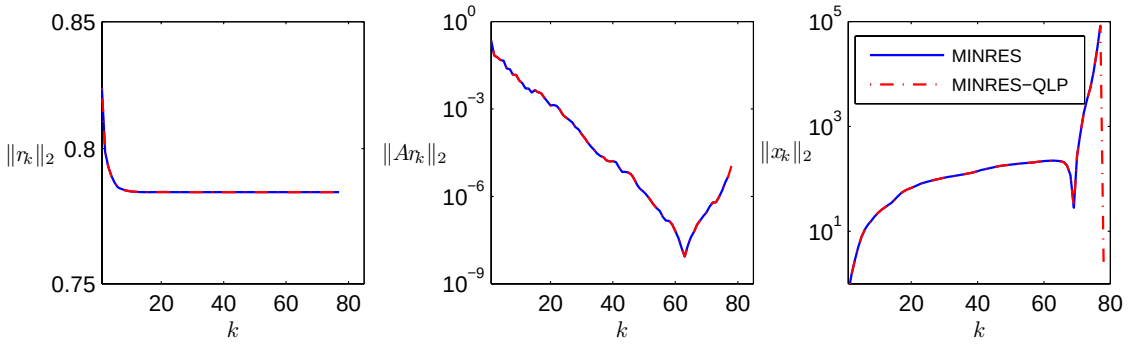


FIGURE 1.3 *Solving* $\min \|Ax - b\|$ *(1.5) with* $A := \hat{A} - I$, *the symmetrized/scaled/shifted* $500 \times 500$ *Harvard matrix,* $\|A\|_2 \approx 2$, $b$ *random, and* $\|b\|_2 = 1$. *The matrix has rank* 499 *and the system* $Ax = b$ *is incompatible. MINRES takes* 77 *iterations to give an exploding solution* $x_k$ *such that* $\|x_k\| \approx 2 \times 10^5$, *while MINRES-QLP takes 78 iterations to give* $\|x_k\| \approx 2.6$, *with* $\|r_k\| \approx 0.78$ *and* $\|Ar_k\| \approx 10^{-5}$ *in both cases. We also computed the truncated eigenvalue-decomposition solution (TEVD solution) and found that it matches our MINRES-QLP solution here. To reproduce this figure, run* `PreviewMINRESQLP1(1)`.

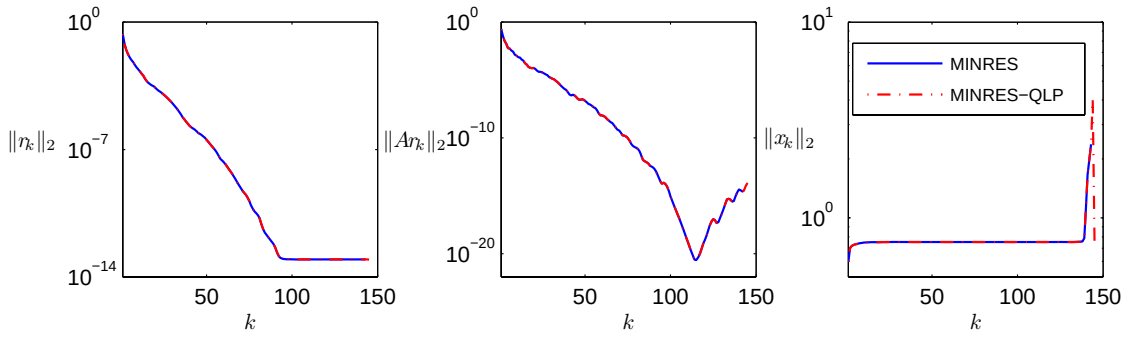


FIGURE 1.4 *Solving* $\min \|Ax - b\|$ *(1.5) with symmetric* $A$ *as in Figure 1.2 and Figure 1.3. We define* $b = Az_1 + z_2$, *where* $z_1$ *and* $z_2$ *are randomly generated with* $\|z\|_1 \approx 13$ *and* $\|z\|_2 \approx 10^{-12}$, *and then we normalize* $b$ *by its* 2*-norm. Thus* $b$ *has very small component in the null space of* $A$*—if any at all. The matrix has rank 499 but the system* $Ax = b$ *is nearly compatible. The plots of MINRES and MINRES-QLP overlap completely except for the last two iterations. MINRES takes 143 iterations to give a nonminimum-length solution* $x_k$ *such that* $\|x_k\| \approx 4.0$, *while MINRES-QLP takes 145 iterations to give* $\|x_k\| \approx 0.75$, *with* $\|r_k\| \approx 10^{-13}$ *and* $\|Ar_k\| \approx 10^{-14}$ *in both cases. We also computed the TEVD solution and found that it matches our MINRES-QLP solution here. If we had not known that this were generated as an almost compatible system, we would have guessed that it is compatible. MINRES-QLP appears to have a better regularization property than MINRES. This example also prompts us to ask the question: how to put a dividing line—in terms of* $\|r_k\|$*—between a linear system and a least-squares problem? To reproduce this figure, run* `PreviewMINRESQLP1(4)`.

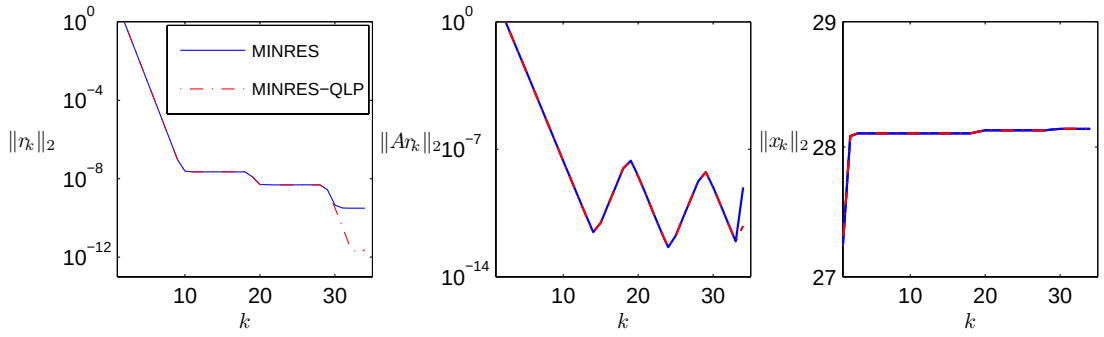


FIGURE 1.5 *Solving $Ax = b$ with symmetric positive definite $A = Q\,\mathrm{diag}([10^{-8}, 2 \times 10^{-8}, 2 : \frac{1}{789} : 3])Q$ of dimension $n = 792$ and norm $\|A\|_2 = 3$, where $Q = I - (2/n)ee^T$ is a Householder matrix generated by $e = [1, \ldots, 1]^T$. We define $b = Ae$ ($\|b\| \approx 70.7$). Thus, the true solution is $x = e$ and $\|x\| = O(\|b\|)$. This example is constructed similar to Figure 4 in Sleijpen et al. [96]. The left and middle plots differ after 30 and 33 iterations, with the final MINRES solution $x_k^M$ giving $\|r_k^M\| \approx 10^{-10}$ and $\|Ar_k^M\| \approx 10^{-10}$, while the final MINRES-QLP solution $x_k^Q$ gives $\|r_k^Q\| \approx 10^{-12}$ and $\|Ar_k^Q\| \approx 10^{-12}$. The right plot shows that $\|x_k\|$ is very similar for both methods; in fact for the final points, $\|x_k^M\| \approx \|x_k^Q\| \approx 2.8$ and $\|x_k^M - x\| \approx \|x_k^Q - x\| \approx 2 \times 10^{-7}$. To reproduce this figure, run* `PreviewMINRESQLP2(2)`.

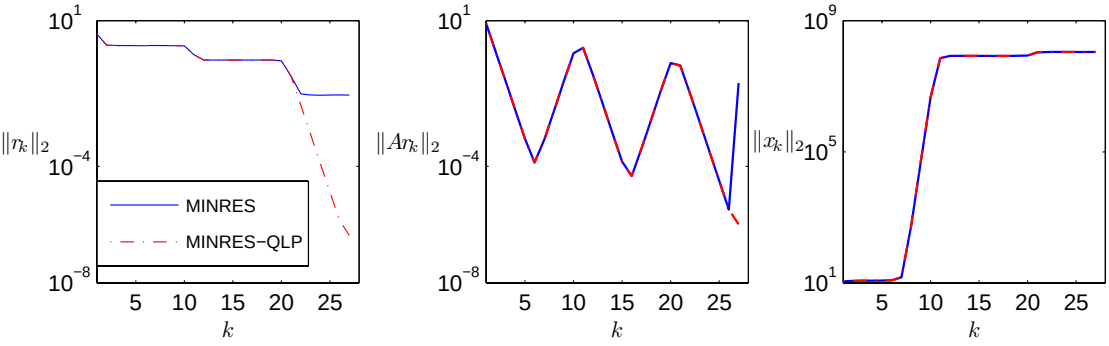


FIGURE 1.6 *Solving $Ax = b$ with the same symmetric positive definite $A$ as in Figure 1.5 but with $b = e$. Since $\mathrm{cond}_2(A) \approx 10^8$ and $\|b\|_2 = \sqrt{n}$, we expect the solution norm to be big ($\|x\| \gg \|b\|$). The left and middle plots differ after 22 and 26 iterations, with the final MINRES solution $x_k^M$ giving $\|r_k^M\| \approx 10^{-2}$ and $\|Ar_k^M\| \approx 10^{-2}$ only, while the final MINRES-QLP solution $x_k^Q$ gives $\|r_k^Q\| \approx 10^{-7}$ and $\|Ar_k^Q\| \approx 10^{-6}$. The right plot shows that $\|x_k\|$ is very similar for both methods; in fact for the final points, $\|x_k^M\| \approx \|x_k^Q\| \approx 10^8$ but $\|x_k^M - x_k^Q\| \approx 1.4$. To reproduce this figure, run* `PreviewMINRESQLP2(1)`.

## 1.2 Preliminaries

### 1.2.1 Problem Description and Formal Solutions

We consider solving for the $n$-vector $x$ in the system of linear equations

$$Ax = b \tag{1.6}$$

when the $n \times n$ real symmetric matrix $A$ is large and sparse, or represents an operator for forming products $Av$. When the real vector $b$ is in the range of $A$, we say that the system is *consistent* or *compatible*; otherwise it is *inconsistent* or *incompatible*.

When $A$ is nonsingular, the system is always consistent and the solution of (1.6) is unique.

When $A$ is singular and (1.6) has at least one solution, we say that the singular system is consistent or compatible, in which case it has infinitely many solutions. To obtain a unique solution, we select the minimum-length solution among all solutions $x$ in $\mathbb{R}^n$ such that $Ax = b$. On the other hand, if the singular system has no solution, we say that it is inconsistent or incompatible, in which case we solve the singular symmetric least-squares problem instead and select the minimum-length solution:

$$x = \arg\min \|Ax - b\|_2. \tag{1.7}$$

More precisely, the minimum-length least-squares problem is defined as

$$\min \|x\|_2 \quad \text{s.t.} \quad x \in \arg\min \|Ax - b\|_2, \tag{1.8}$$

or with the more commonly seen but actually a slight abuse of notation

$$\min \|x\|_2 \quad \text{s.t.} \quad x = \arg\min \|Ax - b\|_2. \tag{1.9}$$

The minimum-length solution of either (1.6) or (1.7) is unique and is also called the *pseudoinverse solution*. Formally,

$$x^\dagger = (A^T A)^\dagger A^T b = (A^2)^\dagger A b,$$

where $A^\dagger$ denotes the pseudoinverse of $A$. We postpone the definition and more discussion of pseudoinverse to section 2.3.1.

We may also consider (1.6) or (1.7) with $A$'s diagonal shifted by a scalar $\sigma$. Shifted problems appear, for example, in inverse iteration (as mentioned in section 1.1) or Rayleigh quotient iteration. The shift is mentioned here because it is best handled within the Lanczos process (see section 2.1) rather than by defining $\hat{A} = A - \sigma I$.

Two related but more difficult problems are known as *Basis Pursuit* and *Basis Pursuit De-Noising* [22, 23] (see also the Lasso problem [103]):

$$\begin{aligned} \min_x \|x\|_1 \quad &\text{s.t.} \quad Ax = b, \\ \min_{x,r} \lambda\|x\|_1 + \tfrac{1}{2}\|r\|_2^2 \quad &\text{s.t.} \quad Ax + r = b, \end{aligned}$$

where $A$ is usually rectangular (with more columns than rows) in signal-processing applications.

TABLE 1.1
*Existing iterative algorithms since CG was created in 1952. All methods require products $Av_k$ for a sequence of vectors $\{v_k\}$. The last column indicates whether a method also requires $A^T u_k$ for a sequence of vectors $\{u_k\}$.*

| **Linear Equations** | **Authors** | **Properties of A** | $\mathbf{A}^T$? |
|---|---|---|---|
| CG | Hestenes and Stiefel (1952) [57] | Symmetric positive definite | |
| CRAIG | Faddeev and Faddeeva (1963)[33] | Square or rectangular | yes |
| MINRES | Paige and Saunders (1975) [81] | Symmetric indefinite | |
| SYMMLQ | Paige and Saunders (1975) [81] | Symmetric indefinite | |
| Bi-CG | Fletcher (1976)[35] | Square unsymmetric | yes |
| LSQR | Paige and Saunders (1982) [82, 83] | Square or rectangular | yes |
| GMRES | Saad and Schultz (1986) [89] | Square unsymmetric | |
| CGS | Sonneveld (1989) [98] | Square unsymmetric | |
| QMR | Freund and Nachtigal (1991) [37] | Square unsymmetric | yes |
| Bi-CGSTAB | Van der Vorst (1992) [109] | Square unsymmetric | yes |
| TFQMR | Freund (1993) [36] | Square unsymmetric | |
| SQMR | Freund and Nachtigal (1994) [38] | Symmetric | |
| **Least Squares** | **Authors** | **Properties of A** | $\mathbf{A}^T$? |
| CGLS | Hestenes and Stiefel (1952) [57] | Square or rectangular | yes |
| RRLS | Chen (1975) [24] | Square or rectangular | yes |
| RRLSQR | Paige and Saunders (1982) [82] | Square or rectangular | yes |
| LSQR | Paige and Saunders (1982) [82] | Square or rectangular | yes |

### 1.2.2 Existing Numerical Algorithms

In this thesis, we are interested in sparse matrices that are so large that direct factorization methods such as Gaussian elimination or Cholesky decomposition are not immediately applicable. Instead, iterative methods and in particular Krylov subspace methods are usually the methods of choice. For example, CG is designed for a symmetric positive definite matrix $A$ (whose eigenvalues are all positive), while SYMMLQ and MINRES are for an indefinite and symmetric matrix $A$ (whose eigenvalues could be positive, negative, or zero).

The main existing iterative methods for symmetric and unsymmetric $A$ are listed in Table 1.1.

### 1.2.3 Background for MINRES

MINRES, first proposed in [81, section 6], is an algorithm for solving indefinite symmetric linear systems. A number of acceleration methods for MINRES using (block) preconditioners have been proposed in [73, 51, 105]. Researchers in various science and engineering disciplines have found MINRES useful in a range of applications, including:

- interior eigenvalue problems [72, 114]
- augmented systems [34]
- nonlinear eigenvalue problems [20]

- characterization of null spaces [21]
- symmetric generalized eigenvalue problems [74]
- singular value computations [112]
- semidefinite programming [111]
- generalized least-squares problems [115].

### 1.2.4 Notation

We keep the lower-case letters $i, j, k$ as subscripts to denote integer indices, $c$ and $s$ to denote cosine and sine of some angle $\theta$, $n$ for order of matrices and length of vectors, and other lower-case letters such as $b$, $u, v$, $w$ and $x$ (possibly with integer subscripts) to denote *column* vectors of length $n$. In particular, $e_k$ denotes the $k$th unit vector. We use upper-case italic letters (possibly with integer subscripts) to denote matrices. The exception is superscript $T$, which denotes the transpose of a vector or matrix. We reserve $I_k$ to denote identity matrix of order $k$, and $Q_k$ and $P_k$ for orthogonal matrices. Lower-case Greek letters denote scalars. The symbol $\|\cdot\|$ denotes the 2-norm of a vector or the Frobenius norm of a matrix. We use $\kappa(A)$ to denote the condition number of matrix $A$; $\mathcal{R}(A)$ and $\mathcal{N}(A)$ to denote the range and null space of $A$; $\mathcal{K}_k(A, b)$ to denote the $k$th Krylov subspace of $A$ and $b$; and $A^\dagger$ is the pseudoinverse of $A$. We use $A \succ 0$ to denote that $A$ is positive definite, $A \not\succ 0$ to mean that $A$ is not positive definite (so $A$ could be negative definite, non-negative definite, indefinite, and so on). When we have a compatible linear system, we often write $Ax = b$. If the linear system is incompatible, we write $Ax \approx b$ as shorthand for the corresponding linear least-squares problem $\min \|Ax - b\|_2$. We use symbols $\|$ to denote parallel vectors, and $\perp$ to denote orthogonality.

Most of the results in our discussion are directly extendable to problems with complex matrices and vectors. When special care is needed in handling complex problems, we will be very specific. We use superscript $H$ to denote the conjugate transpose of a complex matrix or vector.

### 1.2.5 Computations

We use MATLAB 7.0 and double precision for computations unless otherwise specified. We use $\varepsilon$ (varepsilon) to denote machine precision ($= 2^{-52} \approx 2.2 \times 10^{-16}$). In an algorithm, we use // to indicate comments. For measuring mathematical quantities or complexity of algorithms, sometimes we use big-oh $O(\cdot)$ to denote an asymptotic upper bound [94, Definition 7.2]:

> $f(n) = O(g(n))$ if $\exists\ c > 0$ and a positive integer $n_0 \in \mathbb{N}$ such that $\forall n \in \mathbb{N}$, $n \geq n_0$, $f(n) \leq cg(n)$.

Thus a nonzero constant $\alpha = O(1) = O(n)$ and $\alpha n = O(n)$. We note that $f(n) = O(g(n))$ is a slight abuse of notation—to be precise, it is $f(n) \in O(g(n))$.

Following the philosophy of reproducible computational research as advocated in [27, 25], for each figure and example we mention either the source or the specific MATLAB command.

### 1.2.6 Roadmap

We review the iterative algorithms CG, SYMMLQ, and MINRES for Hermitian linear systems and least-squares problems in Chapter 2, and show that MINRES gives a nonminimum-length solution for inconsistent systems. We also review other Krylov subspace methods such as LSQR and GMRES for non-Hermitian problems, and we derive new recursive formulas for efficient estimation of $\|Ar_k\|$, $\|Ax_k\|$, and the condition number of $A$ for MINRES.

In Chapter 3, we present a new algorithm MINRES-QLP for symmetric and possibly singular systems. Chapter 4 gives numerical examples that contrast the solutions of MINRES with the minimum-length solutions of MINRES-QLP on symmetric and Hermitian systems.

In Chapter 5, we return to the null-vector problem for sparse matrices or linear operators, and apply the previously mentioned iterative solvers.

Chapter 6 summarizes our contributions and ongoing work.

# Chapter 2

# Existing Iterative Methods for Hermitian Problems

In this chapter, we review the Lanczos process and the three best known algorithms for Hermitian linear systems: CG, SYMMLQ, and MINRES. In particular, we emphasize the recurrence relations of various mathematical objects. We assume throughout that $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n, A \neq 0$, and $b \neq 0$. However, the algorithms are readily extended to Hermitian $A$ and complex $b$.

## 2.1 The Lanczos Process

The Lanczos process transforms a symmetric matrix $A$ to a symmetric tridiagonal matrix with an additional row at the bottom:

$$\underline{T_k} = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \alpha_3 & \ddots & \\ & & \ddots & \ddots & \beta_k \\ & & & \beta_k & \alpha_k \\ & & & & \beta_{k+1} \end{bmatrix}.$$

If we define $T_k$ to be the first $k$ rows of $\underline{T_k}$, then $T_k$ is square and symmetric, and

$$\underline{T_k} = \begin{bmatrix} T_k \\ \beta_{k+1} e_k^T \end{bmatrix}, \qquad T_k = \begin{bmatrix} T_{k-1} & \beta_k e_{k-1} \\ \beta_k e_{k-1}^T & \alpha_k \end{bmatrix}.$$

The Lanczos process iteratively computes vectors $v_k$ as follows:

$$v_0 = 0, \qquad \beta_1 v_1 = b, \text{ where } \beta_1 \text{ serves to normalize } v_1, \tag{2.1}$$

$$p_k = Av_k, \qquad \alpha_k = v_k^T p_k,$$

$$\beta_{k+1} v_{k+1} = p_k - \alpha_k v_k - \beta_k v_{k-1}, \text{ where } \beta_{k+1} \text{ serves to normalize } v_{k+1}. \tag{2.2}$$

In matrix form,

$$AV_k = V_{k+1}\underline{T_k}, \text{ where } V_k = \begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix}. \tag{2.3}$$

In exact arithmetic, the columns of $V_k$ are orthonormal and the process stops when $\beta_{k+1} = 0$ ($k \leq n$), and then we obtain

$$AV_k = V_k T_k. \tag{2.4}$$

TABLE 2.1
*Algorithm* ***LanczosStep****.*

| **LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \to \alpha_k, \beta_{k+1}, v_{k+1}$ |
|---|
| $p_k = Av_k - \sigma v_k, \qquad \alpha_k = v_k^T p_k, \qquad p_k \leftarrow p_k - \alpha_k v_k$ |
| $v_{k+1} = p_k - \beta_k v_{k-1}, \qquad \beta_{k+1} = \|v_{k+1}\|_2$ |
| **if** $\beta_{k+1} \neq 0, \qquad v_{k+1} \leftarrow v_{k+1}/\beta_{k+1} \qquad$ **end** |

TABLE 2.2
*Algorithm* ***Tridiag****.*

| **Tridiag**$(A, b, \sigma, \text{maxit}) \to T_k, V_k$ `//partial tridiagonalization of` $A - \sigma I$ |
|---|
| $\beta_1 = \|b\|_2, \qquad v_0 = 0, \qquad \beta_1 = \|b\|, \qquad k = 1$ |
| **if** $\beta_1 \neq 0, \qquad v_1 = b/\beta_1 \qquad$ **end** |
| **while** $\beta_k \neq 0$ **and** $k \leq \text{maxit}$ |
| **LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \to \alpha_k, \beta_{k+1}, v_{k+1}$ |
| $k \leftarrow k + 1$ |
| **end** |

The above discussion can be extended for $A - \sigma I$, where $\sigma$ is a scalar shift. We call each iteration in the Lanczos process a *Lanczos step*: **LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \to \alpha_k, \beta_{k+1}, v_{k+1}$. See Table 2.1 and Table 2.2.

We need to keep at most the matrix $A$ or a function that returns $Ax$ if $A$ is a linear operator, 3 vectors, and 3 scalars in memory. In fact, a careful implementation would only require 2 vectors in working memory at a time, if $v_{k+1}$ replaces $v_{k-1}$. Each iteration performs a matrix-vector multiplication, 2 inner products, 3 scalar-vector multiplications, and 2 vector subtractions, which sums up to $2\nu + 9n$ floating-point operations per iteration, where $\nu$ is number of nonzeros in $A$.

The Lanczos process stops in at most $\min\{\text{rank}(A) + 1, n\}$ iterations. It stops sooner when $A$ has clusters of eigenvalues or $b$ has nonzero components along only a few eigenvectors of $A$.

**Definition 2.1 ($k$th Krylov subspace with respect to $A$ and $b$).** *Given a square $n \times n$ matrix $A \in \mathbb{R}^{n\times n}$ and an $n$-vector $b \in \mathcal{R}(A)$, we define the $k$th Krylov subspace of $(A, b)$ as*

$$\mathcal{K}_k(A, b) := \text{span}\{b, Ab, \ldots, A^{k-1}b\} = \text{span}\{v_1, \ldots, v_k\}, \tag{2.5}$$

*where $k$ is a positive integer.*

**Proposition 2.2.** *Given symmetric $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$ and supposing that $\beta_i > 0$ for $i = 1, \ldots, k$ but $\beta_{k+1} = 0$ in the Lanczos process, we have the following results.*

1. *If $b \in \mathcal{N}(A)$, then $\alpha_1 = 0$, $\beta_2 v_2 = 0$ and $\text{rank}(A) \geq 1$.*
2. *If $b \in \mathcal{R}(A)$, then $v_1 \parallel b$ and $v_2, \ldots, v_k \perp b$ are $k$ orthogonal vectors that lie in $\mathcal{R}(A)$ and $n \geq \text{rank}(A) \geq k$.*
3. *If $b \notin \mathcal{R}(A)$ and $b \notin \mathcal{N}(A)$ (that is, $\mathcal{N}(A)$ is nontrivial; $b$ has a nonzero component in $\mathcal{R}(A)$ and a nonzero component in $\mathcal{N}(A)$), then $v_1, \ldots, v_k$ have nonzero components in $\mathcal{R}(A)$ and thus $n > \text{rank}(A) \geq k - 1$.*

*Proof.* 3. Let $b = b_{\mathcal{R}} + b_{\mathcal{N}}$, where $b_{\mathcal{R}}$ is the component of $b$ in $\mathcal{R}(A)$ and $b_{\mathcal{N}}$ is the component of $b$ in $\mathcal{N}(A)$. The first Lanczos step gives $\beta_1 v_1 = \beta_1(v_{1,\mathcal{R}} + v_{1,\mathcal{N}}) = b_{\mathcal{R}} + b_{\mathcal{N}} = b$. So

$$\alpha_1 = v_1^T A v_1 = (v_{1,\mathcal{R}} + v_{1,\mathcal{N}})^T A(v_{1,\mathcal{R}} + v_{1,\mathcal{N}}) = v_{1,\mathcal{R}}^T A v_{1,\mathcal{R}}, \tag{2.6}$$

$$\begin{aligned}\beta_2 v_2 &= A v_1 - \alpha_1 v_1 = A(v_{1,\mathcal{R}} + v_{1,\mathcal{N}}) - \alpha_1(v_{1,\mathcal{R}} + v_{1,\mathcal{N}}) \\ &= \underbrace{A v_{1,\mathcal{R}} - \alpha_1 v_{1,\mathcal{R}}}_{\beta_2 v_{2,\mathcal{R}}} + \underbrace{- \alpha_1 v_{1,\mathcal{N}}}_{\beta_2 v_{2,\mathcal{N}}}.\end{aligned} \tag{2.7}$$

It follows that $v_{2,\mathcal{N}} \parallel v_{1,\mathcal{N}} \parallel b_{\mathcal{N}}$. Moreover, $\beta_2 v_2 = 0 \iff \beta_2 = 0$ and $v_{2,\mathcal{R}} = v_{2,\mathcal{N}} = 0$. The Lanczos process stops if any $\beta_i = 0$. In general, for $i = 2, \ldots, k$,

$$\begin{aligned}\alpha_i &= v_i^T A v_i = v_{i,\mathcal{R}}^T A v_{i,\mathcal{R}}, \\ \beta_{i+1} v_{i+1} &= A v_i - \alpha_i v_i - \beta_i v_{i-1} \\ &= A(v_{i,\mathcal{R}} + v_{i,\mathcal{N}}) - \alpha_i(v_{i,\mathcal{R}} + v_{i,\mathcal{N}}) - \beta_i(v_{i-1,\mathcal{R}} + v_{i-1,\mathcal{N}}) \\ &= \underbrace{A v_{i,\mathcal{R}} - \alpha_i v_{i,\mathcal{R}} - \beta_i v_{i-1,\mathcal{R}}}_{\beta_{i+1} v_{i+1,\mathcal{R}}} + \underbrace{- \alpha_i v_{i,\mathcal{N}} - \beta_i v_{i-1,\mathcal{N}}}_{\beta_{i+1} v_{i+1,\mathcal{N}}},\end{aligned}$$

so that $v_{i+1,\mathcal{N}} \parallel v_{1,\mathcal{N}} \parallel b_{\mathcal{N}}$. Thus

$$\begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix} = \begin{bmatrix} v_{1,\mathcal{R}} & \cdots & v_{k,\mathcal{R}} \end{bmatrix} + v_{1,\mathcal{N}} c^T,$$

where $c^T = \begin{bmatrix} 1 & c_2 & \cdots & c_k \end{bmatrix}$ for some scalars $c_i$. Thus,

$$\operatorname{rank}\left(\begin{bmatrix} v_{1,\mathcal{R}} & \cdots & v_{k,\mathcal{R}} \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix} - v_{1,\mathcal{N}} c^T\right) = k - 1 \text{ or } k$$

since a rank-1 change to a full rank matrix of rank $k$ can only change the matrix rank by at most 1, and an $n \times k$ with $k \leq n$ matrix could have at most rank $k$. Thus

$$\operatorname{rank}(A) \geq \operatorname{rank}\left(\begin{bmatrix} v_{1,\mathcal{R}} & \cdots & v_{k,\mathcal{R}} \end{bmatrix}\right) = k - 1 \text{ or } k.$$

■

**Corollary 2.3.** *Given symmetric $A \in \mathbb{R}^{n \times n}$, we define $r = \operatorname{rank}(A)$.*

1. *If $b \in \mathcal{R}(A)$, then $\beta_{k+1} = 0$ for some $k \leq r \leq n$.*
2. *If $r < n$ and $b \notin \mathcal{R}(A)$, then $\beta_{k+1} = 0$ for some $k \leq r + 1 \leq n$.*

**Theorem 2.4.** *Given a symmetric matrix $A \in \mathbb{R}^{n \times n}$ with $s$ distinct nonzero eigenvalues and $b \in \mathbb{R}^n$ that has nonzero components along $t$ ($t \leq s$) eigenvectors of $A$ that correspond to $t$ distinct nonzero eigenvalues of $A$, then $\beta_{k+1} = 0$ for some $k \leq \min\{t + 1, s\}$ if $b \notin \mathcal{R}(A)$, or $k \leq t$ if $b \in \mathcal{R}(A)$.*

**Example 1.**

1. *Let $A = \operatorname{diag}([\,1\ 2\ 3\ 4\ 5\,])$, $n = 5$, $r = s = 5$.*
   (a) *If $b = [\,1\ 2\ 3\ 4\ 5\,]^T$, then $t = 5$, $\beta_6 = 0$.*

(b) *If* $b = [1\ 2\ 0\ 0\ 0]^T$, *then* $t = 2$, $\beta_3 = 0$.

2. *Let* $A = \text{diag}([1\ 2\ 3\ 0\ 0])$, $n = 5$, $r = s = 3$.

   (a) *If* $b = [1\ 2\ 0\ 0\ 0]^T$, *then* $b \in \mathcal{R}(A)$, $t = 2$, $\beta_3 = 0$.
   (b) *If* $b = [1\ 0\ 0\ 0\ 0]^T$, *then* $b \in \mathcal{R}(A)$, $t = 1$, $\beta_2 = 0$.
   (c) *If* $b = [1\ 2\ 3\ 4\ 0]^T$, *then* $b \notin \mathcal{R}(A)$, $t = 3$, $\beta_4 = 0$.
   (d) *If* $b = [1\ 0\ 0\ 4\ 0]^T$, *then* $b \notin \mathcal{R}(A)$, $t = 1$, $\beta_3 = 0$.

3. *Let* $A = \text{diag}([2\ 2\ 3\ 0\ 0]), r = 3, s = 2$.

   (a) *If* $b = [1\ 2\ 0\ 0\ 0]^T$, *then* $b \in \mathcal{R}(A)$, $t = 1$, $\beta_3 = 0$.
   (b) *If* $b = [1\ 2\ 3\ 4\ 0]^T$, *then* $b \notin \mathcal{R}(A)$, $t = 2$, $\beta_4 = 0$.
   (c) *If* $b = [1\ 0\ 0\ 4\ 0]^T$, *then* $b \notin \mathcal{R}(A)$, $t = 1$, $\beta_3 = 0$.

## 2.2 Lanczos-Based Methods for Linear Systems

In each Lanczos step, we solve a subproblem to find $x_k \in \mathcal{K}_k(A, b)$ such that $x_k = V_k y$ for some $y \in \mathbb{R}^k$. It follows that $r_k = b - Ax_k = V_{k+1}(\beta_1 e_1 - \underline{T_k} y)$, and all Lanczos-based methods attempt to make $\beta_1 e_1 - \underline{T_k} y$ small in one way or another. CG focuses on the first $k$ equations, attempting to solve for $T_k y = \beta_1 e_1$ by applying the Cholesky decomposition to $T_k$. SYMMLQ concentrates on the first $k-1$ equations and wants to solve the underdetermined system $\underline{T_{k-1}}^T y = \beta_1 e_1$. That said, since $\underline{T_k}$ is available in the $k$th iteration, SYMMLQ goes ahead and solves $\underline{T_k}^T y = \beta_1 e_1$ instead by applying the LQ decomposition to $\underline{T_k}^T$. MINRES works to minimize the 2-norm of $\beta_1 e_1 - \underline{T_k} y$ by applying the QR decomposition to $\underline{T_k}$. The following stencil depicts the rationale and focuses of the three methods, where $s$'s represent the last row of the tridiagonal matrix in SYMMLQ's $(k-1)$th iteration, $c$'s in CG's $k$th iteration, $m$'s in MINRES's $k$th iteration, and $*$ for common entries of all three methods:

$$\begin{bmatrix} * & * & & & \\ * & * & * & & \\ s & s & s & & \\ & & c & c & \\ & & & m & \end{bmatrix} y \approx \begin{bmatrix} \beta_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The three methods are best juxtaposed in the framework described by Paige [79], as summarized in Saunders [90]:

> *An iterative process generates certain quantities from the data. At each iteration a subproblem is defined, suggesting how those quantities may be combined to give a new estimate of the required solution. Different subproblems define different methods for solving the original problem. Different ways of solving a subproblem lead to different implementations of the associated method.*

Tables 2.3–2.4 (from [90]) give the subproblem associated with each method, and the mechanism for defining solution estimates for the original problem in terms of various transformed bases. CG and LanczosCG are two implementations of the same method.

TABLE 2.3
*Subproblem definitions of CG, SYMMLQ, and MINRES.*

| Method | Subproblem | Factorization | Estimate of $x_k$ |
|---|---|---|---|
| LanczosCG or CG [57] | $T_k y_k = \beta_1 e_1$ | Cholesky: $T_k = L_k D_k L_k^T$ | $x_k = V_k y_k \in \mathcal{K}_k(A,b)$ |
| SYMMLQ [81, 90] | $y_{k+1} = \arg\min_{y\in\mathbb{R}^{k+1}} \{\|y\| \mid \underline{T_k}^T y = \beta_1 e_1\}$ | LQ: $\underline{T_k}^T Q_k = \begin{bmatrix} L_k & 0 \end{bmatrix}$ | $x_k = V_{k+1} y_{k+1} \in \mathcal{K}_{k+1}(A,b)$ |
| MINRES [81] | $y_k = \arg\min_{y\in\mathbb{R}^k} \left\|\underline{T_k} y - \beta_1 e_1\right\|$ | QR: $Q_k \underline{T_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k \in \mathcal{K}_k(A,b)$ |

TABLE 2.4
*Bases and subproblem solutions in CG, SYMMLQ, and MINRES.*

| Method | New basis | $z_k$ | Estimate of $x_k$ |
|---|---|---|---|
| LanczosCG | $W_k := V_k L_k^{-T}$ | $L_k D_k z_k = \beta_1 e_1$ | $x_k = W_k z_k$ |
| CG | $W_k := V_k L_k^{-T} \Phi_k$ <br> $\Phi_k := \text{diag}(\|r_1\|, \ldots, \|r_k\|)$ | $L_k D_k \Phi_k z_k = \beta_1 e_1$ | $x_k = W_k z_k$ |
| SYMMLQ | $W_k := V_{k+1} Q_k \begin{bmatrix} I_k \\ 0 \end{bmatrix}$ | $L_k z_k = \beta_1 e_1$ | $x_k = W_k z_k$ |
| MINRES | $D_k := V_k R_k^{-1}$ | $R_k z_k = \beta_1 \begin{bmatrix} I_k & 0 \end{bmatrix} Q_k e_1$ | $x_k = D_k z_k$ |

Another way to classify Krylov subspace methods is based on the error and residual properties as described in Demmel [29, section 6.6.2]:

1. Minimum-residual method:
   find $x_k \in \mathcal{K}_k(A,b)$ such that $\|r_k\|$ is minimized.

2. Orthogonal-residual/Galerkin method:
   find $x_k \in \mathcal{K}_k(A,b)$ such that $r_k \perp \mathcal{K}_k(A,b)$; that is, $V_k^T r_k = 0$.

3. Minimum-error method:
   find $x_k = \text{argmin}_{\bar{x}_k \in \mathcal{K}_k(A,b)} \|x - \bar{x}_k\|$, where $x$ denotes the true solution.

Table 2.5 gives an expanded description.

TABLE 2.5
*Residual and error properties of CG, SYMMLQ, and MINRES.*

| | $k$th residual | $k$th error |
|---|---|---|
| CG for $A \succeq 0$ | $\min \|r_k\|_{A^{-1}}, \quad r_k \perp \mathcal{K}_k(A,b), \quad Ar_k \perp \mathcal{K}_{k-1}(A,b)$ | $\min \|x - x_k\|_A$ |
| SYMMLQ | $r_k \perp \mathcal{K}_k(A,b), \quad Ar_k \perp \mathcal{K}_{k-1}(A,b)$ | $\min \|x - x_k\|_2$ |
| MINRES | $\min \|r_k\|_2, \quad \beta_{k+1} = 0 \Rightarrow r_k \perp \mathcal{K}_k(A,b), \quad Ar_k \perp \mathcal{K}_k(A,b)$ | — |

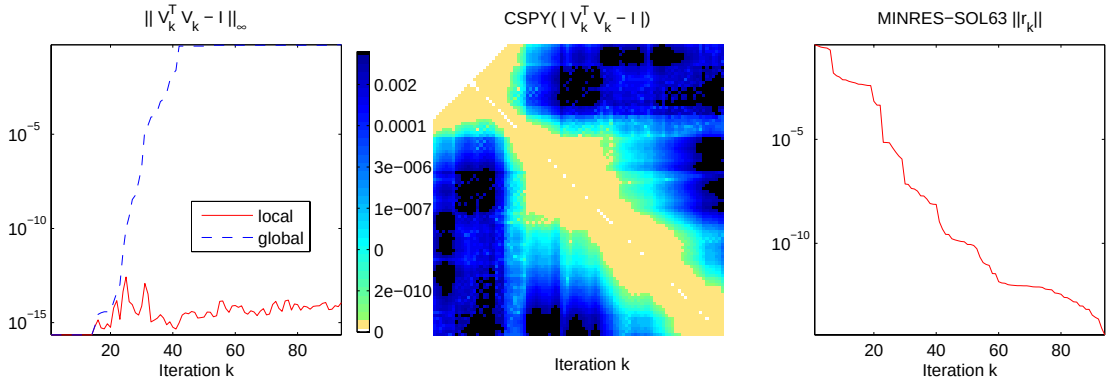


FIGURE 2.1 *$A$ is symmetric tridiagonal of order* 100 *and full rank, and $b$ is a scalar multiple of $e_1$. The Lanczos vectors are the sparsest possible: $v_k = e_k$. **Left**: In double precision, loss of local orthogonality among $v_{k-2}, v_{k-1}, v_k$ for each iteration $k = 1, \ldots, 94$, and loss of global orthogonality among $v_1, \ldots, v_k$. **Middle**: Color-spying elementwise-absolute values of $V_k^T V_k - I$. The color patterns are symmetric. The upper left corner is usually closest to zero (of order $\varepsilon$) and white in color. The area closer to the diagonal indicates the extent of loss of local orthogonality. In contrast, the areas in the upper right and lower left corners correspond to the loss of global orthogonality, which is larger in magnitude and darker in color. **Right**: Loss of global orthogonality in the Lanczos basis, however, implies convergence of solution in the Lanczos-based solver MINRES. This figure can be reproduced by* `LossOrthogonality(1)`.

In finite-precision arithmetic, the columns of $V_k$ are observed to lose orthogonality when the $x_k$'s from one of the Lanczos-based methods are converging to the solution [78, 84]. See Figure 2.1.

### 2.2.1 CG

In this section, we present two equivalent CG algorithms. One is derived from the Lanczos process for academic interest (Table 2.6), and the other is the standard CG algorithm (Table 2.7), which is more memory efficient and commonly found in the literature (e.g., [50]).

The $k$th iteration of CG works on the Cholesky factors of $T_k$ from the Lanczos process:

$$T_k = L_k D_k L_k^T, \qquad L_k = \begin{bmatrix} 1 & & & \\ \iota_2 & 1 & & \\ & \ddots & \ddots & \\ & & \iota_k & 1 \end{bmatrix}, \qquad D_k = \operatorname{diag}(\delta_1, \ldots, \delta_k).$$

In the rest of this section, we highlight a few important properties of CG. We first assume $A \succ 0$ and then relax it to $A \succeq 0$ later.

**Proposition 2.5 ($\|Ar_k\|$ for CG).**

1. $\|Ar_0\| = \|r_0\| \sqrt{\frac{1+\mu_2}{\nu_1^2}}$.
2. $\|Ar_k\| = \|r_k\| \sqrt{\frac{\mu_{k+1}}{\nu_k^2}\left(1 + \mu_{k+1} + \frac{2\nu_k}{\nu_{k+1}}\right) + \frac{1+\mu_{k+2}}{\nu_{k+1}^2}}$ *for* $k = 1, \ldots$ *when* $q_k^T A q_k \neq 0$.
3. $\|Ar_k\| = \|r_k\| \sqrt{\frac{\mu_{k+1}}{\nu_k^2}(1 + \mu_{k+1})}$ *when* $q_k^T A q_k = 0$.

TABLE 2.6
*Algorithm **LanczosCG**. We assume $A$ is symmetric only.*

**LanczosCG**$(A, b, \sigma, \text{maxit}) \to x, \phi$
$\beta_1 = \|b\|_2, \quad v_0 = 0, \quad \beta_1 v_1 = b, \quad x_0 = 0, \quad \phi_0 = \beta_1, \quad k = 1$
**while** no stopping condition is true,
  **LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \to \alpha_k, \beta_{k+1}, v_{k+1}$
  `//Cholesky factorization`
  **if** $k = 1$
    $\iota_1 = 1, \quad \delta_1 = \alpha_1$
  **else**
    $\iota_k = \frac{\beta_k}{\delta_{k-1}}, \quad \delta_k = \alpha_k - \delta_{k-1}\iota_k^2$
  **end**
  `//update solution and residual norm`
  **if** $\delta_k \le 0$, STOP **end** `//`$A$ `indefinite, perhaps unstable to continue`
  **if** $k = 1$
    $\zeta_1 = \frac{\beta_1}{\delta_1}, \quad w_1 = v_1, \quad x_1 = \zeta_1 w_1$
  **else**
    $\zeta_k = -\frac{\delta_{k-1}\iota_k\zeta_{k-1}}{\delta_k}, \quad w_k = v_k - \iota_k w_{k-1}, \quad x_k = x_{k-1} + \zeta_k w_k$
  **end**
  $\phi_k = |\zeta_k|\,\beta_{k+1}, \quad k \leftarrow k + 1$
**end**
$x = x_k, \quad \phi = \phi_k$

TABLE 2.7
*Algorithm **CG**. We assume $A = A^T \succeq 0$. A careful implementation would need to keep the matrix $A$ (or a function that returns $Ax$ if $A$ is a linear operator) and 2 to 4 vectors in working memory. The algorithm also estimates $\phi = \|r_k\|$, $\chi = \|x_k\|$, $\mathcal{A} \approx \|A\|_2$, and $\kappa \approx \kappa(A)$.*

**CG**$(A, b, \text{tol}, \text{maxit}) \to x, \phi, \chi, \mathcal{A}, \kappa$ `//if` $x = 0$`, no converged solution.`
$x_0 = 0, \quad r_0 = b, \quad \beta_1 = \|b\|, \quad \chi_0 = 0, \quad \phi_0^2 = \|r_0\|^2, \quad q_1 = r_0$
$k = 1, \quad \kappa = 1, \quad \mathcal{A} = 0, \quad \nu_{\min} = 0$
**while** $\left(\frac{\phi_k}{\mathcal{A}\chi_k + \beta_1} > \text{tol}\right)$ **or** $(k < \text{maxit})$
  $s_k = Aq_k, \quad \xi_k = q_k^T s_k$
  **if** $\xi_k \le 0$
    $x_k := 0, \quad \phi_k = \beta_1, \quad \chi_k = 0,$ STOP `//`$q_k$ `is a null vector`
  **end**
  $\nu_k = \phi_{k-1}^2/\xi_k, \quad x_k = x_{k-1} + \nu_k q_k, \quad r_k = r_{k-1} - \nu_k s_k, \quad \chi_k = \|x_k\|$
  $\phi_k^2 = \|r_k\|^2, \quad \mu_{k+1} = \phi_k^2/\phi_{k-1}^2, \quad q_{k+1} = r_k + \mu_{k+1} q_k$ `//gradient`
  $\nu_{\min} = \min\{\nu_{\min}, \nu_k\}, \quad \mathcal{A} = \max\{\mathcal{A}, \nu_k\}, \quad \kappa = \frac{\mathcal{A}}{\nu_{\min}}, \quad k = k + 1$
**end**
$x = x_k, \quad \phi = \phi_k, \quad \chi = \chi_k$

The following lemma implies that CG is only applicable to symmetric linear systems.

**Lemma 2.6.** *$\|r_k\| = 0$ if and only if $\|Ar_k\| = 0$.*

**Proposition 2.7 (Null vector of $A \succeq 0$ from CG's breakdown).** *In exact arithmetic, if $A \succeq 0$ and $\xi_k = q_k^T A q_k = 0$, then $\nu_k$ becomes undefined and CG breaks down, and the gradient $q_k$ is a null vector of $A$.*

**Proposition 2.8 (Null vector of $A \succeq 0$ from CG's exploding solution).** *In finite-precision arithmetic, if $A \succeq 0$ and $\xi_k = q_k^T A q_k = O(\varepsilon)$ in CG, then $\nu_k$ and $x_k$ explode, and $x_k$ (normalized) is an approximate null vector of $A$.*

When we know in advance that $A$ is symmetric negative semidefinite, we can apply CG to $(-A)x = -b$ to get a solution since $A \preceq 0$ if and only if $-A \succeq 0$.

Most textbook discussions restrict application of CG to a symmetric positive definite matrix $A$ because $\|\cdot\|_A$ and $\|\cdot\|_{A^{-1}}$ are in general not defined for singular $A$. However, CG can often be applied to a symmetric positive semidefinite matrix $A$ (all eigenvalues of $A$ nonnegative) without failure if $b \in \mathcal{R}(A)$. Moreover, CG sometimes also works with a symmetric indefinite (singular) matrix if we change the stopping condition from $(\xi_k \leq 0)$ to $(\xi_k = 0)$. For example,

$$A = \begin{bmatrix} 1 & & & \\ & 2 & & \\ & & -1 & \\ & & & 0 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \end{bmatrix}.$$

We label this variation of CG as CGI (see Table 2.8). CGI will not work when $q_k$ is a null vector of $A$ or a solution of $x^T A x = 0$. With CGI, Proposition 2.7 and Proposition 2.8 become the following.

**Proposition 2.9 (Solution of $x^T A x = 0$ from CGI's breakdown).** *In exact arithmetic, if $\xi_k = q_k^T A q_k = 0$, then $\nu_k$ becomes undefined and CGI breaks down, and the gradient $q_k$ is a solution of the quadratic equation $x^T A x = 0$.*

**Proposition 2.10 (Solution of $x^T A x = 0$ from CGI's exploding solution).** *In finite-precision arithmetic, if $\xi_k = q_k^T A q_k = O(\varepsilon)$ in CGI, then $\nu_k$ and $x_k$ explode, and $x_k$ (normalized) is an approximate solution of the quadratic equation $x^T A x = 0$.*

**Example 2.** *A case when CG and CGI fail.*

$$A = \begin{bmatrix} -20 & & & \\ & -19 & & \\ & & \ddots & \\ & & & 20 \end{bmatrix}, \qquad b = Ae = \begin{bmatrix} -20 \\ -19 \\ \vdots \\ 20 \end{bmatrix},$$

*$Ab \neq 0$, but $q_1^T A q_1 = b^T A b = 0$, rendering failure of CGI. However, SYMMLQ and MINRES work to give the solution $\begin{bmatrix} 1_{10} & 0 & 1_{10} \end{bmatrix}$.*

TABLE 2.8
*Algorithm **CGI**. We assume $A = A^T$ only.*

**CGI**$(A, b, \text{tol}, \text{maxit}) \to x, \phi, \chi, \mathcal{A}, \kappa$ `//if` $x = 0$`, no converged solution.`
$x_0 = 0, \quad r_0 = b, \quad \beta_1 = \|b\|, \quad \chi_0 = 0, \quad \phi_0^2 = \|r_0\|^2, \quad q_1 = r_0,$
$k = 1, \quad \kappa = 1, \quad \mathcal{A} = 0$
**while** $\left(\frac{\phi_k}{\mathcal{A}\chi_k + \beta_1} > \text{tol}\right)$ **or** $(k < \text{maxit})$
  $s_k = Aq_k, \quad \xi_k = q_k^T s_k$
  **if** $\xi_k = 0 \quad x_k := 0, \quad \phi_k = \beta_1, \quad \chi_k = 0, \quad$ STOP **end**
  $\nu_k = \phi_{k-1}^2/\xi_k, \quad x_k = x_{k-1} + \nu_k q_k, \quad r_k = r_{k-1} - \nu_k s_k, \quad \chi_k = \|x_k\|$
  $\phi_k^2 = \|r_k\|^2, \quad \mu_{k+1} = \phi_k^2/\phi_{k-1}^2, \quad q_{k+1} = r_k + \mu_{k+1} q_k$ `//gradient`
  $\nu_{\min} = \min\{\nu_{\min}, |v_k|\}, \quad \mathcal{A} = \max\{\mathcal{A}, |v_k|\}, \quad \kappa = \frac{\mathcal{A}}{\nu_{\min}}, \quad k = k + 1$
**end**
$x = x_k, \quad \phi = \phi_k, \quad \chi = \chi_k$

### 2.2.2 SYMMLQ

When $A$ is not symmetric positive definite, CG is no longer applicable. SYMMLQ was first published in [81, section 5] for solving $Ax = b$ with $A$ being symmetric indefinite. Later, the associated subproblem was found to be the following [90]:

$$y_{k+1} = \arg\min \left\{ \|y\| \mid \underline{T_k}^T y = \beta_1 e_1, \quad y \in \mathbb{R}^{k+1} \right\}, \tag{2.8}$$

where $\underline{T_k}^T$ is available at the $k$th Lanczos Step. The subproblem is best solved using the LQ decomposition

$$\underline{T_k^T} P_k = \begin{bmatrix} & L_{k-1} & & \\ \epsilon_k^{(1)} & \delta_k^{(2)} & \gamma_k^{(1)} & \beta_{k+1} \end{bmatrix} P_{k,k+1} = \begin{bmatrix} \gamma_1^{(1)} & & & & & \\ \delta_2^{(2)} & \gamma_2^{(2)} & & & & \\ \epsilon_3^{(1)} & \ddots & \ddots & & & \\ & \ddots & \ddots & \ddots & & \\ & & \epsilon_k^{(1)} & \delta_k^{(2)} & \gamma_k^{(2)} & 0 \end{bmatrix} := \begin{bmatrix} L_k & 0 \end{bmatrix}, \tag{2.9}$$

where $P_k = P_{1,2} P_{2,3} \cdots P_{k,k+1}$ is a product of suitable orthogonal matrices. The implementation uses Householder reflectors of dimension 2 [107, Exercise 10.4]—very similar to Givens rotations. For each $k$, $P_{k,k+1}$ is orthogonal and symmetric, and is constructed to annihilate $\beta_{k+1}$, the bottom-right element of $\underline{T_k^T}$. A compact way to describe the action of $P_{k,k+1}$ is

$$\begin{bmatrix} \gamma_k^{(1)} & \beta_{k+1} \end{bmatrix} \begin{bmatrix} c_k & s_k \\ s_k & -c_k \end{bmatrix} = \begin{bmatrix} \gamma_k^{(2)} & 0 \end{bmatrix}, \qquad \rho_k = \sqrt{\gamma_k^{(1)} + \beta_{k+1}^2}, \quad c_k := \frac{\gamma_k^{(1)}}{\rho_k}, \quad s_k := \frac{\beta_{k+1}}{\rho_k}.$$

However, that definition of $c_k$ and $s_k$ should not be directly implemented. A more stable implementation of the orthogonal transformation is given in Table 2.9. The complexity is at most 6 flops and a square root.

TABLE 2.9
*Algorithm* ***SymOrtho.***

**SymOrtho**$(a, b) \to c, s, r$
**if** $b = 0$
  $s = 0, \quad r = |a|, \quad$ **if** $a = 0, \quad c = 1 \quad$ **else** $\quad c = \text{sign}(a) \quad$ **end**
**elseif** $a = 0$
  $c = 0, \quad s = \text{sign}(b), \quad r = |b|$
**elseif** $|b| > |a|$
  $\tau = a/b, \quad s = \text{sign}(b)/\sqrt{1+\tau^2}, \quad c = s\tau, \quad r = b/s$
**elseif** $|a| > |b|$
  $\tau = b/a, \quad c = \text{sign}(a)/\sqrt{1+\tau^2}, \quad s = c\tau, \quad r = a/c$
**end**

Each $P_{i,i+1}$ is defined in terms of previous $c_i$ and $s_i$:

$$P_{i,i+1} := \begin{bmatrix} I_{i-1} & & & \\ & c_i & s_i & \\ & s_i & -c_i & \\ & & & I_{k-i} \end{bmatrix}. \tag{2.10}$$

If we define $y_{k+1} = P_k \bar{z}_{k+1}$, then our subproblem (2.8) is solved by

$$L_k z = \beta_1 e_1, \qquad \bar{z}_{k+1} = \begin{bmatrix} z_k \\ 0 \end{bmatrix}, \tag{2.11}$$

and SYMMLQ computes $x_k$ in $\mathcal{K}_{k+1}(A, b)$ as an approximate solution to our problem $Ax = b$:

$$x_k = V_{k+1} y_{k+1} = V_{k+1} P_k \begin{bmatrix} z_k \\ 0 \end{bmatrix} = W_k z_k = x_{k-1} + \zeta_k w_k, \tag{2.12}$$

where $V_{k+1} P_k = \begin{bmatrix} W_k & \bar{w}_{k+1} \end{bmatrix}$, $\zeta_k$ is the last component of $z_k$, and

$$\zeta_k = \frac{-\epsilon_k^{(1)} \zeta_{k-2} - \delta_k^{(2)} \zeta_{k-1}}{\gamma_k^{(2)}}, \qquad w_k = c_k \overline{w}_k + s_k v_{k+1}, \qquad \overline{w}_{k+1} = s_k \overline{w}_k - c_k v_{k+1}. \tag{2.13}$$

We list the algorithm SYMMLQ in Table 2.10 and give a list of properties including recurrence relations for $r_k$, $Ar_k$, and their 2-norms. Note that $r_k$ is not usually explicitly computed and its norm can be obtained only in iteration $k+1$. We do not compute $Ar_k$ or its norm because (as we will see) SYMMLQ is designed for compatible linear systems but not least-squares problems. Most of the following SYMMLQ properties are presented and succinctly proved in the later part of [81, section 5].

**Proposition 2.11 ($r_k$ of SYMMLQ).**

1. $r_0 = \beta_1 v_1 = b$ *and* $\|r_0\| = \beta_1$.

TABLE 2.10

*Algorithm **SYMMLQ** with possible transfer to the CG point at the end. This algorithm also estimates solution and residual norms $\chi = \|x_k\|$, $\phi = \|r_k\|$. At the end of the algorithm, if the recurrently computed residual norm of CG point $\phi_k^C$ is smaller than that from SYMMLQ, the algorithm will compute the CG iterate $x_k^C$ from the SYMMLQ iterate $x_k$.*

**SYMMLQ**$(A, b, \sigma, \text{maxit}) \rightarrow x, \phi, \chi$

$\beta_1 = \|b\|_2, \qquad v_0 = 0, \qquad \beta_1 v_1 = b, \qquad \overline{w}_1 = v_1, \qquad x_0 = 0$

$\phi_0 = \beta_1, \qquad \chi_0 = 0, \qquad \zeta_{-1} = \zeta_0 = 0, \qquad k = 1$

**while** no stopping condition is true

**LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \rightarrow \alpha_k, \beta_{k+1}, v_{k+1}$

`//last right orthogonalization on middle two entries in last row of` $\underline{T_k^T}$

$\delta_k^{(2)} = c_{k-1}\delta_k^{(1)} + s_{k-1}\alpha_k, \qquad \gamma_k^{(1)} = s_{k-1}\delta_k^{(1)} - c_{k-1}\alpha_k$

`//last right orthogonalization to produce first two entries of` $\underline{T_{k+1}} e_{k+2}$

$\epsilon_{k+1}^{(1)} = s_{k-1}\beta_{k+1}, \qquad \delta_{k+1}^{(1)} = -c_{k-1}\beta_{k+1}$

`//current right orthogonalization to zero out` $\beta_{k+1}$

**SymOrtho**$(\gamma_k^{(1)}, \beta_{k+1}) \rightarrow c_k, s_k, \gamma_k^{(2)}$

`//update solution, solution norm, residual norm, CG residual norm`

**if** $\gamma_k^{(2)} = 0$

STOP `//`$\beta_{k+1} = 0$ `and` $x = x_{k-1}$`; or` $b \notin \mathcal{R}(A)$ `and no solution`

**else**

**if** $k = 1, \qquad \zeta_1 = \frac{\beta_1}{\gamma_1^{(1)}}$ **else** $\zeta_k = \frac{-\epsilon_k^{(1)}\zeta_{k-2} - \delta_k^{(2)}\zeta_{k-1}}{\gamma_k^{(2)}}$ **end**

$\chi_k = \sqrt{\chi_{k-1}^2 + \zeta_k^2}, \qquad \phi_{k-1} = \left\| \begin{bmatrix} -\gamma_k^{(2)}\zeta_k & \epsilon_{k+1}^{(1)}\zeta_{k-1} \end{bmatrix} \right\|, \qquad \phi_k^C = \frac{s_k c_{k-1}}{c_k}\phi_{k-1}^C$

**if** $\phi_{k-1}$ is small

$x_k = x_{k-1}, \qquad \phi_k = \phi_{k-1}.$ STOP `//`$x_{k-1}$ `is solution`

**else**

$w_k = c_k\overline{w}_k + s_k v_{k+1}, \qquad \overline{w}_{k+1} = s_k\overline{w}_k - c_k v_{k+1}, \qquad x_k = x_{k-1} + \zeta_k w_k$

**end**

**end**

$k \leftarrow k + 1$

**end**

**if** $\phi_k^C < \phi_k$ **and** $c_k \neq 0,$ `//transfer to CG point`

$x_k \leftarrow x_k + \left(\frac{\zeta_k s_k}{c_k}\right)\overline{w}_{k+1}, \qquad \chi_k \leftarrow \sqrt{\chi_k^2 + \left(\frac{\zeta_k s_k}{c_k}\right)^2}, \qquad \phi_k = \phi_k^C$

**end**

$x = x_k, \qquad \chi = \chi_k, \qquad \phi = \phi_k$

2. *For $k \geq 1$, define $\omega_{k+1} = \gamma_{k+1}^{(2)}\zeta_{k+1}$ and $\varrho_{k+2} = \epsilon_{k+2}^{(1)}\zeta_k$. Then*

$$r_k = \omega_{k+1}v_{k+1} - \varrho_{k+2}v_{k+2}, \tag{2.14}$$

$$\phi_k := \|r_k\| = \left\| \begin{bmatrix} \omega_{k+1} & \varrho_{k+2} \end{bmatrix} \right\| = \begin{bmatrix} \varrho_{k+1} + \delta_{k+1}^{(2)}\zeta_k & \varrho_{k+2} \end{bmatrix}, \text{ where } \varphi_k = \epsilon_{k+1}^{(1)}\zeta_{k-1}. \tag{2.15}$$

*Thus, $V_k^T r_k = 0$.*

**Proposition 2.12 ($Ar_k$ of SYMMLQ).**

1. *$Ar_0 = \beta_1(\alpha_1 v_1 + \beta_2 v_2)$ and $\|Ar_0\| = \beta_1\sqrt{\alpha_1^2 + \beta_2^2}$.*
2. *Define $\omega_{k+1} = \gamma_{k+1}^{(2)}\zeta_{k+1}$ and $\varrho_{k+2} = \epsilon_{k+2}^{(1)}\zeta_k$. Then*

$$\begin{aligned} Ar_k = {} & \beta_{k+1}\omega_{k+1}v_k - (\alpha_{k+1}\omega_{k+1} - \beta_{k+2}\varrho_{k+2})v_{k+1} \\ & - (\beta_{k+2}\omega_{k+1} - \alpha_{k+2}\varrho_{k+2})v_{k+2} - \beta_{k+3}\varrho_{k+2}v_{k+3}, \end{aligned} \tag{2.16}$$

$$\|Ar_k\| = \left\| \begin{bmatrix} \beta_{k+1}\omega_{k+1} \\ \alpha_{k+1}\omega_{k+1} - \beta_{k+2}\varrho_{k+2} \\ \beta_{k+2}\omega_{k+1} - \alpha_{k+2}\varrho_{k+2} \\ -\beta_{k+3}\varrho_{k+2} \end{bmatrix} \right\| \text{ for } k = 1, \ldots. \tag{2.17}$$

The following results say that it is impossible to have $\|Ar_k\| = 0$ while $\|r_k\| \neq 0$, which is a property of a symmetric least-squares solution. Thus SYMMLQ is not applicable to incompatible symmetric linear system of equations.

**Lemma 2.13.** $\phi_k = 0 \Leftrightarrow \psi_k = 0$.

**Lemma 2.14 (Solution norm of SYMMLQ and its monotonicity).** *Let $\chi_0 = 0$. Then $\chi_k = \|x_k\|_2 = \|z_k\| = \sqrt{\chi_{k-1}^2 + \zeta_k^2}$ is monotonically increasing as $k$ increases.*

**Proposition 2.15 (SYMMLQ's breakdown on incompatible systems).** *Suppose we want to solve $Ax = b$ where $A = A^T$ and $b$ are given. In exact arithmetic, if $\gamma_k^{(2)} = 0$, then SYMMLQ breaks down. If $\delta_k^{(2)} = \epsilon_k^{(1)} = 0$, then $x_{k-1}$ is our solution; otherwise, $b \notin \mathcal{R}(A)$ and there is no solution from SYMMLQ.*

In finite precision, we may be able to obtain an exploding solution of SYMMLQ by disabling the normal stopping rules. However, that is usually not a null vector of $A$. To obtain a null vector of $A$, we recommend transferring to a CG point at the end or using $\bar{w}_k$ when $\beta_{k+1} = 0$.

**Proposition 2.16 (Transfer to CG point).** *Suppose that $A$ is symmetric positive semidefinite. Let $x_k^C$ denote the $k$th iterate from CG, $e_k^C := x - x_k^C$, and $\phi_k^C$ be the norm of the corresponding residual $r_k^C = b - Ax_k^C$. Then we have the following results:*

1. $x_k^C = x_k + \left(\frac{\zeta_k s_k}{c_k}\right)\overline{w}_{k+1}$.
2. $\left\|x_k^C\right\|_2 = \sqrt{\|x_k\|_2^2 + \left(\frac{\zeta_k s_k}{c_k}\right)^2} \geq \|x_k\|_2$.
3. $\phi_k^C = \frac{\beta_1\beta_{k+1}s_1s_2s_3\cdots s_{k-1}}{\left|\gamma_k^{(1)}\right|} = \frac{|c_{k-1}|s_k}{|c_k|}\phi_{k-1}^C$.

**Lemma 2.17.** *If $\beta_{k+1} = 0$ and $\gamma_k^{(2)} = 0$, then $\bar{w}_k$ is a unit null vector of $A$.*

*Proof.*

$$\beta_{k+1} = 0 \Rightarrow L_k = T_k Q_{k-1} = \begin{bmatrix} \gamma_1^{(1)} & & & & \\ \delta_2^{(1)} & \gamma_2^{(2)} & & & \\ \epsilon_3^{(1)} & \ddots & \ddots & & \\ & \ddots & \ddots & \gamma_{k-1}^{(2)} & \\ & & \epsilon_k^{(1)} & \delta_k^{(2)} & 0 \end{bmatrix},$$

and thus $A\bar{w}_k = AV_kQ_{k-1}e_k = V_kT_kQ_{k-1}e_k = V_kL_ke_k = 0$, $\|\bar{w}_k\|_2 = \|V_kQ_{k-1}e_k\| = 1$. ∎

### 2.2.3 MINRES

MINRES is also built upon the Lanczos process. Within each Lanczos step, we solve the least-squares subproblem

$$y_k = \arg\min_{y \in \mathbb{R}^k} \|\beta_1 e_1 - \underline{T_k} y\|_2, \tag{2.18}$$

by computing the QR factorization

$$Q_k \underline{T_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix} = \begin{bmatrix} \gamma_1^{(1)} & \delta_2^{(1)} & \epsilon_3^{(1)} & & & \\ & \gamma_2^{(2)} & \delta_3^{(2)} & \epsilon_4^{(1)} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \epsilon_k^{(1)} \\ & & & & \ddots & \delta_k^{(2)} \\ & & & & & \gamma_k^{(2)} \\ & & & & & 0 \end{bmatrix}, \qquad Q_k(\beta_1 e_1) = \begin{bmatrix} t_k \\ \phi_k \end{bmatrix}, \tag{2.19}$$

where $Q_k = Q_{k,k+1} \cdots Q_{2,3} Q_{1,2}$ is a product of $(k+1) \times (k+1)$ Householder reflectors designed to annihilate the $\beta_i$'s in the subdiagonal of $\underline{T_k}$. Of course, this is the transpose of the LQ factorization used in SYMMLQ, with $Q_k = P_k^T$ and $Q_{k,k+1} = P_{k,k+1}$ in (2.9)–(2.10). Thus our subproblem becomes

$$y_k = \arg\min_{y \in \mathbb{R}^k} \left\| \begin{bmatrix} t_k \\ \phi_k \end{bmatrix} - \begin{bmatrix} R_k \\ 0 \end{bmatrix} y \right\|_2, \tag{2.20}$$

where $t_k = \begin{bmatrix} \tau_1 & \tau_2 & \cdots & \tau_k \end{bmatrix}^T$ and

$$\begin{bmatrix} t_k \\ \phi_k \end{bmatrix} = \beta_1 Q_{k,k+1} \cdots Q_{2,3} \begin{bmatrix} c_1 \\ s_1 \\ 0_{k-1} \end{bmatrix} = \beta_1 Q_{k,k+1} \cdots Q_{3,4} \begin{bmatrix} c_1 \\ s_1 c_2 \\ s_1 s_2 \\ 0_{k-2} \end{bmatrix} = \beta_1 \begin{bmatrix} c_1 \\ s_1 c_2 \\ \vdots \\ s_1 \cdots s_{k-1} c_k \\ s_1 \cdots s_{k-1} s_k \end{bmatrix}. \tag{2.21}$$

TABLE 2.11
*Algorithm **MINRES**. The algorithm also estimates $\phi = \|r_k\|$, $\psi = \|Ar_k\|$, $\chi = \|x_k\|$, $\mathcal{A} = \|A\|$, $\kappa = \mathrm{cond}(A)$.*

**MINRES**$(A, b, \sigma, \mathrm{maxit}) \to x, \phi, \psi, \chi, \mathcal{A}, \kappa$
$\beta_1 = \|b\|_2, \quad v_0 = 0, \quad \beta_1 v_1 = b, \quad \phi_0 = \tau_0 = \beta_1, \quad \chi_0 = 0, \quad \kappa = 1$
$\delta_1^{(1)} = \gamma_{\min} = 0, \quad c_0 = -1, \quad s_0 = 0, \quad d_0 = d_{-1} = x_0 = 0, \quad k = 1$
**while** no stopping condition is true,
  **LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \to \alpha_k, \beta_{k+1}, v_{k+1}$
  `//last left orthogonalization on middle two entries in last column of` $\underline{T_k}$
  $\delta_k^{(2)} = c_{k-1}\delta_k^{(1)} + s_{k-1}\alpha_k, \quad \gamma_k^{(1)} = s_{k-1}\delta_k^{(1)} - c_{k-1}\alpha_k$
  `//last left orthogonalization to produce first two entries of` $\underline{T_{k+1}}e_{k+1}$
  $\epsilon_{k+1}^{(1)} = s_{k-1}\beta_{k+1}, \quad \delta_{k+1}^{(1)} = -c_{k-1}\beta_{k+1}$
  `//current left orthogonalization to zero out` $\beta_{k+1}$
  **SymOrtho**$(\gamma_k^{(1)}, \beta_{k+1}) \to c_k, s_k, \gamma_k^{(2)}$
  `//right-hand side, residual norms, and matrix norm`
  $\tau_k = c_k\phi_{k-1}, \quad \phi_k = s_k\phi_{k-1}, \quad \psi_{k-1} = \phi_{k-1}\sqrt{(\gamma_k^{(1)})^2 + (\delta_{k+1}^{(1)})^2}$
  **if** $k = 1 \quad \mathcal{A}_k = \sqrt{\alpha_1^2 + \beta_2^2}$ **else** $\mathcal{A}_k = \max\{\mathcal{A}_{k-1}, \sqrt{\beta_k^2 + \alpha_k^2 + \beta_{k+1}^2}\}$ **end**
  `//update solution and matrix condition number`
  **if** $\gamma_k^{(2)} \neq 0$,
    $d_k = \left(v_k - \delta_k^{(2)} d_{k-1} - \epsilon_k^{(1)} d_{k-2}\right)/\gamma_k^{(2)}, \quad x_k = x_{k-1} + \tau_k d_k, \quad \chi_k = \|x_k\|$
    $\gamma_{\min} = \min\{\gamma_{\min}, \gamma_k^{(2)}\}, \quad \kappa = \mathcal{A}_k/\gamma_{\min}$
  **end**
  $k \leftarrow k + 1$
**end**
$x = x_k, \quad \phi = \phi_k, \quad \psi = \phi_k\sqrt{(\gamma_{k+1}^{(1)})^2 + (\delta_{k+2}^{(1)})^2}, \quad \chi = \chi_k, \quad \mathcal{A} = \mathcal{A}_k$

A compact way to describe the action of $Q_{k,k+1}$ is

$$\begin{bmatrix} c_k & s_k \\ s_k & -c_k \end{bmatrix} \left[\begin{array}{ccc|c} \gamma_k^{(1)} & \delta_{k+1}^{(1)} & 0 & \phi_{k-1} \\ \beta_{k+1} & \alpha_{k+1} & \beta_{k+2} & 0 \end{array}\right] = \left[\begin{array}{ccc|c} \gamma_k^{(2)} & \delta_{k+1}^{(2)} & \varepsilon_{k+2}^{(1)} & \tau_k \\ 0 & \gamma_{k+1}^{(1)} & \delta_{k+2}^{(1)} & \phi_k \end{array}\right]. \tag{2.22}$$

MINRES computes $x_k$ in $\mathcal{K}_k(A, b)$ as an approximate solution to our problem $Ax = b$:

$$x_k = V_k y_k = V_k R_k^{-1} t_k =: D_k \begin{bmatrix} t_{k-1} \\ \tau_k \end{bmatrix} = \begin{bmatrix} D_{k-1} & d_k \end{bmatrix} \begin{bmatrix} t_{k-1} \\ \tau_k \end{bmatrix} = x_{k-1} + \tau_k d_k, \tag{2.23}$$

where it can be shown that

$$d_k = \left(v_k - \delta_k^{(2)} d_{k-1} - \epsilon_k^{(1)} d_{k-2}\right)/\gamma_k^{(2)}. \tag{2.24}$$

A careful implementation of MINRES needs memory for at most the matrix $A$ and 5 working $n$-vectors for $v_k$, $v_{k+1}$, $d_{k-1}$, $d_k$, and $x_k$ in each iteration (not counting the vector $b$). There are $2\nu + 9n$ flops per iteration, where $\nu$ is number of nonzeros in $A$.

TABLE 2.12
*Algorithm **CR** [88, Algorithm 6.20].*

**CR**$(A, b, \text{maxit}) \to x, \phi$
$x_0 = 0, \quad r_0 = p_0 = b, \quad z_0 = Ar_0, \quad w_0 = Ap_0, \quad \phi_0 = \|b\|, \quad \mu_0 = r_0^T z_0$
$k = 1$
**while** no stopping condition is true
$\alpha_k = \mu_{k-1} / \|w_{k-1}\|^2, \quad x_k = x_{k-1} + \alpha_k p_{k-1}, \quad r_k = r_{k-1} - \alpha_k w_{k-1}, \quad \phi_k = \|r_k\|$
$z_k = Ar_k, \quad \mu_k = r_k^T z_k, \quad \beta_k = \mu_k / \mu_{k-1}, \quad p_k = r_k + \beta_k p_{k-1}$
$w_k = z_k + \beta_k w_{k-1}, \quad k \leftarrow k + 1$
**end**
$x = x_k, \quad \phi = \phi_k$

Saad [88, Algorithm 6.20] derived a MINRES variant from GMRES (Arnoldi process [3] for solving unsymmetric square linear system) and called it the *conjugate residual* (CR) algorithm. CR to MINRES is like CG to LanczosCG; the residual vectors $r_k$ and their norms in CR and CG are directly computed. CR needs 5 working vectors $(x_k, p_k, r_k, w_k, z_k)$ in memory per iteration, not counting $b$. See Table 2.12 for the algorithm. Note: Saad [88, Algorithm 5.3] listed another algorithm called Minimal Residual (MR) iteration, but this is unrelated to MINRES (we want to caution the reader).

The following lemma gives a recurrence relation for $r_k$. It says that the intermediate $r_k$'s are not orthogonal to $\mathcal{K}_k(A, b)$ except when $\beta_{k+1} = 0$. In that case, $s_k = 0$ and $r_k = -\phi_k v_{k+1}$ is finally orthogonal to $\mathcal{K}_k(A, b)$. The residual norm can be recurred without computing $r_k$.

**Lemma 2.18 ($r_k$ for MINRES and monotonicity of $\|r_k\|_2$).** *$r_k = s_k^2 r_{k-1} - \phi_k c_k v_{k+1}$ and $\|r_k\|_2 = \|r_{k-1}\|_2 s_k$. It follows that $\|r_k\|_2 \le \|r_{k-1}\|_2$.*

Similarly, $\|Ar_k\|$ can be efficiently computed by the following recurrence relation. While $\|r_k\|_2$ is monotonically decreasing, $\|Ar_k\|$ is often observed to be oscillating.

**Lemma 2.19 ($Ar_k$ for MINRES).**

$$Ar_k = \|r_k\| \left( \gamma_{k+1}^{(1)} v_{k+1} + \delta_{k+2}^{(1)} v_{k+2} \right),$$

$$\|Ar_k\| = \|r_k\| \sqrt{\left[\gamma_{k+1}^{(1)}\right]^2 + \left[\delta_{k+2}^{(1)}\right]^2}.$$

**Lemma 2.20 (Recurrence formula for $\|Ax_k\|$ for MINRES).**

$$\|Ax_k\|_2 = \|t_k\|_2 = \left\| \begin{bmatrix} t_{k-1} \\ \tau_k \end{bmatrix} \right\|.$$

**Proposition 2.21.** *If $b \in \mathcal{R}(A)$, and in MINRES $\beta_i > 0$ for $i = 1, \ldots, k$, but $\beta_{k+1} = 0$, then $\gamma_k^{(1)} > 0$ and thus $T_k$ and $R_k$ are nonsingular.*

*Proof.* Suppose $\gamma_k^{(1)} = 0$. Then $s_k = 0$ and thus $r_k = 0$ and $\phi_k = \|r_k\| = s_k \phi_{k-1} = 0$. Then $R_k = Q_k T_k$ is singular—of order $k$ and rank $k-1$—and MINRES will proceed to set $x_k := x_{k-1}$.

It follows that $r_k := r_{k-1}$ and $\phi_k = \phi_{k-1} = 0$. However, this contradicts the fact that MINRES had not stopped at the $(k-1)$th iteration. ■

**Corollary 2.22.** *If in MINRES $\beta_i > 0$ for $i = 1, \ldots, k$, and $\beta_{k+1} = 0$, and $\gamma_k^{(1)} = 0$, then $T_k$ and $R_k$ are singular (both of order $k$ and rank $k-1$) and $b \notin \mathcal{R}(A)$.*

In the following, we review the definition of minimum-length solution or pseudoinverse solution for a linear system. Then we prove that MINRES returns the unique minimum-length solution for any symmetric compatible (possibly singular) system.

**Definition 2.23 (Moore-Penrose conditions and pseudoinverse [50]).** *Given any $m \times n$ matrix $A$, $X$ is the pseudoinverse of $A$ if it satisfies the four Moore-Penrose conditions:*

1. $AXA = A$.
2. $XAX = X$.
3. $(AX)^H = AX$.
4. $(XA)^H = XA$.

**Theorem 2.24 (Existence and uniqueness of the pseudoinverse).** *The pseudoinverse of a matrix always exists and is unique.*

If $A$ is square and nonsingular, then $A^\dagger$, the pseudoinverse of $A$, is the matrix inverse $A^{-1}$.

Even if $A$ is square and nonsingular, we rarely compute $A^{-1}$. Instead, we would compute say the LU decomposition $PA = LU$ or QR decomposition $A = QR$. If we want the solution of $Ax = b$, we do not compute $x = A^{-1}b$ but instead, solve the triangular systems $Ly = Pb$ and $Ux = y$ if we have computed LU decomposition of $A$, or $Rx = Q^Tb$ in the case of QR decomposition. Likewise, we rarely compute the pseudoinverse of $A$. It is mainly an analytical tool. If $A$ is singular, $A^{-1}$ does not exist, but $Ax = b$ may have a solution. In that case, there are infinitely many solutions. In some applications we want the unique minimum-length solution, which could be written in terms of the pseudoinverse of $A$: $x^\dagger = A^\dagger b$. However, to compute $x^\dagger$, we would not compute $A^\dagger$. Instead we could compute some rank-revealing factorization of $A$ such as the *reduced* singular value decomposition $A = U\Sigma V^T$, where $U$ and $V$ have orthogonal columns and $\Sigma$ is diagonal with positive entries. Then the minimum-length solution is $x^\dagger = V\Sigma^{-1}U^Tb$.

**Theorem 2.25.** *If $b \in \mathcal{R}(A)$, and in MINRES $\beta_i > 0$ for $i = 1, \ldots, k$, but $\beta_{k+1} = 0$, then $x_k$ is the pseudoinverse solution of $Ax = b$.*

*Proof.* We know that $\text{span}(v_1, \ldots, v_k) \subseteq \mathcal{R}(A)$. However, we assume $\text{span}(v_1, \ldots, v_k) = \mathcal{R}(A)$. Without this assumption, the result is still true but the proof will be more complicated.

By Proposition 2.21, when $\beta_{k+1} = 0$, $R_k^{-1}$ exists. Moreover,

$$x_k = V_k y_k = V_k R_k^{-1} t_k = V_k R_k^{-1} \beta_1 Q_{k-1} e_1 = V_k R_k^{-1} Q_{k-1} V_k^T b. \tag{2.25}$$

Thus, we define

$$A^\natural := V_k R_k^{-1} Q_{k-1} V_k^T = V_k T_k^{-1} V_k^T \quad \text{since} \quad Q_{k-1} T_k = R_k.$$

TABLE 2.13
*Subproblem definitions of MINRES, GMRES, QMR, and LSQR.*

| Method (Underlying process) | Subproblem (Matrix structure) | Factorization | Estimate of $x_k$ |
|---|---|---|---|
| MINRES [81] (Lanczos [63]) | $y_k = \arg\min_{y\in\mathbb{R}^k} \left\|\underline{T_k}y - \beta_1 e_1\right\|$ ($T_k$ symmetric tridiagonal) | QR: $Q_k\underline{T_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k$ $\in \mathcal{K}_k(A,b)$ |
| GMRES [89] (Arnoldi [3]) | $y_k = \arg\min_{y\in\mathbb{R}^k} \left\|\underline{H_k}y - h_{1,0} e_1\right\|$ ($H_k$ upper Hessenberg) | QR: $Q_k\underline{H_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k$ $\in \mathcal{K}_k(A,b)$ |
| QMR [37] (Lanczos bi-orthogonalization [63]) | $y_k = \arg\min_{y\in\mathbb{R}^k} \left\|\underline{T_k}y - \beta_1 e_1\right\|$ ($T_k$ unsymmetric tridiagonal) | QR: $Q_k\underline{T_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k$ $\in \mathcal{K}_k(A,b)$ |
| LSQR [82, 83] (Golub-Kahan bi-diagonalization [47]) | $y_k = \arg\min_{y\in\mathbb{R}^k} \left\|\underline{B_k}y - \beta_1 e_1\right\|$ ($B_k$ lower bidiagonal) | QR: $Q_k\underline{B_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k$ $\in \mathcal{K}_k(A^TA, A^Tb)$ |

We want so show in the following that $A^\natural$ is the pseudoinverse of $A$ and thus $x_k$ is the minimum-length solution of $Ax = b$. We start with the third and the fourth Moore-Penrose conditions:

$$AA^\natural = AV_kT_k^{-1}V_k^T = V_kT_kT_k^{-1}V_k^T = V_kV_k^T,$$
$$A^\natural A = V_kT_k^{-1}V_k^TA = V_kT_k^{-1}T_kV_k^T = V_kV_k^T,$$

Thus, $AA^\natural$ and $A^\natural A$ are symmetric, meaning $A^\natural$ satisfies the third and fourth Moore-Penrose conditions. Lastly we show $A^\natural$ satisfies the first and the second Moore-Penrose conditions. By our assumption, the columns of $V_k$ span $\mathcal{R}(A)$. Thus $V_kV_k^TA = A$. It follows that $AA^\natural A = V_kV_k^TA = A$ and $A^\natural AA^\natural = V_kV_k^TA = A$. ■

## 2.3 Existing Iterative Methods for Hermitian Least-Squares

When we have a large and sparse Hermitian least-squares problem, MINRES is the natural solver. In each iteration, it solves a least-squares subproblem:

$$\min \left\|\underline{T_k}y_k - \beta_1 e_1\right\|, \qquad x_k = V_k y_k.$$

However, we want to point out that while the MINRES solution is a least-squares solution (where $\|r_k\|$ is minimized), it may not be the minimum-length solution (where $\|y_k\|$ and $\|x_k\|$ are minimized).

In this section, we review MINRES on singular symmetric least-squares problems. We also mention some Krylov subspace methods for sparse least-squares problems when $A$ is not necessarily symmetric. In particular, GMRES and QMR are applicable for $A$ unsymmetric, and LSQR is applicable to any rectangular matrix $A$. These solvers all have subproblems in the form of least-squares problems. See Table 2.13 and Table 2.14.

TABLE 2.14
*Bases and subproblem solutions in MINRES, GMRES, QMR, and LSQR.*

| Method | New basis | $z_k$ | Estimate of $x_k$ |
|---|---|---|---|
| MINRES | $D_k := V_k R_k^{-1}$ | $R_k z_k = \beta_1 \begin{bmatrix} I_k & 0 \end{bmatrix} Q_k e_1$ | $x_k = D_k z_k$ |
| GMRES | – | – | $x_k = V_k y_k$ |
| QMR | $W_k := V_k R_k^{-1}$ | $R_k z_k = \beta_1 \begin{bmatrix} I_k & 0 \end{bmatrix} Q_k e_1$ | $x_k = W_k z_k$ |
| LSQR | $W_k := V_k R_k^{-1}$ | $R_k z_k = \beta_1 \begin{bmatrix} I_k & 0 \end{bmatrix} Q_k e_1$ | $x_k = W_k z_k$ |

### 2.3.1 MINRES

In this section, we want to show that MINRES produces a generalized-inverse solution when we have a least-squares problem $\min \|Ax - b\|$, where $A$ is singular and $b \notin \mathcal{R}(A)$.

The pseudoinverse is a kind of generalized inverse and there are other kinds (see [7]). Generalized inverses of a rank-deficient matrix may not be unique.

**Definition 2.26 (Generalized inverses).** *For $i = 1, 2, 3, 4$, $X$ is the $\{i\}$-inverse of an $m \times n$ matrix $A$ if it satisfies the $i$th Moore-Penrose condition. Likewise, $X$ is the $\{i, j\}$-inverse of $A$ if it satisfies both the $i$th and $j$th Moore-Penrose conditions. Lastly, $X$ is the $\{i, j, k\}$-inverse of $A$ if it satisfies the $i$th, $j$th, and $k$th Moore-Penrose conditions.*

**Theorem 2.27.** *Consider a symmetric linear least-squares problem $\min \|Ax - b\|$, with $A = A^T$ singular and $b \notin \mathcal{R}(A)$. If $\beta_i \neq 0$ for $i = 1, \ldots, k$, $\beta_{k+1} = \gamma_k^{(1)} = 0$ and $\|Ar_k\| = 0$, then $x_k := x_{k-1}$ is a $\{2, 3\}$-inverse solution, meaning $x_k = Xb$ for some $X$ being a $\{2, 3\}$-inverse of $A$. Moreover, $x_k$ is a $\{1, 2, 3\}$-inverse solution if the columns of $V_k$ span $\mathcal{R}(A)$.*

*Proof.* If $\beta_{k+1} = 0$ and also $\gamma_k^{(1)} = 0$ in (2.20), then iteration $k$ is going to be our last iteration in the Lanczos process, and (2.20) becomes the following underdetermined least-squares problem:

$$\min \left\| \begin{bmatrix} R_{k-1} & s \\ 0 & 0 \\ & 0 \end{bmatrix} \begin{bmatrix} y_{k-1} \\ \eta_k \end{bmatrix} - \begin{bmatrix} t_{k-1} \\ \phi_{k-1} \\ 0 \end{bmatrix} \right\| = \min \left\| \begin{bmatrix} R_{k-1} & s \end{bmatrix} \begin{bmatrix} y_{k-1} \\ \eta_k \end{bmatrix} - \begin{bmatrix} t_{k-1} \\ \phi_{k-1} \end{bmatrix} \right\|,$$

where

$$s := \begin{bmatrix} \epsilon_k^{(1)} e_{k-2} \\ \delta_k^{(2)} \end{bmatrix} \neq 0 \quad \text{since} \quad \epsilon_k^{(1)} = s_{k-2}\beta_k.$$

We choose to set $\eta_k = 0$, thus simplifying the subproblem to $R_{k-1} y_{k-1} = t_{k-1}$, which is actually our previous subproblem in the $(k-1)$th iteration. Therefore

$$x_k := x_{k-1} = V_{k-1} y_{k-1} = V_k y_k, \text{ where } y_k := \begin{bmatrix} y_{k-1} \\ 0 \end{bmatrix}, \tag{2.26}$$

$$\|r_k\| = \|r_{k-1}\| = \phi_{k-1} > 0 \text{ (or we would have stopped in the } (k-1)\text{th iteration)}, \tag{2.27}$$

$$\|Ar_k\| = \|Ar_{k-1}\| = \|r_{k-1}\| \sqrt{\left[\gamma_k^{(2)}\right]^2 + \left[\delta_{k+1}^{(2)}\right]^2} = 0, \tag{2.28}$$

since $\beta_{k+1} = \gamma_k^{(1)} = 0 \Longrightarrow \gamma_k^{(2)} = \delta_{k+1}^{(2)} = 0$ by (2.22), confirming that $x_{k-1}$ is our least-squares

solution. Moreover, by (2.26)

$$x_k = V_k y_k = V_k \begin{bmatrix} y_{k-1} \\ 0 \end{bmatrix} = V_k \begin{bmatrix} R_{k-1}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} t_{k-1} \\ \phi_{k-1} \end{bmatrix} = V_k \begin{bmatrix} R_{k-1}^{-1} & 0 \\ 0 & 0 \end{bmatrix} (\beta_1 Q_{k-1} e_1) \tag{2.29}$$

$$= V_k R_k^\sharp Q_{k-1} V_k^T b, \text{ where } R_k^\sharp := \begin{bmatrix} R_{k-1}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \tag{2.30}$$

$$= A^\sharp b, \text{ where } A^\sharp := V_k R_k^\sharp Q_{k-1} V_k^T. \tag{2.31}$$

We will check if $A^\sharp$ satisfies any of the Moore-Penrose conditions in the following. Recall that when $\beta_{k+1} = \gamma_k^{(1)} = 0$, then

$$AV_k = V_{k+1}\underline{T_k} = V_k T_k = V_k Q_{k-1}^T R_k, \qquad R_k = \begin{bmatrix} R_{k-1} & s \\ 0 & 0 \end{bmatrix}, \qquad V_k^T A V_k = T_k. \tag{2.32}$$

First, we show that $A^\sharp$ satisfies the third but not the fourth Moore-Penrose conditions:

$$AA^\sharp = AV_k R_k^\sharp Q_{k-1} V_k^T = V_k Q_{k-1}^T R_k R_k^\sharp Q_{k-1} V_k^T = V_k Q_{k-1}^T \begin{bmatrix} I_{k-1} & \\ & 0 \end{bmatrix} Q_{k-1} V_k^T, \tag{2.33}$$

$$A^\sharp A = V_k R_k^\sharp Q_{k-1} V_k^T A = V_k \overline{R}_k^\sharp Q_{k-1} T_k V_k^T = V_k R_k^\sharp R_k V_k^T, \tag{2.34}$$

so that $AA^\sharp$ is symmetric, but $\left(R_k^\sharp R_k\right)^T = \left[\begin{smallmatrix} I_{k-1} & R_{k-1}^{-1} s \\ 0 & 0 \end{smallmatrix}\right]^T \neq R_k^\sharp R_k$, since $s \neq 0$ by (2.27). Thus $A^\sharp A$ is not symmetric.

Next, we check the first Moore-Penrose condition:

$$AA^\sharp A = (AV_k) R_k^\sharp Q_{k-1} \left(V_k^T A\right) = \left(V_k Q_{k-1}^T R_k\right) R_k^\sharp Q_{k-1} \left(T_k V_k^T\right) \text{ by (2.31) – (2.32)} \tag{2.35}$$

$$= V_k Q_{k-1}^T R_k R_k^\sharp R_k V_k^T = V_k Q_{k-1}^T R_k V_k^T \text{ since it is easy to verify } R_k R_k^\sharp R_k = R_k \tag{2.36}$$

$$= AV_k V_k^T \text{ by (2.32)} \tag{2.37}$$

$$= A(V_{k,\mathcal{R}} + v_{1,\mathcal{N}} c^T)(V_{k,\mathcal{R}} + v_{1,\mathcal{N}} c^T)^T, \text{ where } V_{k,\mathcal{R}} = \begin{bmatrix} v_{1,\mathcal{R}} & \cdots & v_{k,\mathcal{R}} \end{bmatrix} \text{ by (2.2)} \tag{2.38}$$

$$= AV_{k,\mathcal{R}} V_{k,\mathcal{R}}^T = A \text{ if the columns of } V_{k,\mathcal{R}} \text{ span } \mathcal{R}(A). \tag{2.39}$$

Lastly, $A^\sharp$ satisfies the second Moore-Penrose condition:

$$A^\sharp A A^\sharp = V_k R_k^\sharp R_k V_k^T V_k R_k^\sharp Q_{k-1} V_k^T \text{ by (2.34)} \tag{2.40}$$

$$= V_k R_k^\sharp R_k R_k^\sharp Q_{k-1} V_k^T = V_k R_k^\sharp Q_{k-1} V_k^T = A^\sharp \text{ since } R_k^\sharp R_k R_k^\sharp = R_k^\sharp. \tag{2.41}$$

■

**Example 3.** *MINRES on* $\min \|Ax - b\|$ *with* $A = \left[\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right]$, $b = \left[\begin{smallmatrix} 1 \\ 1 \\ 1 \end{smallmatrix}\right]$. *The minimum-length solution is* $x^\dagger = \left[\begin{smallmatrix} 1 \\ 1 \\ 0 \end{smallmatrix}\right]$ *and the residuals are* $r^\dagger = b - Ax^\dagger = \left[\begin{smallmatrix} 0 \\ 0 \\ 1 \end{smallmatrix}\right]$ *and* $Ar^\dagger = 0$. *However, MINRES returns a least-squares solution* $x^\sharp = \left[\begin{smallmatrix} 1 \\ 1 \\ 1 \end{smallmatrix}\right]$ *with residuals* $r^\sharp = b - Ax^\sharp = \left[\begin{smallmatrix} 0 \\ 0 \\ 1 \end{smallmatrix}\right]$ *and* $Ar^\sharp = 0$. *Thus we need a new stopping condition* $\|Ar_k\| \leq \text{tol}$ *and a modified MINRES algorithm to get the minimum-length solution.*

TABLE 2.15
*Algorithm* ***Arnoldi.***

**Arnoldi**$(A, b, \text{maxit}) \to V_k, H_k$
$\beta_1 = \|b\|_2, \quad v_1 = b/\beta_1, \quad k = 0$
**while** $h_{k,k-1} \neq 0$ **and** $k \leq \text{maxit}$
  $k \leftarrow k+1, \quad w := Av_k$
  **for** $i = 1, \ldots, k$ `//modified Gram-Schmidt`
    $h_{i,k} := w^T v_i, \quad w := w - h_{i,k} v_i, \quad h_{k+1,k} = \|w\|_2$
  **end**
  **if** $h_{k+1,k} = 0, \quad v_{k+1} = 0$ **else** $v_{k+1} = w/h_{k+1,k}$ **end**
**end**

### 2.3.2 GMRES

The Lanczos process, the Arnoldi process, and the modified Gram-Schmidt process are closely related. Given $k$ linearly independent vectors $u_1, \ldots, u_k$ in $\mathbb{R}^n$, the modified Gram-Schmidt process generates $k$ orthonormal vectors $v_1, \ldots, v_k$, where each $v_i \in \text{span}\{u_1, \ldots, u_i\}$. Given $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$, modified Gram-Schmidt on $\{b, Ab, \ldots, A^{k-1}b\}$ is called the Arnoldi process, and when $A$ is symmetric, it is equivalent to the Lanczos process.

Given $A$ and $b$, the Arnoldi process computes vectors $v_k$ as follows:

$$\beta_1 v_1 = b, \text{ where } \beta_1 = \|b\|_2 \text{ serves to normalize } v_1, \tag{2.42}$$

$$w_k = Av_k, \quad h_{i,k} = w_k^T v_i,$$

$$h_{k+1,k} v_{k+1} = w_k - h_{1,k} v_1 - \cdots - h_{k,k} v_k, \tag{2.43}$$

where $h_{k+1,k}$ serves to normalize $v_{k+1}$ (see Table 2.15). In matrix form,

$$AV_k = V_{k+1} \underline{H_k}, \text{ where } V_k = \begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix}, \quad \underline{H_k} = \begin{bmatrix} H_k \\ h_{k+1,k} e_k^T \end{bmatrix}, \quad H_k = [h_{i,j}]_{j=1,i=1}^{k,j+1}. \tag{2.44}$$

Note that $H_k$ is an upper Hessenberg matrix. In exact arithmetic the columns of $V_k$ are orthonormal, and the process stops when $h_{k+1,k} = 0$ ($k \leq n$). We then obtain $AV_k = V_k H_k$.

GMRES [89] is an algorithm for solving $Ax = b$ for square and unsymmetric $A$. In each Arnoldi iteration, GMRES is prepared to solve the least-squares subproblem

$$y_k = \arg\min_{y \in \mathbb{R}^k} \|\underline{H_k} y - \beta_1 e_1\|_2$$

and set $x_k = V_k y_k$. All vectors $v_1, \ldots, v_k$ are saved, and only the final $y_k$ and $x_k$ need be computed, using QR factorization of $\begin{bmatrix} \underline{H_k} & \beta_1 e_1 \end{bmatrix}$. We list the algorithm in Table 2.16.

When $A = A^T$, GMRES is mathematically equivalent to MINRES but does not enjoy the short recurrence relation. When $k$ is large, $V_k$ and $\underline{H_k}$ become memory-consuming. For GMRES to be practical on large systems, it is often *restarted* [110, Figure 6.1] every $m$ steps for some small positive integer $m$. However, the convergence properties are then unpredictable except in special cases, and stagnation (lack of progress) may occur for some values of $m$ [88, p. 172].

TABLE 2.16
*Algorithm **GMRES**. This algorithm also estimates $\phi = \|b - Ax\|$.*

**GMRES**$(A, b, \text{tol}, \text{maxit}) \to x, \phi$
$\beta_1 = \|b\|_2, \quad v_1 = b/\beta_1, \quad x_0 = 0, \quad \phi_0 = \beta_1, \quad k = 0$
**while** $(h_{k,k-1} \neq 0)$ **or** $(\phi_i > \text{tol})$ **or** $(k < \text{maxit})$
  $k \leftarrow k + 1, \quad w := Av_k$
  **for** $i = 1, \ldots, k$ `//modified Gram-Schmidt`
    $h_{i,k} := w^T v_i, \quad w := w - h_{i,k} v_i, \quad h_{k+1,k} = \|w\|_2$
  **end**
  **if** $h_{k+1,k} = 0$
    $v_{k+1} = 0$
  **else**
    $v_{k+1} = w/h_{k+1,k}$
    **for** $j = 2, \ldots, k$
      $r^{(2)}_{j-1,k} = c_{j-1} r^{(1)}_{j-1,k} + s_{j-1} h_{j,k}, \quad r^{(1)}_{j,k} = s_{j-1} r^{(1)}_{j-1,k} - c_{j-1} h_{j,k}$
    **end**
    $r^{(2)}_{k,k} = \sqrt{\left[r^{(1)}_{k,k}\right]^2 + h^2_{k+1,k}}, \quad c_k = r^{(1)}_{k,k}/r^{(2)}_{k,k}, \quad s_k = h^{(2)}_{k+1,k}/r^{(2)}_{k,k}$
    $\tau_k = c_k \phi_{k-1}, \quad \bar{\phi}_k = s_k \bar{\phi}_{k-1}, \quad \phi_k = \left|\bar{\phi}_k\right|$
  **end**
**end**
Solve $R_k y_k = t_k$ by back substitution, $\quad x = V_k y_k, \quad \phi = \phi_k$

### 2.3.3 LSQR

Given a linear least-squares problem $\min \|Ax - b\|$, $r := b - Ax$, the Golub-Kahan bidiagonalization [47] may be derived by applying the Lanczos process to the augmented system

$$\begin{bmatrix} I & A \\ A^T & \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix},$$

but the process has structure and is better stated directly. Given $A$ and $b$, the Golub-Kahan process computes two sets of orthogonal vectors $v_k$ and $u_k$ according to

$$\begin{aligned} \beta_1 u_1 &= b, & \alpha_1 v_1 &= A^T u_1, \\ \beta_{k+1} u_{k+1} &= A v_k - \alpha_k u_k, & \alpha_{k+1} v_{k+1} &= A^T u_{k+1} - \beta_{k+1} v_k, \end{aligned}$$

where $\beta_i$ and $\alpha_i$ serve to normalize $u_i$ and $v_i$ respectively. In matrix form,

$$\begin{aligned} AV_k &= U_{k+1} \underline{B_k}, \text{ where } V_k = \begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix}, \\ A^T U_{k+1} &= V_k B_k^T, \text{ where } U_{k+1} = \begin{bmatrix} u_1 & \cdots & u_{k+1} \end{bmatrix}, \end{aligned}$$

TABLE 2.17
*Algorithm **Bidiag1** (the Golub-Kahan process) [47],[82, section 3].*

**Bidiag1**$(A, b, \text{maxit}) \to u_1, \ldots, u_{k+1}, v_1, \ldots, v_{k+1}, \alpha_1, \ldots, \alpha_{k+1}, \beta_1, \ldots, \beta_{k+1}$
$\beta_1 u_1 = b, \qquad \alpha_1 v_1 = A^T u_1, \qquad k = 1$
**while** $\alpha_k \neq 0$ **and** $\beta_k \neq 0$ **and** $k \leq \text{maxit}$
  $\beta_{k+1} u_{k+1} = A v_k - \alpha_k u_k$ `//`$\beta_{k+1}$ `normalizes` $u_{k+1}$ `in 2-norm`
  $\alpha_{k+1} v_{k+1} = A^T u_{k+1} - \beta_{k+1} v_k$ `//`$\alpha_{k+1}$ `normalizes` $v_{k+1}$ `in 2-norm`
  $k \leftarrow k + 1$
**end**

$$B_k := \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \ddots & & \\ & \ddots & \alpha_{k-1} & \\ & & \beta_{k-1} & \alpha_k \end{bmatrix}, \quad \underline{B_k} = \begin{bmatrix} B_k \\ \beta_{k+1} e_k^T \end{bmatrix}.$$

In exact arithmetic, the columns of $V_k$ and $U_k$ are orthonormal and the process stops when $\beta_{k+1} = 0$ or $\alpha_{k+1} = 0$ $(k \leq n)$. Table 2.17 lists a way of implementing the process `Bidiag1` [82, section 3].

LSQR [82, 83] uses the Golub-Kahan process to solve least-squares problems $\min \|Ax - b\|_2$ with $A$ of arbitrary shape and rank. In the $k$th iteration of `Bidiag1`, LSQR solves a subproblem that involves the lower bidiagonal matrix $\underline{B_k}$ of size $(k+1) \times k$:

$$\min \|\underline{B_k} y - \beta_1 e_1\|.$$

Since this is an overdetermined problem, we cannot just apply forward substitution. QR factorization is the natural tool. A sequence of Householder reflectors tranforms $\underline{B_k}$ to an upper bidiagonal matrix:

$$Q_k \begin{bmatrix} \underline{B_k} & \beta_1 e_1 \end{bmatrix} = \begin{bmatrix} R_k & f_k \\ & \phi_{k+1}^{(1)} \end{bmatrix} = \begin{bmatrix} \rho_1 & \theta_2 & & & \phi_1 \\ & \ddots & \ddots & & \vdots \\ & & \rho_{k-1} & \theta_k & \phi_{k-1} \\ & & & \rho_k & \phi_k \\ & & & & \phi_{k+1}^{(1)} \end{bmatrix},$$

where $\rho_i = \rho_i^{(2)}$ and $\phi_i = \phi_i^{(2)}$ in Table 2.18.

The convergence of LSQR depends on the number of distinct nonzero singular values of $A$, as illustrated by the following example.

**Example 4.** *If* $A = \text{diag}([-1\ 1\ i\ -i\ 0])$ *and* $b = e = [1\ 1\ 1\ 1\ 0]^T$, *then* $A$ *has only one distinct nonzero singular value* 1 *and LSQR takes* 1 *iteration (*2 *matrix-vector multiplications) to converge to the minimum-length solution* $x^\dagger = [-1\ 1\ -i\ i\ 0]^T$. *We note that* $A$ *is complex symmetric, but not Hermitian. Hence, CG, MINRES and SYMMLQ are not necessarily applicable to this problem.*

TABLE 2.18
*Algorithm **LSQR** [82, section 4]. This algorithm also estimates $\phi = \|r\|$, $\psi = \|A^T r\|$, where $r = b - Ax$.*

**LSQR**$(A, b, \text{tol}, \text{maxit}) \to x, \phi, \psi$
$\beta_1 u_1 = b, \qquad \alpha_1 v_1 = A^T u_1, \qquad w_1 = v_1, \qquad x_0 = 0,$
$\phi_1^{(1)} = \beta_1, \qquad \rho_1^{(1)} = \alpha_1, \qquad k = 0$
**while** stopping conditions not satisfied
  $k = k + 1$
  $\beta_{k+1} u_{k+1} = A v_k - \alpha_k u_k, \qquad \alpha_{k+1} v_{k+1} = A^T u_{k+1} - \beta_{k+1} v_k$
  **SymOrtho**$(\rho_k^{(1)}, \beta_{k+1}) \to c_k, s_k, \rho_k^{(2)}$
  $\theta_{k+1} = s_k \alpha_{k+1}, \qquad \rho_{k+1}^{(1)} = -c_k \alpha_{k+1}, \qquad \phi_k^{(2)} = c_k \phi_k^{(1)},$
  $\phi_{k+1}^{(1)} = s_k \phi_k^{(1)}, \qquad \psi_k = \phi_{k+1}^{(1)} |\rho_{k+1}^{(1)}|$
  $x_k = x_{k-1} + (\phi_k^{(2)} / \rho_k^{(2)}) w_k, \qquad w_{k+1} = v_{k+1} - (\theta_{k+1} / \rho_k^{(2)}) w_k$
**end**
$x = x_k, \qquad \phi = \phi_{k+1}^{(1)}, \qquad \psi = \psi_k$

### 2.3.4 QMR and SQMR

When a matrix is unsymmetric and short recurrence relations are desired (a property not available in the Arnoldi process), we may use the unsymmetric Lanczos process to produce two sets of biorthogonal vectors $\{v_i\}$ and $\{w_i\}$. If we define $V_k := [\, v_1 \ \cdots \ v_k \,]$ and $W_k := [\, w_1 \ \cdots \ w_k \,]$, then

$$W_k^T V_k = D, \qquad \langle v_1, \ldots, v_k \rangle = \mathcal{K}_k(A, v_1), \qquad \langle w_1, \ldots, w_k \rangle = \mathcal{K}_k(A^T, w_1),$$

where $D$ is a nonsingular diagonal matrix.

Fletcher [35] originated Bi-CG and van der Vorst [109] designed Bi-CGSTAB (a stabilized version) for solving unsymmetric $Ax = b$, both based on the Lanczos biorthogonalization process. They are not intended for incompatible systems.

Freund and Nachtigal's QMR [37] uses different subproblems from those in Bi-CG (more like the least-squares subproblems in MINRES and LSQR). It would apply to incompatible systems if a stopping rule based on $\|Ar_k\|$ were implemented. SQMR [38] is a simplified version for symmetric linear systems. When $A$ is symmetric, QMR and SQMR without preconditioner are mathematically equivalent to MINRES.

## 2.4 Stopping Conditions and Norm Estimates

This section summarizes the stopping conditions and various estimates that may be computed in CG, SYMMLQ and MINRES. Some are new and improved over what we had before. For convergence rates, see [62].

The stopping conditions for the solvers are much more complicated than for the Lanczos process itself. In fact, we recommend a family of stopping conditions in a similar spirit to the suggestions in [82, 83, 2, 84, 80]:

| Lanczos | Normwise relative backward errors (NRBE) | Regularization attempts |
|---|---|---|
| $\beta_{k+1} \le n\|A\|\varepsilon$ | $\|r_k\|_2 / (\|A\|\|x_k\| + \|b\|) \le \text{tol}$ | $\kappa(A) \ge \text{maxcond}$ |
| $k = \text{maxit}$ | $\|Ar_k\|_2 / (\|A\|\|r_k\|) \le \text{tol}$ | $\|x_k\|_2 \ge \text{maxxnorm}$ |

where tol, maxit, maxcond, and maxxnorm are input parameters. All quantities are estimated cheaply by updating estimates from the preceding iteration. The estimate of $\|Ar_k\|$ is needed for incompatible systems.

Different relative residual norms have been defined and we prefer the following:

$$\frac{\|r_k\|_2}{\|A\|_F \|x_k\|_2 + \|b\|_2} \quad \text{and} \quad \frac{\|Ar_k\|_2}{\|A\|_F \|r_k\|_2}, \tag{2.45}$$

or

$$\frac{\|r_k\|_2}{\|A\|_2 \|x_k\|_2 + \|b\|_2} \quad \text{and} \quad \frac{\|Ar_k\|_2}{\|A\|_2 \|r_k\|_2}. \tag{2.46}$$

Relative norms are much more telling than absolute norms when $\|A\|$, $\|b\|$, or $\|x\|$ are tiny or large. Since $\|A\|_F = \sqrt{\sum \sigma_i^2} \ge \sigma_1 = \|A\|_2$, (2.45) could make the algorithms stop sooner than (2.46).

### 2.4.1 Residual and Residual Norm

In CG, $\|r_k\|$ is directly computed while $r_k$ is given by a short recurrence relation. In LanczosCG, it can be shown that

$$r_k = (-1)^k \|r_k\| v_{k+1}, \ \|r_k\|_2 = |\zeta_k| \beta_{k+1}.$$

For SYMMLQ, by Proposition 2.11, $r_0 = \beta_1 v_1$ and $\|r_0\| = \beta_1$. Moreover, if we define $\omega_{k+1} = \gamma_{k+1}^{(2)} \zeta_{k+1}$ and $\varrho_{k+2} = \epsilon_{k+2}^{(1)} \zeta_k$, we have

$$r_k = \omega_{k+1} v_{k+1} - \varrho_{k+2} v_{k+2}, \qquad \|r_k\|_2 = \left\| \begin{bmatrix} \omega_{k+1} & \varrho_{k+2} \end{bmatrix} \right\|.$$

For MINRES, by Lemma 2.18, the residual in the $k$th step is

$$r_k = s_k^2 r_{k-1} - \phi_k c_k v_{k+1}, \qquad \|r_k\|_2 = \phi_k = \phi_{k-1} s_k = \|r_{k-1}\|_2 s_k \le \|r_{k-1}\|_2 .$$

### 2.4.2 Norm of $Ar_k$

For LanczosCG, $\|Ar_k\|$ can be obtained only in iteration $k+1$ when $\beta_{k+2}$ is available:

$$\begin{aligned} Ar_0 &= \|r_0\| (\beta_2 v_2 + \alpha_1 v_1), \qquad \|Ar_0\| = \|r_0\| \sqrt{\alpha_1^2 + \beta_2^2}, \\ Ar_k &= (-1)^k \|r_k\| (\beta_{k+2} v_{k+2} + \alpha_{k+1} v_{k+1} + \beta_{k+1} v_k), \\ \|Ar_k\| &= \|r_k\| \sqrt{\beta_{k+1}^2 + \alpha_{k+1}^2 + \beta_{k+2}^2} \text{ for } k = 1, \ldots. \end{aligned}$$

For CG, by Lemma 2.5, $\|Ar_k\|$ can be computed when $\mu_{k+2}$ and $\nu_{k+1}$ are available in iteration $k+1$:

$$\|Ar_0\| = \|r_0\| \sqrt{\frac{1+\mu_2}{\nu_1^2}},$$
$$\|Ar_k\| = \|r_k\| \sqrt{\frac{\mu_{k+1}}{\nu_k^2}\left(1+\mu_{k+1}+\frac{2\nu_k}{\nu_{k+1}}\right)+\frac{1+\mu_{k+2}}{\nu_{k+1}^2}} \text{ for } k=1,\ldots \text{ when } q_k^T A q_k \neq 0,$$
$$\|Ar_k\| = \|r_k\| \sqrt{\frac{\mu_{k+1}}{\nu_k^2}(1+\mu_{k+1})} \text{ when } q_k^T A q_k = 0.$$

Lemma 2.6 says that CG is good for compatible symmetric linear systems, but not linear least-squares problem. Thus we usually do not compute $Ar_k$ or its norm.

For SYMMLQ, recall from Proposition 2.12, with $\varrho_{k+2} := \epsilon_{k+2}^{(1)}\zeta_k$:

$$\begin{aligned}
Ar_0 &= \beta_1(\alpha_1 v_1 + \beta_2 v_2), \qquad \|Ar_0\| = \beta_1\sqrt{\alpha_1^2+\beta_2^2},\\
Ar_k &= \beta_{k+1}\omega_{k+1}v_k - (\alpha_{k+1}\omega_{k+1} - \beta_{k+2}\varrho_{k+2})v_{k+1}\\
&\quad - (\beta_{k+2}\omega_{k+1} - \alpha_{k+2}\varrho_{k+2})v_{k+2} - \beta_{k+3}\varrho_{k+2}v_{k+3},\\
\|Ar_k\| &= \left\| \begin{bmatrix} \beta_{k+1}\omega_{k+1} \\ \alpha_{k+1}\omega_{k+1} - \beta_{k+2}\varrho_{k+2} \\ \beta_{k+2}\omega_{k+1} - \alpha_{k+2}\varrho_{k+2} \\ -\beta_{k+3}\varrho_{k+2} \end{bmatrix} \right\| \text{ for } k=1,\ldots.
\end{aligned}$$

However, by Lemma 2.13, SYMMLQ is like CG: good for linear systems but not least-squares problems. Thus, we usually do not compute $Ar_k$ or its norm.

Lastly for MINRES, by Lemma 2.19,

$$Ar_k = \|r_k\|\left(\gamma_{k+1}^{(1)}v_{k+1} + \delta_{k+2}^{(1)}v_{k+2}\right), \qquad \|Ar_k\| = \|r_k\|\sqrt{\left[\gamma_{k+1}^{(1)}\right]^2 + \left[\delta_{k+2}^{(1)}\right]^2}.$$

### 2.4.3 Solution Norms

For CG and MINRES, we recommend computing $\|x_k\|$ directly. For SYMMLQ, by Lemma 2.14, we have the following short recurrence relation:

$$\chi_1 = \|x_1\|_2 = \zeta_1, \qquad \|x_k\|_2 = \|z_k\| = \sqrt{\chi_{k-1}^2 + \zeta_k^2}, \qquad k > 1.$$

### 2.4.4 Matrix Norms

The relative stopping conditions (2.45)–(2.46) require estimates of $\|A\|_2$ and $\|A\|_F$. We now discuss a few methods for estimating these two matrix norms.

The MATLAB function `NORMEST` applies the power method on $A^TA$ to estimate $\|A\|_2$ up to some specified tolerance and is recommended for large and sparse $A$. The method could fail for reasons that the power method could fail—for example, if the initial vector is orthogonal to the dominant eigenvector of $A^TA$ or if it lies in the nullspace of $A$. However, unlike the standard power method, it would work even if $A^TA$ has multiple dominant eigenvalues of the

same magnitude because the convergence condition is

$$\left| \|Ax^{(k)}\| - \|Ax^{(k-1)}\| \right| < \text{tol}\, \|Ax^{(k)}\|.$$

**Lemma 2.28 ([107, Theorem 5.3]).** *Let $A = U\Sigma V^T$ be the full singular value decomposition of $A$ with $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)$, where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ and $\sigma_i = 0$ for $i > r = \text{rank}(A)$. Then $\|A\|_2 = \sigma_1$ and $\|A\|_F = \left(\sum_{i=1}^{r} \sigma_i^2\right)^{1/2} = \|\Sigma\|_F \geq \|A\|_2$.*

Given a matrix, we can estimate its singular values from any submatrix.

**Theorem 2.29 (Interlacing property of matrix singular values [102]).** *Suppose $A$ is a complex matrix of size $m \times n$ and $B$ is a submatrix of $A$ obtained by deleting 1 column (or 1 row) from $A$. Define $r := \min\{m, n\}$ and $s := \min\{m, n-1\}$. Clearly, $s = r$ or $s = r - 1$. Let $\sigma_1, \ldots, \sigma_r$ denote the singular values of $A$ sorted in descending order ($\sigma_1 \geq \cdots \geq \sigma_r \geq 0$) and $\gamma_1, \ldots, \gamma_s$ be the singular values of $B$, also in descending order. Then*

$$\sigma_1 \geq \gamma_1 \geq \sigma_2 \geq \gamma_2 \geq \cdots \geq \sigma_{r-1} \geq \gamma_{r-1} \geq \sigma_r \text{ if } s = r - 1,$$
$$\sigma_1 \geq \gamma_1 \geq \sigma_2 \geq \gamma_2 \geq \cdots \geq \sigma_{r-1} \geq \gamma_{r-1} \geq \sigma_r \geq \gamma_r \text{ if } s = r.$$

*In particular, the equalities hold if the deleted column (or row) is a zero vector.*

Since $\|A\|_2 = \sigma_1(A)$, an immediate result of the interlacing theorem is the following.

**Corollary 2.30.** *The two-norm of a matrix is greater than or equal to that of its submatrix.*

**Lemma 2.31.** *For LanczosCG, SYMMLQ, and MINRES, $\|A\|_2 \geq \left\|\underline{T_k}\right\|_2 \geq \|T_k\|_2$.*

*Proof.* Assuming exact arithmetic, $V_k$ has orthonormal columns. It follows that

$$\begin{aligned}\|A\|_2 &= \sup_{\|x\|_2=1} \|Ax\|_2 \geq \sup_{\|V_k x\|_2 = \|x\|_2 = 1} \|AV_k x\|_2 = \|AV_k\|_2 = \left\|V_{k+1}\underline{T_k}\right\|_2 \\ &= \left\|\underline{T_k}\right\|_2 \geq \|T_k\|_2 \text{ by Corollary 2.30.}\end{aligned}$$

■

To estimate a lower bound for $\|A\|$ from within the Lanczos process, we can use $\max_{i=1,\ldots,k} \|p_i\|$ with $p_i = Av_i$, because $v_i$ is a unit vector. The largest $\|p_i\|$ is the best estimate. However, $p_i = Av_i$ is the $i$-th column of $AV_i = V_{i+1}\underline{T_i}$ and hence $p_i = V_{i+1}\underline{T_i}e_i$. A good approximation to $\|p_i\|$ should therefore be

$$\left\|\underline{T_i}e_i\right\|_2 = \left\| \begin{bmatrix} \beta_i & \alpha_i & \beta_{i+1} \end{bmatrix}^T \right\|,$$

which is cheaper than directly computing $\|p_i\|$. Although orthogonality of $V_{i+1}$ is assumed, the norms are essentially the same in practice. See Figure 2.2 to see how typically close they are. In fact, Paige [78, equation (19)] showed that if $A$ has at most $m$ nonzeros per row,

$$\left| \left\| \begin{bmatrix} \beta_k & \alpha_k & \beta_{k+1} \end{bmatrix}^T \right\|^2 - \|p_k\|^2 \right| \leq 4k(3n + 19 + m\,\|p_k - \alpha_k v_k\|)\,\|A\|_2^2\,\varepsilon.$$

We have proven the following lemma, which allows us to estimate a lower bound for $\|A\|_2$ from $k$ samples within SYMMLQ and MINRES, without having to use `NORMEST` as we have recommended in the CG case.

**Lemma 2.32.** *For LanczosCG, SYMMLQ, and MINRES,* $\|A\|_2 \geq \max\{\|\underline{T_1}e_1\|_2, \ldots, \|\underline{T_k}e_k\|_2\}$, *where* $\|\underline{T_i}e_i\|_2 = \|[\,\beta_i \;\alpha_i \;\beta_{i+1}\,]^T\|$. *If we define* $\mathcal{A}_2^{(1)} = \|\underline{T_1}e_1\|_2$, *then* $\mathcal{A}_2^{(k)} = \max\{\mathcal{A}_2^{(k-1)}, \|\underline{T_k}e_k\|_2\}$ *is monotonically increasing and thus gives an improving estimate of* $\|A\|_2$ *as* $k$ *increases.*

A lower bound for $\|A\|_2$ in (2.45) means that the Lanczos-based algorithms may iterate more than necessary, so we want a good lower bound. With the test cases we have run, Lemma 2.32 does seem to provide a good estimate of the order of matrix norms—see Figure 2.2 for the norm estimates on 12 matrices of different sizes from the Florida matrix collection [108].

If (2.46) is to be used instead of (2.45), then the following lemma would be helpful.

**Lemma 2.33.** *For LanczosCG, MINRES, and SYMMLQ,* $\|A\|_F \geq \|\underline{T_k}\|_F$. *Moreover, if we define* $\mathcal{A}_F^{(1)} := \|\underline{T_1}\|_F = \|[\,\alpha_1 \;\beta_2\,]^T\|_2$, *then* $\mathcal{A}_F^{(k)} := \sqrt{(\mathcal{A}_F^{(k-1)})^2 + \|\underline{T_k}e_k\|_2^2}$ *is strictly increasing and thus gives an improving estimate of* $\|A\|_F$ *as* $k$ *increases.*

*Proof.* Assuming exact arithmetic, $V_k$ has orthonormal columns. Let $A = U\Sigma V^T$ be the full singular value decomposition of $A$. It follows that

$$\|\underline{T_k}\|_F = \|V_{k+1}\underline{T_k}\|_F = \|AV_k\|_F = \|U\Sigma V^T V_k\|_F = \|\Sigma V^T V_k\|_F \quad =: \|\Sigma W\|_F\,,$$

where $W = V^T V_k$ is of size $n \times k$ $(n \geq k)$ with orthonormal columns. Thus

$$\begin{aligned}
\|\Sigma W\|_F^2 &= \operatorname{trace}(\Sigma W W^T \Sigma) = \operatorname{trace}(\Sigma Z \Sigma), \text{ where } Z := WW^T \\
&= \sum_{i=1}^{n} \sigma_i^2 z_{ii} = \sum_{i=1}^{n} \sigma_i^2 \sum_{j=1}^{k} w_{ij}^2 \\
&= \sum_{i=1}^{n} \sigma_i^2 \|W(i,:)\|_2^2\,, \text{ where } W(i,:) \text{ denotes the } i\text{th row of } W \\
&\leq \sum_{i=1}^{n} \sigma_i^2 \|W\|_2^2 \text{ by Corollary 2.30} \\
&\leq \sum_{i=1}^{n} \sigma_i^2 = \|\Sigma\|_F^2 = \|A\|_F^2 \text{ since } \sigma_1(W) = 1 = \|W\|_2\,.
\end{aligned}$$

■

Earlier implementations of SYMMLQ and MINRES use $\|A\|_F$ estimated from $\|T_k\|_F$, but it is an upper bound for $\|A\|_2$ by Lemma 2.28. This upper bound actually works well in most test cases we have tried. However, when there are many iterations (meaning $T_k$ has high dimension possibly $> n$), it could be a large overestimate and lead to premature termination.

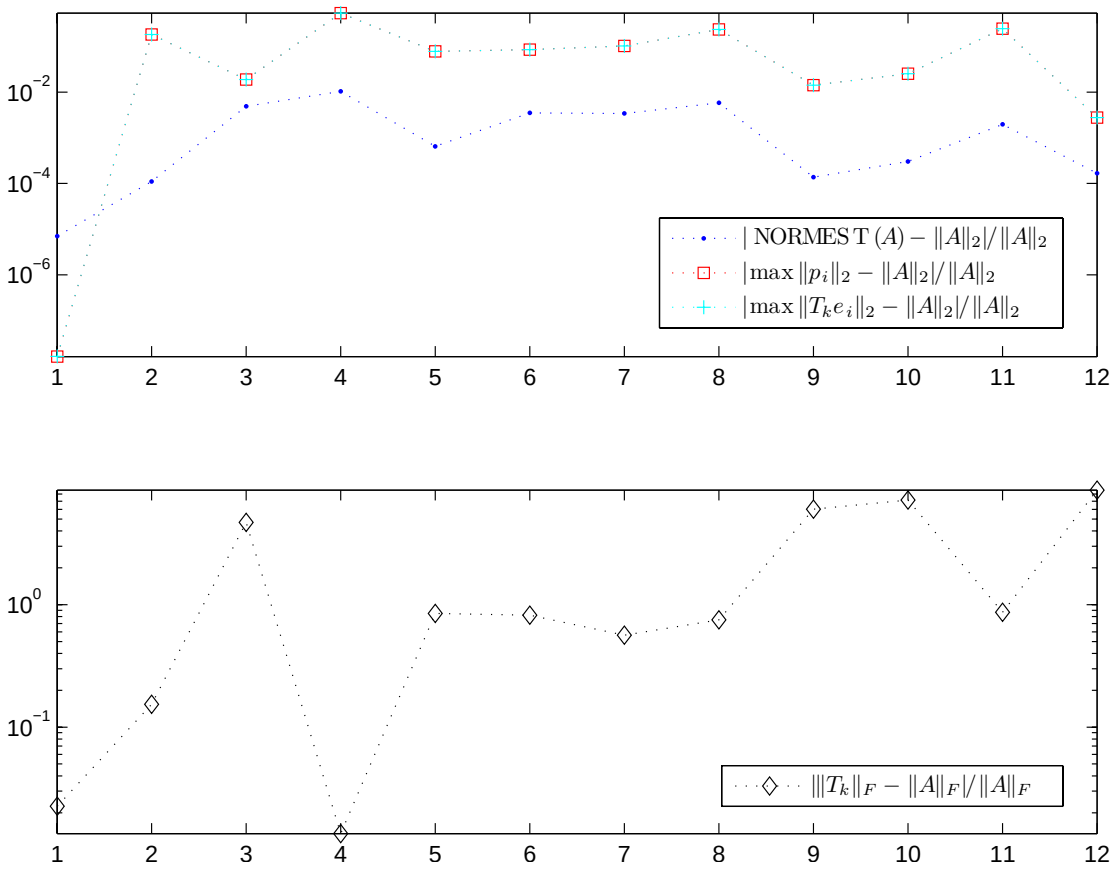


FIGURE 2.2 *Estimating* $\|A\|_2$ *and* $\|A\|_F$ *using different methods on* 12 *test cases. The results show that Lemmas 2.32 and 2.33 provide good estimates of the matrix norms. This figure can be reproduced by* `testminresQLPNormA4`.

### 2.4.5 Matrix Condition Numbers

**Theorem 2.34.** *[107, Theorem* 12.1*, Theorem* 12.2*] Let* $A \in \mathbb{R}^{n\times n}$ *be nonsingular and consider the equation* $Ax = b$*. The problem of computing* $b$*, given* $x$*, has condition number*

$$\kappa_b = \frac{\|A\|\,\|x\|}{\|b\|} \le \kappa(A)$$

*with respect to perturbations of* $x$*. The problem of computing* $x$*, given* $b$*, has condition number*

$$\kappa_x = \frac{\|A^{-1}\|\,\|x\|}{\|b\|} \le \kappa(A).$$

The above theorem says that if $A$ is perturbed by $\Delta A$ and $x$ is given, then the perturbation in $b$ is approximately bounded as follows:

$$\frac{\|\Delta b\|\,/\,\|b\|}{\|\Delta A\|\,/\,\|A\|} \approx \kappa_b \le \kappa(A) \qquad \Longrightarrow \qquad \|\Delta b\| \lesssim \frac{\kappa(A)\,\|b\|\,\|\Delta A\|}{\|A\|}.$$

Likewise, if $A$ is perturbed by $\Delta A$ and $b$ is given, then the perturbation in $x$ is approximately bounded as follows:

$$\frac{\|\Delta x\| / \|x\|}{\|\Delta A\| / \|A\|} \approx \kappa_x \leq \kappa(A) \qquad \Longrightarrow \qquad \|\Delta x\| \lesssim \frac{\kappa(A)\,\|x\|\,\|\Delta A\|}{\|A\|}.$$

Suppose $\|\Delta A\| = O(\varepsilon)$. Then the output could be perturbed by $\frac{\kappa(A)\|b\|}{\|A\|} O(\varepsilon)$. If $\kappa(A)$ is too large, then the perturbed output may no longer be a good approximation. Hence, in iterative methods where matrix-vector multiplications are carried out repeatedly, it is important to monitor $\kappa(A)$.

The matrix condition number with respect to the two-norm could be expensive. If the matrix is triangular, we use the following lemma to estimate the condition number with respect to the infinity norm, which is much cheaper.

**Lemma 2.35 ([41, Exercise 4.9]).** *Given a nonsingular upper-triangular matrix $U$, the diagonal elements of $U^{-1}$ are the reciprocals of the diagonal elements of $U$. Moreover,*

$$\|U\|_\infty \geq \max_i |u_{ii}|, \qquad \left\|U^{-1}\right\|_\infty \geq \frac{1}{\min_i |u_{ii}|}, \qquad \kappa_\infty(U) \geq \frac{\max_i |u_{ii}|}{\min_i |u_{ii}|}. \tag{2.47}$$

Similarly for a nonsingular lower-triangular matrix.

In LanczosCG, since the subproblem $T_k y_k = \beta_1 e_1$ involves $T_k$, we are interested in $\kappa(T_k)$.

$$\begin{aligned}
\|T_k\|_\infty &= \left\|L_k D_k L_k^T\right\|_\infty \leq \|L_k\|_\infty \|D_k\|_\infty \left\|L_k^T\right\|_\infty \approx \max_i \delta_i, \\
\left\|T_k^{-1}\right\|_\infty &= \left\|L_k^{-T} D_k^{-1} L_k^{-1}\right\|_\infty \leq \left\|L_k^{-T}\right\|_\infty \left\|D_k^{-1}\right\|_\infty \left\|L_k^{-1}\right\|_\infty \approx \frac{1}{\min_i \delta_i}, \\
\kappa_\infty(T_k) &= \|T_k\|_\infty \left\|T_k^{-1}\right\|_\infty \approx \frac{\max_i \delta_i}{\min_i \delta_i}.
\end{aligned} \tag{2.48}$$

For CG, it can be shown that $\delta_k = \frac{1}{\nu_k}$, and thus (2.48) becomes

$$\|T_k\|_\infty \approx \frac{1}{\min_i \nu_i}, \qquad \left\|T_k^{-1}\right\|_\infty \approx \max \nu_i, \qquad \kappa_\infty(T_k) \approx \frac{\max_i \nu_i}{\min_i \nu_i}. \tag{2.49}$$

As for MINRES and SYMMLQ, assuming orthogonality of $V_k$,

$$\begin{aligned}
\kappa_2(AV_k) &= \kappa_2(V_{k+1}\underline{T_k}) = \kappa_2(\underline{T_k}) \\
&= \kappa_2(Q_k \underline{T_k}) = \kappa_2(\underline{R_k}) \ = \kappa_2(R_k) \text{ by Theorem 2.29} \\
&= \kappa_2(L_k^T) = \kappa_2(L_k). \\
\kappa_\infty(R_k) &\approx \frac{\max_i \gamma_i^{(2)}}{\min_i \gamma_i^{(2)}}.
\end{aligned} \tag{2.50}$$

# Chapter 3

# MINRES-QLP: an Algorithm for Hermitian Systems

## 3.1 Introduction

This chapter develops the main new algorithm in our thesis: MINRES-QLP. The aim is to deal reliably with singular symmetric systems, and to return the minimum-length least-squares solution. At the same time, we improve the accuracy of MINRES on ill-conditioned nonsingular systems.

### 3.1.1 Effects of Rounding Errors in MINRES

Recall that in the $k$th Lanczos iteration, MINRES is based on the subproblem

$$\min \|\underline{T_k} y_k - \beta_1 e_1\|_2, \qquad x_k = V_k y_k,$$

the QR factorization

$$Q_k \begin{bmatrix} \underline{T_k} & \beta_1 e_1 \end{bmatrix} = \begin{bmatrix} R_k & t_k \\ 0 & \phi_k \end{bmatrix}, \tag{3.1}$$

and the update

$$x_k = (V_k R_k^{-1}) t_k \;\equiv\; D_k t_k \;\equiv\; x_{k-1} + \tau_k d_k.$$

The algorithm should stop if $R_k$ is singular (which would imply singularity of $A$). Singularity was not discussed by Paige and Saunders [81], but they did raise the question: Is MINRES stable when $R_k$ is ill-conditioned? Their concern was that $\|D_k\|$ could be large and there could be cancellation in forming $x_{k-1} + \tau_k d_k$.

Sleijpen, Van der Vorst, and Modersitzki [96] analyze the effects of rounding errors in MINRES and report examples of apparent failure with a matrix of the form $A = QDQ^T$, where $D$ is an ill-conditioned diagonal matrix and $Q$ involves a single Givens rotation. We attempted but unfortunately failed to reproduce MINRES's performance on the two examples defined in Figure 4 of their paper. We modified their examples by using an $n \times n$ Householder transformation for $Q$, and then observed similar problems with MINRES—see Figure 3.1. The recurred residual norms $\phi_k$ are good approximations of the directly computed $\|r_k\|$ until the last few iterations. The $\phi_k$'s then keep decreasing but the directly computed $\|r_k\|$'s become stagnant or even increase.

The analysis in [96] focuses on the rounding errors involved in the $n$ triangular solves for the rows of $D_k$, compared to the single triangular solve $R_k y_k = t_k$ and then $x_k = V_k y_k$ that would be possible (at the final $k$) if $V_k$ were stored as in GMRES. The key feature of MINRES-QLP is that a single (lower) triangular solve suffices with no need to store $V_k$ (much like in SYMMLQ).

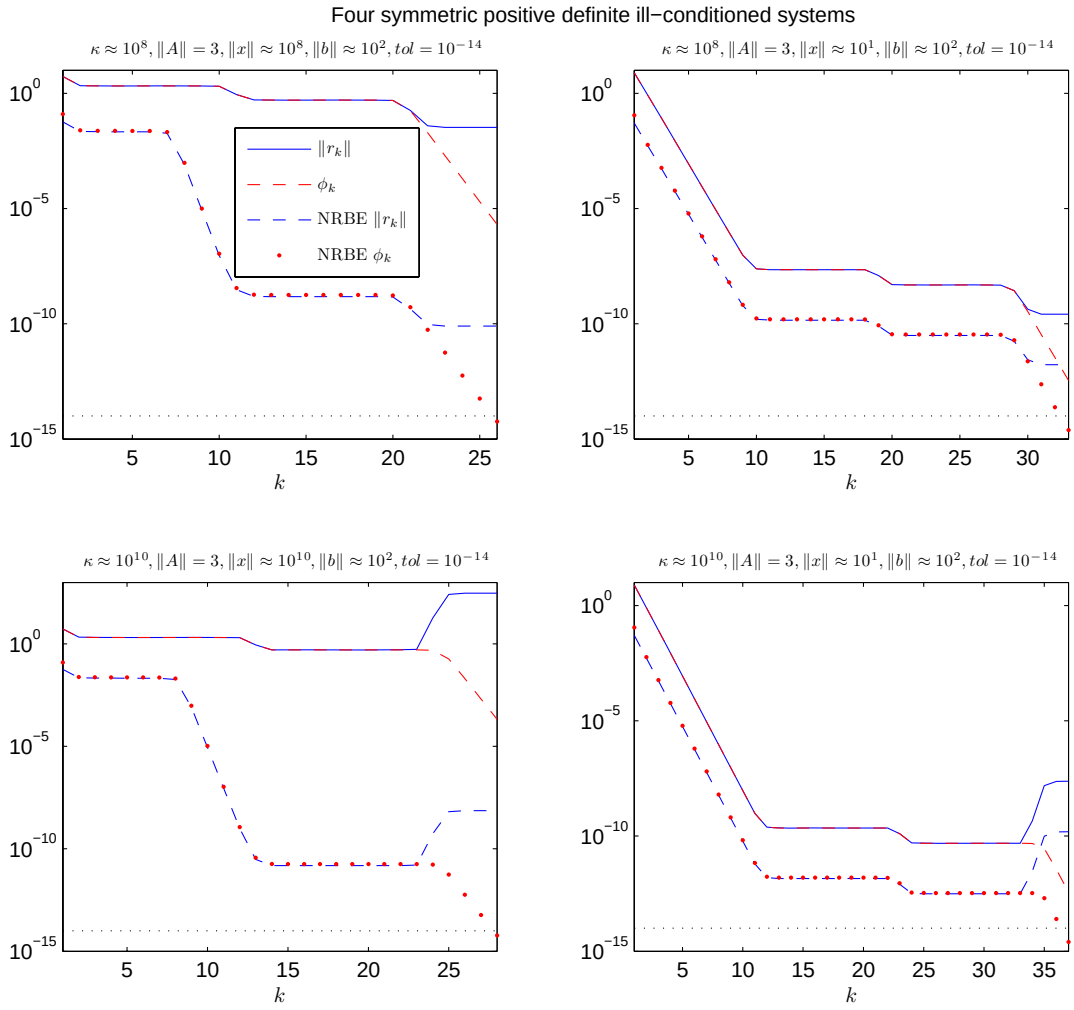


FIGURE 3.1 *MINRES solving* $Ax = b$ *with symmetric positive definite* $A = Q\,\mathrm{diag}([\eta, 2\eta, 2 : \frac{1}{789} : 3])Q$ *of dimension* $n = 792$ *and norm* $\|A\|_2 = 3$, *where* $Q = I - (2/n)ee^T$ *is a Householder matrix generated by* $e = [1, \ldots, 1]^T$. *These plots illustrate the effect of rounding errors in MINRES similar to the examples reported in [96, Figure 4]. The upper part of each plot shows the computed and recurred residual norms, and the lower part shows the computed and recurred normwise relative backward errors (NRBE). MINRES terminates when the recurred NRBE is less than the given* $\mathrm{tol} = 10^{-14}$.

***Upper left:*** $\eta = 10^{-8}$ *and thus* $\kappa(A) \approx 10^8$. *Also* $b = e$ *and therefore* $\|x\| \gg \|b\|$. *The graphs of directly computed residual norms* $\|r_k\|$ *and recurrently computed residual norms* $\phi_k$ *start to differ at iteration 21 when* $\|r_k\| \approx 10^{-1}$. *While the* $\|r_k\|$*'s eventually level off at* $10^{-2}$, *the* $\phi_k$*'s decrease monotonically and are misleadingly small in the last few iterations.*

***Upper right:*** *Again* $\eta = 10^{-8}$ *but* $b = Ae$. *Thus* $\|x\| = \|e\| = O(\|b\|)$. *The graphs of* $\|r_k\|$ *and* $\phi_k$ *start to differ when they reach a much smaller level of* $10^{-10}$ *at iteration 33. The final* $\|r_k\| \approx 10^{-10}$ *is satisfactory but not as accurate as* $\phi_k$ *claims at* $10^{-13}$.

***Lower left:*** $\eta = 10^{-10}$ *and thus* $A$ *is even more ill-conditioned than the matrix in the upper plots. Here* $b = e$ *and* $\|x\|$ *is again exploding. MINRES ends with* $\|r_k\| \approx 10^2 > \|b\|$, *which means no convergence.*

***Lower right:*** $\eta = 10^{-10}$ *and* $b = Ae$. *The solution norm is small and the final* $\|r_k\| \approx 10^{-8}$ *is satisfactory but not as accurate as* $\phi_k$ *claims at* $10^{-13}$.

*This figure can be reproduced from the Matlab program* `DPtest5b.m`.

### 3.1.2 Existing Approaches to Solving Hermitian Least-Squares

If we talk about a symmetric or Hermitian least-squares problem $\min \|Ax - b\|$, we mean that $A$ is singular (otherwise $Ax = b$ is simply a linear system with unique solution).

Inconsistent (singular) symmetric systems could arise from discretized semidefinite Neumann boundary value problems [61, section 6], and naturally any systems involving measurement errors in $b$. Another potential application is large symmetric indefinite singular Toeplitz least-squares problems as described in [39, section 6].

Recall from Theorem 2.27 that MINRES does not give the minimum-length least-squares solution to an inconsistent symmetric system $Ax \approx b$. To obtain the minimum-length solution, we could apply MINRES to various modified *compatible* systems as follows (cf. Theorem 2.25). (We write $A^T$ at times because some of the methods are applicable to general least-squares problems.)

**Normal equations:** The classical symmetric compatible system is $A^TAx = A^Tb$ (or $A^2x = Ab$ in the symmetric case), but when $A$ is ill-conditioned there will be loss of accuracy in forming $A^Tb$ and the products $A^T(Av_k)$ in the Lanczos process.

**Augmented systems:** We could apply MINRES to the larger compatible system

$$\begin{bmatrix} \gamma I & A \\ A^T & \delta I \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix} \tag{3.2}$$

with $\gamma = 1$ and $\delta \leq 0$. However, LSQR already does that more efficiently. On the other hand, a special version of MINRES does seem a viable approach for solving (3.2) when $\gamma > 0$ and $\delta > 0$; in fact, AMRES [97] is such a method. One purpose of AMRES is for computing left or right singular vectors of $A$ by inverse iteration (with $\gamma = \delta = \sigma_i$, a singular value of $A$).

LSQR and AMRES are both based on the Golub-Kahan process for $(A, b)$ in place of Lanczos on the augmented system (3.2), and their convergence rate is governed by the eigenvalues of $A^TA - \gamma\delta I$.

**Two-step procedure:** Another approach [12, Algorithm 6.1] is equivalent to solving one least-squares problem followed by one linear system:

1. Compute a least-squares solution $x_{LS}$ of $\min \|Ax - b\|$, e.g., using MINRES.
2. Compute the minimum-length solution of the compatible system $Ax = Ax_{LS}$.

Note that only $Ax_{LS}$ is required in step 2 (not $x_{LS}$ itself). This could be obtained in various ways. For example, if we know an orthogonal basis $Y$ for the range space $\mathcal{R}(A)$ or an orthogonal basis $Z$ for the null space $\mathcal{N}(A^T)$ (e.g., [17, section 4]), then we have $Ax_{LS} = YY^Tb = b - ZZ^Tb$. In either case, step 1 is not needed.

**MINRES-L:** Bobrovnikova and Vavasis [14] treat weighted least-squares problems by applying MINRES to certain symmetric, indefinite, singular systems.

These approaches are expensive or numerically not ideal. We see the need for a MINRES-like method that can handle singular incompatible systems directly.

### 3.1.3 Orthogonal Matrix Decompositions for Singular Matrices

A *complete orthogonal decomposition* of a singular matrix $A$ takes the form $A = U \left[\begin{smallmatrix} T & 0 \\ 0 & 0 \end{smallmatrix}\right] V$ for some orthogonal matrices $U$ and $V$ and triangular $T$ [10].

The singular value decomposition (SVD) is clearly a complete orthogonal decomposition. It is well known that it is rank-revealing, while the QR decomposition is sometimes not. However, the SVD of a large matrix is usually too expensive to compute.

In 1965 Golub originated *QR decomposition with column pivoting (QRP)* for solving least-squares problems [19, 43]:

$$QA\Pi = R, \tag{3.3}$$

where $Q$ is orthogonal, $\Pi$ a permutation matrix, and $R$ upper triangular. The diagonal elements of $R$, later called the *R-values* by Stewart [100], are generally good estimates for the singular values $\sigma_i$ of $A$. If $A$ is singular, so is $R$, and we can write $R = \left[\begin{smallmatrix} R_1 & S \\ 0 & 0 \end{smallmatrix}\right]$, where $R_1$ is upper triangular and $S$ is rectangular. Although QRP is often rank-revealing, it is not a complete orthogonal decomposition.

Hanson and Lawson in 1969 [56] applied a series of Householder transformations from the right of $\left[\begin{smallmatrix} R_1 & S \end{smallmatrix}\right]$ to yield $\left[\begin{smallmatrix} R_2 & 0 \end{smallmatrix}\right]$, where $R_2$ is upper triangular:

$$QA\Pi H = \begin{bmatrix} R_1 & S \\ 0 & 0 \end{bmatrix} H = \begin{bmatrix} R_2 & 0 \\ 0 & 0 \end{bmatrix}.$$

This is a complete orthogonal decomposition of $A$. It takes advantage of the triangularity of $R_1$.

In 1999 Stewart proposed the *pivoted QLP decomposition* [100], which is equivalent to two consecutive QRP decompositions: on $A$ as before (see (3.3)), then on $R^T$:

$$Q_R A \Pi_R = \begin{bmatrix} R_1 & S \\ 0 & 0 \end{bmatrix}, \qquad Q_L \begin{bmatrix} R_1^T & 0 \\ S^T & 0 \end{bmatrix} \Pi_L = \begin{bmatrix} \bar{R}_2 & 0 \\ 0 & 0 \end{bmatrix}. \tag{3.4}$$

This gives

$$A = QLP, \quad \text{where} \quad Q = Q_R^T \Pi_L, \qquad L = \begin{bmatrix} \bar{R}_2^T & 0 \\ 0 & 0 \end{bmatrix}, \qquad P = Q_L \Pi_R^T, \tag{3.5}$$

with $Q$ and $P$ orthogonal. Stewart demonstrates that the diagonal elements of $L$ (the *L-values*) are better singular-value estimates than the $R$-values, and the accuracy is particularly good for the extreme singular values $\sigma_1$ and $\sigma_n$:

$$R_{ii} \approx \sigma_i, \qquad L_{ii} \approx \sigma_i, \qquad \sigma_1 \geq \max_i L_{ii} \geq \max_i R_{ii}, \qquad \min_i R_{ii} \geq \min_i L_{ii} \geq \sigma_n. \tag{3.6}$$

The first permutation $\Pi_R$ in pivoted-QLP is as important as in QRP. However, the main purpose of the second permutation $\Pi_L$ is to make sure the $L$-values present themselves monotonically decreasing, and is not always necessary. (If $\Pi_R = \Pi_L = I$, we call it simply the *QLP decomposition*.)

As a historical note, Golub and Van Loan [49, section 5.4.2], [50, sections 5.4–5.5] used QRP and then QR to obtain a pivoted QLP decomposition (with $\Pi_L = I$) without naming it so. As here, the context was minimum-length solution of singular least-squares problems.

## 3.2 MINRES-QLP

In this section, we develop MINRES-QLP for solving ill-conditioned or singular symmetric systems $Ax \approx b$. The Lanczos framework is the same as in MINRES, but we allow the subproblem to be singular.

### 3.2.1 The MINRES-QLP Subproblem

When $A$ is singular, both $T_k$ and $\underline{T_k}$ in the Lanczos process could also be singular (rank $< k$). The subproblem that defines MINRES-QLP is therefore chosen to be

$$\min_y \|y\| \quad \text{s.t.} \quad y = \arg\min_{y \in \mathbb{R}^k} \|\underline{T_k} y - \beta_1 e_1\|. \tag{3.7}$$

The solution $y_k$ then defines $x_k = V_k y_k \in \mathcal{K}_k(A, b)$ as the $k$th approximation to $x$. In the nonsingular case, $y_k$ and $x_k$ are the same as in MINRES.

As usual, $y_k$ is not actually computed because all elements change when $k$ increases.

### 3.2.2 Solving the Subproblem

Ideally, we would like to compute a pivoted QLP decomposition of each $\underline{T_k}$. However, for implementation reasons it must be without pivoting, to permit updating of the factors as $k$ increases. Perhaps because of the tridiagonal structure of $\underline{T_k}$ and the convergence properties of the underlying Lanczos process, our experience is that the desired rank-revealing properties are retained.

The unpivoted QLP decomposition is the MINRES QR factorization followed by an LQ factorization of the triangular factor:

$$Q_k \underline{T_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}, \qquad R_k P_k = L_k,$$

where $R_k$ is upper tridiagonal and $L_k$ is lower tridiagonal. As in MINRES, $Q_k$ is a product of Householder reflectors, while $P_k$ involves a product of *pairs* of reflectors:

$$Q_k = \cdots \; Q_{34} \;\; Q_{23} \;\; Q_{12}, \qquad P_k = P_{12} \;\; P_{13}P_{23} \;\; P_{24}P_{34} \;\; P_{35}P_{45} \;\; \cdots .$$

Conceptually, the QR and LQ factorizations could proceed separately as in Figure 3.2 (upper part). However, to be efficient, in the $k$th iteration of MINRES-QLP, the left reflector $Q_{k,k+1}$ and right reflectors $P_{k-2,k}, P_{k-1,k}$ are *interleaved* so that only the lower-right $3 \times 3$ submatrices of $\underline{T_k}$ are changed, as in Figure 3.2 (lower part).

The QLP decomposition allows subproblem (3.7) with $y = P_k u$ to be written equivalently as

$$\min_u \|u\| \quad \text{s.t.} \quad u = \arg\min_{u \in \mathbb{R}^k} \left\| \begin{bmatrix} L_k \\ 0 \end{bmatrix} u - \begin{bmatrix} t_k \\ \phi_k \end{bmatrix} \right\|, \tag{3.8}$$

where $t_k$ and $\phi_k$ are as in (3.1). At iteration $k$, the first $k-3$ components of $u_k$ are already known. The remainder depend on the rank of $L_k$. In particular,

1. if $\text{rank}(L_k) = k$, then we need to solve the bottom three equations of $L_k u_k = t_k$;

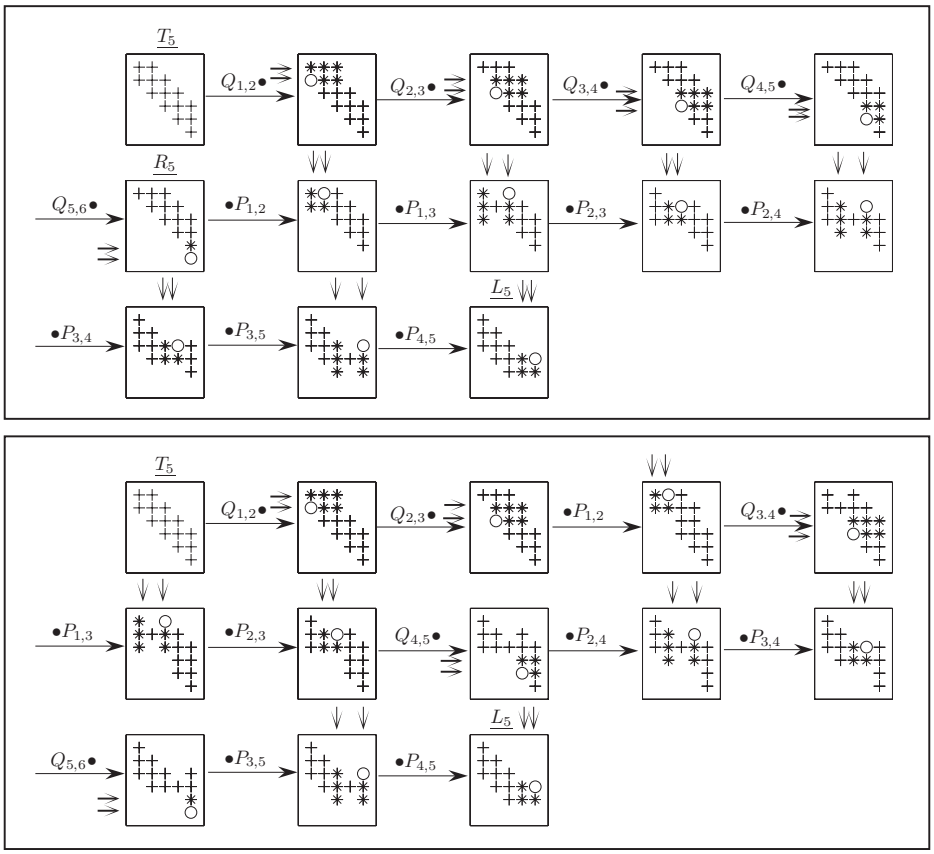


FIGURE 3.2 ***Upper**: QLP with left reflectors and then right reflectors on $\underline{T_5}$. **Lower**: QLP with interleaving left and right reflectors on $\underline{T_5}$. This figure can be reproduced by `QLPfig3.m`.*

2. if $\operatorname{rank}(L_k) = k-1$, then we only need to solve the bottom two equations of

$$L_{k-1}^{(2)} u_{k-1}^{(2)} = t_{k-1}, \qquad L_k = \begin{bmatrix} L_{k-1}^{(2)} & 0 \\ 0 & 0 \end{bmatrix}, \qquad u_k = \begin{bmatrix} u_{k-1}^{(2)} \\ 0 \end{bmatrix}.$$

The corresponding solution estimate $x_k = V_k y_k = V_k P_k u_k$ suggests that we change from the orthonormal basis $V_k$ to another orthonormal basis $W_k = V_k P_k$ in $\mathcal{K}_k(A, b)$ and update $x_k$ by short-recurrence orthogonal steps:

$$\begin{aligned} W_k &= \begin{bmatrix} W_{k-3} & W_k(:, J) \end{bmatrix}, \qquad J = k-2:k, \\ x_k &= W_k u_k = W_{k-3} u_{k-3}^{(3)} + W_k(:, J) u_k(J) \\ &= x_{k-3}^{(2)} + \left( \mu_{k-2}^{(3)} w_{k-2}^{(4)} + \mu_{k-1}^{(2)} w_{k-1}^{(3)} + \mu_k^{(1)} w_k^{(2)} \right) \\ &= \underbrace{x_{k-3}^{(2)} + \mu_{k-2}^{(3)} w_{k-2}^{(4)}}_{x_{k-2}^{(2)}} + \mu_{k-1}^{(2)} w_{k-1}^{(3)} + \mu_k^{(1)} w_k^{(2)}, \end{aligned} \tag{3.9}$$

where $w_j$ and $\mu_j$ refer to columns of $W_k$ and elements of $u_k$, the superscripts show how many times each quantity is updated, and $x_{k-2}^{(2)}$ is needed later (sections 3.2.4 and 3.3.5).

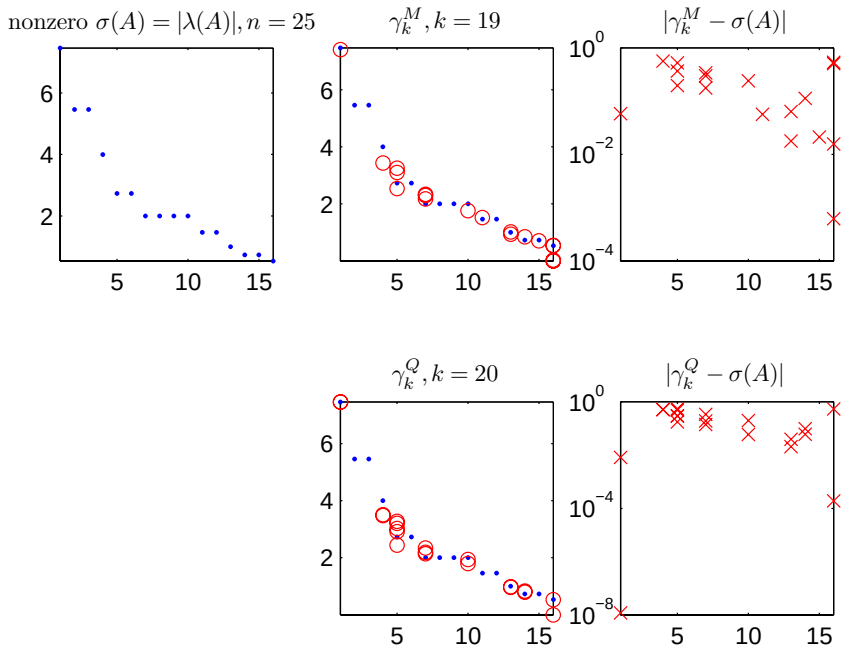


FIGURE 3.3 ***Upper left:*** *Matrix ID 1177 from [108] ($n = 25$). Nonzero singular values of $A$ sorted in decreasing order.* ***Upper middle and right:*** *Each diagonal element $\gamma_i^M$ of $R_k$ from MINRES is plotted as a red circle above or below the nearest nonsingular value of $A$. The $\gamma_i^M$'s approximate the extreme nonzero singular values of $A$ particularly well.* ***Lower:*** *The diagonal elements of $L_k$ (red circles) from MINRES-QLP approximate the extreme nonzero singular values of $A$ even better than those of $R_k$ from MINRES. This figure illustrates equation (3.6). An immediate implication is that the ratio of the largest and smallest diagonals of $L_k$ provides a good estimate of the condition number $\kappa_2(A)$. To reproduce this figure, run* `testminresQLP27(2)`.

### 3.2.3 Further Details

Figure 3.3 illustrates the relation between the singular values of $A$ and the diagonal elements of $R_k$ ($k = 19$) and $L_k$ ($k = 20$). This is for matrix ID 1177 from [108] with $n = 25$.

In MINRES, if $\beta_{k+1} = 0$ then no further work is necessary since $s_k = 0$, $\|r_k\| = \|r_{k-1}\|\, s_k = 0$, and the algorithm stops. In MINRES-QLP, if $\beta_{k+1} = 0$ it is still true that $\|r_k\| = 0$ but we continue to apply reflectors on the right in order to obtain the minimum-length solution.

The following theorem follows from the proofs of Theorem 2.25 and Theorem 2.27 with only slight modification needed.

**Theorem 3.1 (Pseudoinverse solution of MINRES-QLP).** *In MINRES-QLP, if $\beta_i > 0$ for $i = 1, \ldots, k$ but $\beta_{k+1} = 0$, then $x_k$ is the pseudoinverse solution of $Ax \approx b$.*

We list the algorithm developed in the above discussion in Table 3.1. Software code of MINRES-QLP in MATLAB is available as freeware implemented by the author [97]. A detailed derivation of MINRES-QLP is also given there.

MINRES-QLP requires only 1 more vector of storage compared to MINRES. As for flops, MINRES-QLP would need per iteration: 4 more saxpy's ($y \leftarrow \alpha x + y$) and 3 more vector scalings ($x \leftarrow \alpha x$) in comparison with MINRES. We compare MINRES-QLP with CG, SYMMLQ, and MINRES in Tables 3.2 and 3.3.

TABLE 3.1

*Algorithm **MINRES-QLP**. The algorithm also computes $\phi_k = \|r_k\|$, $\psi_k = \|Ar_k\|$, $\chi_k = \|x_k\|$, $\mathcal{A} = \|A\|$, $\kappa = \kappa(A)$, and $\omega = \|Ax_k\|$. The superscript numbers in parentheses indicate how many times the variables have been changed in the program (assuming total number of iterations $k \geq 4$). A more memory-efficient implementation is demonstrated in `minresQLPs.m`.*

**MINRES-QLP**$(A, b, \sigma, \text{maxit}) \rightarrow x, \phi, \psi, \chi, \mathcal{A}, \kappa, \omega$

$\beta_1 = \|b\|_2, \quad \beta_1 v_1 = b, \quad v_{-1} = v_0 = 0, \quad w_0 = w_{-1} = 0, \quad x_{-2} = x_{-1} = x_0 = 0$

$c_{01} = c_{02} = c_{03} = -1, \quad s_{01} = s_{02} = s_{03} = 0, \quad \phi_0 = \beta_1, \quad \tau_0 = \omega_0 = \chi_{-2} = \chi_{-1} = \chi_0 = 0$

$\delta_1^{(1)} = \gamma_{-1} = \gamma_0 = \eta_{-1} = \eta_0 = \eta_1 = \vartheta_{-1} = \vartheta_0 = \vartheta_1 = \mu_{-1} = \mu_0 = 0, \quad \kappa = 1, \quad k = 1$

**while** no stopping condition is true

**LanczosStep**$(A, v_k, v_{k-1}, \beta_k, \sigma) \rightarrow \alpha_k, \beta_{k+1}, v_{k+1}$

**if** $k = 1, \quad \rho_k = \sqrt{\alpha_k^2 + \beta_{k+1}^2}$ **else** $\rho_k = \sqrt{\alpha_k^2 + \beta_k^2 + \beta_{k+1}^2}$ **end**

`//last left orthogonalization on the middle two entries in` $\underline{T_k} e_k$

$\delta_k^{(2)} = c_{k-1,1}\delta_k^{(1)} + s_{k-1,1}\alpha_k, \quad \gamma_k^{(1)} = s_{k-1,1}\delta_k^{(1)} - c_{k-1,1}\alpha_k$

`//last left orthogonalization to produce the first two entries in` $\underline{T_{k+1}} e_{k+1}$

$\epsilon_{k+1}^{(1)} = s_{k-1,1}\beta_{k+1}, \quad \delta_{k+1}^{(1)} = -c_{k-1,1}\beta_{k+1}$

`//current left orthogonalization and first right orthogonalization`

**SymOrtho**$(\gamma_k^{(1)}, \beta_{k+1}) \rightarrow c_{k1}, s_{k1}, \gamma_k^{(2)}, \quad$ **SymOrtho**$(\gamma_{k-2}^{(5)}, \epsilon_k^{(1)}) \rightarrow c_{k2}, s_{k2}, \gamma_{k-2}^{(6)}$

$\delta_k^{(3)} = s_{k2}\vartheta_{k-1}^{(1)} - c_{k2}\delta_k^{(2)}, \quad \gamma_k^{(3)} = -c_{k2}\gamma_k^{(2)}, \quad \eta_k^{(1)} = s_{k2}\gamma_k^{(2)}$

$\vartheta_{k-1}^{(2)} = c_{k2}\vartheta_{k-1}^{(1)} + s_{k2}\delta_k^{(2)}$

`//second right orthogonalization to zero out` $\delta_k^{(3)}$

**SymOrtho**$(\gamma_{k-1}^{(4)}, \delta_k^{(3)}) \rightarrow c_{k3}, s_{k3}, \gamma_{k-1}^{(5)}, \quad \vartheta_k^{(1)} = s_{k3}\gamma_k^{(3)}, \quad \gamma_k^{(4)} = -c_{k3}\gamma_k^{(3)}$

`//update rhs, residual norms, matrix norms and condition no.,` $\|Ax_k\|$

$\tau_k = c_{k1}\phi_{k-1}, \quad \phi_k = s_{k1}\phi_{k-1}, \quad \psi_{k-1} = \phi_{k-1}\sqrt{(\gamma_k^{(1)})^2 + (\delta_{k+1}^{(1)})^2}$

**if** $k = 1, \quad \gamma_{\min} = \gamma_1$ **else** $\gamma_{\min} \leftarrow \min\{\gamma_{\min}, \gamma_{k-2}^{(6)}, \gamma_{k-1}^{(5)}, |\gamma_k^{(4)}|\}$ **end**

$\mathcal{A}_2^{(k)} = \max\{\mathcal{A}_2^{(k-1)}, \rho_k, \gamma_{k-2}^{(6)}, \gamma_{k-1}^{(5)}, |\gamma_k^{(4)}|\}, \quad \kappa \leftarrow \mathcal{A}_2^{(k)}/\gamma_{\min}, \quad \omega_k = \sqrt{\omega_k^2 + \tau_k^2}$

`//update` $w_k$, $x_k$ `and solution norm`

$w_k^{(1)} = -c_{k2}v_k + s_{k2}w_{k-2}^{(3)}, \quad w_{k-2}^{(4)} = s_{k2}v_k + c_{k2}w_{k-2}^{(3)}$

**if** $k > 2$,

$w_k^{(2)} = s_{k3}w_{k-1}^{(2)} - c_{k3}w_k^{(1)}, \quad w_{k-1}^{(3)} = c_{k3}w_{k-1}^{(2)} + s_{k3}w_k^{(1)}$

$\mu_{k-2}^{(3)} = (\tau_{k-2} - \mu_{k-3}^{(3)}\vartheta_{k-2}^{(1)})/\gamma_{k-2}^{(6)}$

**end**

**if** $k > 1, \quad \mu_{k-1}^{(2)} = (\tau_{k-1} - \eta_{k-1}^{(1)}\mu_{k-3}^{(3)} - \vartheta_{k-1}^{(2)}\mu_{k-2}^{(3)})/\gamma_{k-1}^{(5)}$ **end**

**if** $\gamma_k^{(2)} \neq 0, \quad \mu_k^{(1)} = (\tau_k - \eta_k^{(1)}\mu_{k-2}^{(3)} - \vartheta_k^{(1)}\mu_{k-1}^{(2)})/\gamma_k^{(4)}$ **else** $\mu_k^{(1)} = 0$ **end**

$x_{k-2} = x_{k-3} + \mu_{k-2}^{(3)}w_{k-2}^{(3)}, \quad \chi_{k-2} = \sqrt{(\chi_{k-3})^2 + (\mu_{k-2}^{(3)})^2}$

$x_k = x_{k-2} + \mu_{k-1}^{(2)}w_{k-1}^{(3)} + \mu_k^{(1)}w_k^{(2)}, \quad \chi_k = \sqrt{(\chi_{k-2})^2 + (\mu_{k-1}^{(2)})^2 + (\mu_k^{(1)})^2}$

$k \leftarrow k + 1$

**end**

$x = x_k, \quad \phi = \phi_k, \quad \psi = \phi_k\sqrt{(\gamma_{k+1}^{(1)})^2 + (\delta_{k+2}^{(1)})^2}, \quad \chi = \chi_k, \quad \mathcal{A} = \mathcal{A}_2^{(k)}, \quad \omega = \omega_k$

TABLE 3.2
*Subproblem definitions of CG, SYMMLQ, MINRES, and MINRES-QLP.*

| Method | Subproblem | Factorization | Estimate of $x_k$ |
|---|---|---|---|
| LanczosCG or CG [57] | $T_k y_k = \beta_1 e_1$ | Cholesky: $T_k = L_k D_k L_k^T$ | $x_k = V_k y_k \in \mathcal{K}_k(A,b)$ |
| SYMMLQ [81, 90] | $y_{k+1} = \arg\min_{y\in\mathbb{R}^{k+1}} \{\|y\| \mid \underline{T_k}^T y = \beta_1 e_1\}$ | LQ: $\underline{T_k}^T Q_k = \begin{bmatrix} L_k & 0 \end{bmatrix}$ | $x_k = V_{k+1} y_{k+1} \in \mathcal{K}_{k+1}(A,b)$ |
| MINRES [81] | $y_k = \arg\min_{y\in\mathbb{R}^k} \|\underline{T_k} y - \beta_1 e_1\|$ | QR: $Q_k \underline{T_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k \in \mathcal{K}_k(A,b)$ |
| MINRES-QLP | $y_k = \arg\min_{y\in\mathbb{R}^k} \|y\|$ s.t. $y = \arg\min \|\underline{T_k} y - \beta_1 e_1\|$ | QLP: $Q_k \underline{T_k} P_k = \begin{bmatrix} L_k \\ 0 \end{bmatrix}$ | $x_k = V_k y_k \in \mathcal{K}_k(A,b)$ |

TABLE 3.3
*Bases and subproblem solutions in CG, SYMMLQ, MINRES, and MINRES-QLP.*

| Method | New basis | $z_k$ | Estimate of $x_k$ |
|---|---|---|---|
| LanczosCG | $W_k := V_k L_k^{-T}$ | $L_k D_k z_k = \beta_1 e_1$ | $x_k = W_k z_k$ |
| CG | $W_k := V_k L_k^{-T} \Phi_k$ $\Phi_k := \text{diag}(\|r_1\|, \ldots, \|r_k\|)$ | $L_k D_k \Phi_k z_k = \beta_1 e_1$ | $x_k = W_k z_k$ |
| SYMMLQ | $W_k := V_{k+1} Q_k \begin{bmatrix} I_k \\ 0 \end{bmatrix}$ | $L_k z_k = \beta_1 e_1$ | $x_k = W_k z_k$ |
| MINRES | $D_k := V_k R_k^{-1}$ | $R_k z_k = \beta_1 \begin{bmatrix} I_k & 0 \end{bmatrix} Q_k e_1$ | $x_k = D_k z_k$ |
| MINRES-QLP | $W_k := V_k P_k$ | $L_k u_k = \beta_1 \begin{bmatrix} I_k & 0 \end{bmatrix} Q_k e_1$ | $x_k = W_k z_k$ |

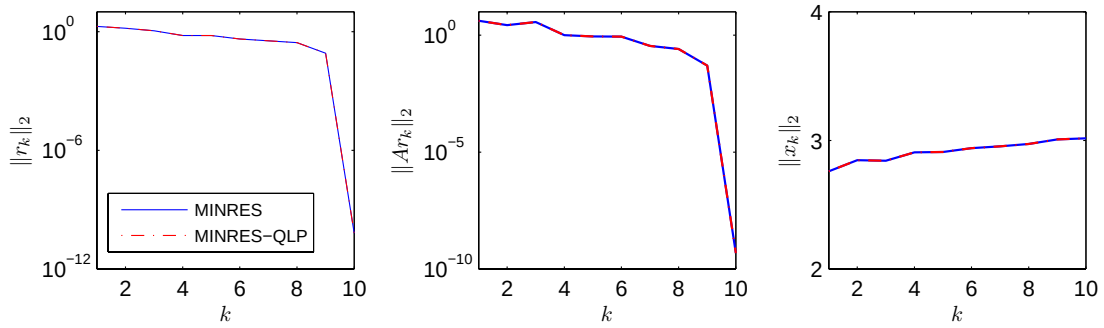


FIGURE 3.4 *The behavior of MINRES and MINRES-QLP is almost identical on a well-conditioned linear system such as this one, where* $\|A\| = \kappa(A) = O(10)$. *This figure can be reproduced by* `testminresQLP27(1)`.

### 3.2.4 Transfer from MINRES to MINRES-QLP

The behavior of MINRES and MINRES-QLP is very similar on well-conditioned linear systems. For example, see Figure 3.4. However, MINRES is cheaper in terms of both memory and flops. Thus it would be a desirable feature to invoke MINRES-QLP from MINRES only if $A$ is ill-conditioned or if we have a least-squares problem at hand. The key idea is to transfer from MINRES to MINRES-QLP at an iteration where $T_k$ is not yet too ill-conditioned. At such a point, the MINRES and MINRES-QLP solution estimates are the same:

$$x_k^M = x_k \qquad \Longleftrightarrow \qquad D_k t_k = W_k u_k = W_k L_k^{-1} t_k.$$

Therefore,

$$W_k = D_k L_k \tag{3.10}$$

and the last three columns of $W_k$ can be obtained from the last three columns of $D_k$ and $L_k$. (Thus, we transfer the three MINRES basis vectors $d_{k-2}, d_{k-1}, d_k$ to $w_{k-2}, w_{k-1}, w_k$.) In addition, we need to generate $x_{k-2}^{(2)}$ using (3.9):

$$x_{k-2}^{(2)} = x_k^M - \mu_{k-1}^{(2)} w_{k-1}^{(3)} - \mu_k^{(1)} w_k^{(2)}. \tag{3.11}$$

It is clear from (3.10) that we still need to do the right orthogonalizations $R_k P_k = L_k$ in the MINRES phase and keep the lower-right $3 \times 3$ submatrix of $L_k$ for each $k$ so that we are ready to transfer to MINRES-QLP when necessary. We then obtain a short recurrence for $\|x_k\|$ (see section 3.3.5) and thus save flops relative to the original MINRES algorithm, where $\|x_k\|$ is computed directly.

In the implementation, an input parameter trancond determines when the transfer from the MINRES iterates to the MINRES-QLP iterates should occur: when an estimate of the condition number of $T_k$ (see (3.20)) exceeds trancond. Thus, trancond $> 1/\varepsilon$ leads to MINRES iterates throughout, while trancond $= 1$ generates MINRES-QLP iterates from the start.

## 3.3 Stopping Conditions and Norm Estimates

This section summarizes the stopping conditions and various estimates computed in MINRES-QLP. Some of them are new and improved estimates relative to those presented in section 2.4. We derive a recurrence relation for $\|x_k\|$ whose cost is as cheap as computing the norm of a 3- or 4- vector. This feature is not available in MINRES.

MINRES-QLP uses the same three groups of stopping conditions as MINRES:

| Lanczos | Normwise relative backward errors (NRBE) | Regularization attempts |
|---|---|---|
| $\beta_{k+1} \le n\|A\|\varepsilon$ | $\|r_k\|_2 / (\|A\|\|x_k\| + \|b\|) \le$ tol | $\kappa(A) \ge$ maxcond |
| $k =$ maxit | $\|Ar_k\|_2 / (\|A\|\|r_k\|) \le$ tol | $\|x_k\|_2 \ge$ maxxnorm |

where tol, maxit, maxcond, and maxxnorm are input parameters.

### 3.3.1 Residual and Residual Norm

The following proposition says that the intermediate $r_k$'s in MINRES-QLP are not orthogonal to $\mathcal{K}_k(A,b)$; only if $\beta_{k+1}=0$, then $s_k=0$ and thus $r_k=-\phi_k v_{k+1}$ is finally orthogonal to $\mathcal{K}_k(A,b)$. Moreover, $\|r_k\|_2$ can be obtained without computing $r_k$.

**Proposition 3.2 ($r_k$ for MINRES-QLP and monotonicity of $\|r_k\|_2$).**

$$r_k = \begin{cases} s_k^2 r_{k-1} - \phi_k c_k v_{k+1} & \text{if } \operatorname{rank}(L_k) = k \\ r_{k-1} & \text{if } \operatorname{rank}(L_k) = k-1 \end{cases},$$

$$\|r_k\|_2 = \begin{cases} \|r_{k-1}\|_2 \, s_k & \text{if } \operatorname{rank}(L_k) = k \\ \|r_{k-1}\|_2 & \text{if } \operatorname{rank}(L_k) = k-1 \end{cases}.$$

*It follows that* $\|r_k\|_2 \le \|r_{k-1}\|_2$.

*Proof.* The residual in the $k$th step is

$$\begin{aligned} r_k &= b - Ax_k = \beta_1 v_1 - AV_k y_k = \beta_1 v_1 - V_{k+1}\underline{T_k} y_k = V_{k+1}\left(\beta_1 e_1 - \underline{T_k} y_k\right) \\ &= V_{k+1} Q_k^T \left( \beta_1 Q_k e_1 - \begin{bmatrix} R_k \\ 0 \end{bmatrix} P_k u_k \right) \text{ where } y_k = P_k u_k \\ &= V_{k+1} Q_k^T \left( \begin{bmatrix} t_k \\ \phi_k \end{bmatrix} - \begin{bmatrix} L_k \\ 0 \end{bmatrix} u_k \right) \text{ where } t_k =: \begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \\ \tau_k \end{bmatrix} = \beta_1 \begin{bmatrix} c_1 \\ s_1 c_2 \\ \vdots \\ s_1 s_2 \cdots s_{k-1} c_k \end{bmatrix} \\ &= V_{k+1} Q_k^T \left( \begin{bmatrix} t_k \\ \phi_k \end{bmatrix} - \begin{bmatrix} L_k u_k \\ 0 \end{bmatrix} \right). \end{aligned} \tag{3.12}$$

Note that $\phi_k = \beta_1 s_1 s_2 \cdots s_{k-1} s_k \ge 0$ since $\beta_i = \|v_i\|_2 \ge 0$ and $s_i = \frac{\beta_{i+1}}{\sqrt{\left[\delta_i^{(1)}\right]^2 + \beta_{i+1}^2}} \ge 0$.

**Case 1** If $\operatorname{rank}(L_k) = k$, we can solve $L_k u_k = t_k$ and simplify (3.12):

$$\begin{aligned} r_k &= \phi_k V_{k+1} Q_k^T e_{k+1} \qquad (3.13) \\ &= \phi_k V_{k+1} \begin{bmatrix} Q_{k-1}^T & \\ & 1 \end{bmatrix} Q_{k,k+1} e_{k+1} \\ &= \phi_k \begin{bmatrix} V_k & v_{k+1} \end{bmatrix} \begin{bmatrix} Q_{k-1}^T & \\ & 1 \end{bmatrix} \begin{bmatrix} I_{k-1} & & \\ & c_k & s_k \\ & s_k & -c_k \end{bmatrix} \begin{bmatrix} 0_{k-1} \\ 0 \\ 1 \end{bmatrix} \\ &= \phi_k \begin{bmatrix} V_k & v_{k+1} \end{bmatrix} \begin{bmatrix} Q_{k-1}^T & \\ & 1 \end{bmatrix} \begin{bmatrix} s_k e_k \\ -c_k \end{bmatrix} = \phi_k \begin{bmatrix} V_k & v_{k+1} \end{bmatrix} \begin{bmatrix} s_k Q_{k-1}^T e_k \\ -c_k \end{bmatrix} \qquad (3.14) \\ &= \phi_k s_k V_k Q_{k-1}^T e_k - \phi_k c_k v_{k+1} = \phi_{k-1} s_k^2 V_k Q_{k-1}^T e_k - \phi_k c_k v_{k+1} \\ &= s_k^2 r_{k-1} - \phi_k c_k v_{k+1} \text{ by (3.13)}. \qquad (3.15) \end{aligned}$$

By (3.14), the recurrence relation for the $k$th residual norm is

$$\|r_k\|_2 = \phi_k \left\| \begin{bmatrix} s_k Q_{k-1}^T e_k \\ -c_k \end{bmatrix} \right\|_2 = \phi_k \sqrt{\left\| s_k Q_{k-1}^T e_k \right\|_2^2 + c_k^2} = \phi_k \sqrt{\|s_k e_k\|_2^2 + c_k^2} = \phi_k \sqrt{s_k^2 + c_k^2}$$
$$= \phi_k = \phi_{k-1} s_k = \|r_{k-1}\|_2 \, s_k.$$

**Case 2** If $\text{rank}(L_k) = k-1$, then the last column and row of $L_k$ are zero. Thus $\|r_k\|_2 = \|r_{k-1}\|_2$.

■

### 3.3.2 Norm of $Ar_k$

Next we want to derive recurrence relations for $Ar_k$ and its norm. The following proposition also shows that $Ar_k$ is orthogonal to $\mathcal{K}_k(A, b)$.

**Proposition 3.3 ($Ar_k$ for MINRES-QLP).**

$$Ar_k = \begin{cases} \|r_k\| \left( \gamma_{k+1}^{(1)} v_{k+1} + \delta_{k+2}^{(1)} v_{k+2} \right) & \text{if } \text{rank}(L_k) = k \\ Ar_{k-1} & \text{if } \text{rank}(L_k) = k-1 \end{cases},$$
$$\|Ar_k\| = \begin{cases} \|r_k\| \sqrt{\left[\gamma_{k+1}^{(1)}\right]^2 + \left[\delta_{k+2}^{(1)}\right]^2} & \text{if } \text{rank}(L_k) = k \\ \|Ar_{k-1}\| & \text{if } \text{rank}(L_k) = k-1 \end{cases}.$$

*Proof.* **Case 1** If $\text{rank}(L_k) = k$, by (3.13),

$$Ar_k = \phi_k A V_{k+1} Q_k^T e_{k+1} = \phi_k V_{k+2} \underline{T_{k+1}} Q_k^T e_{k+1}. \tag{3.16}$$

Using the recurrence relations

$$\underline{T_{k+1}} = \begin{bmatrix} T_{k+1} \\ \beta_{k+2} e_{k+1}^T \end{bmatrix} = \begin{bmatrix} T_k & \beta_{k+1} e_k \\ \beta_{k+1} e_k^T & \alpha_{k+1} \\ & \beta_{k+2} \end{bmatrix},$$

we have

$$\begin{aligned} Q_k \underline{T_{k+1}^T} &= Q_k \begin{bmatrix} T_k & \beta_{k+1} e_k & \\ \beta_{k+1} e_k^T & \alpha_{k+1} & \beta_{k+2} e_{k+1} \end{bmatrix} \\ &= \begin{bmatrix} \begin{bmatrix} R_k \\ 0 \end{bmatrix} & \beta_{k+1} Q_k e_k + \alpha_{k+1} Q_k e_{k+1} & \beta_{k+2} Q_k e_{k+1} \end{bmatrix} \\ &= \begin{bmatrix} \begin{bmatrix} R_k \\ 0 \end{bmatrix} & \beta_{k+1} \begin{bmatrix} s_{k-1} e_{k-1} \\ -c_{k-1} c_k \\ -c_{k-1} s_k \end{bmatrix} + \alpha_{k+1} \begin{bmatrix} s_k e_k \\ -c_k \end{bmatrix} & \beta_{k+2} \begin{bmatrix} s_k e_k \\ -c_k \end{bmatrix} \end{bmatrix} \end{aligned} \tag{3.17}$$

since

$$Q_k e_k = Q_{k,k+1}\cdots Q_{2,3}Q_{1,2}e_k = Q_{k,k+1}\cdots Q_{2,3}e_k = Q_{k,k+1}Q_{k-1,k}e_k$$
$$= \begin{bmatrix} I_{k-1} & & \\ & c_k & s_k \\ & s_k & -c_k \end{bmatrix} \begin{bmatrix} I_{k-2} & & \\ & c_{k-1} & s_{k-1} \\ & s_{k-1} & -c_{k-1} \\ & & & 1 \end{bmatrix} e_k$$
$$= \begin{bmatrix} I_{k-1} & & \\ & c_k & s_k \\ & s_k & -c_k \end{bmatrix} \begin{bmatrix} 0_{k-2} \\ s_{k-1} \\ -c_{k-1} \\ 0 \end{bmatrix} = \begin{bmatrix} 0_{k-2} \\ s_{k-1} \\ -c_{k-1}c_k \\ -c_{k-1}s_k \end{bmatrix},$$

$$Q_k e_{k+1} = Q_{k,k+1}\cdots Q_{2,3}Q_{1,2}e_{k+1} = Q_{k,k+1}\cdots Q_{2,3}e_{k+1} = Q_{k,k+1}e_{k+1} = \begin{bmatrix} s_k e_k \\ -c_k \end{bmatrix}.$$

Hence the last row of (3.17) gives

$$\underline{T_{k+1}}Q_k^T e_{k+1} = \begin{bmatrix} (-\beta_{k+1}c_{k-1}s_k - \alpha_{k+1}c_k)e_{k+1} \\ -\beta_{k+2}c_k \end{bmatrix} = \begin{bmatrix} (\delta_{k+1}^{(1)}s_k - \alpha_{k+1}c_k)e_{k+1} \\ \delta_{k+2}^{(1)} \end{bmatrix} \text{ by (2.22)}$$
$$= \begin{bmatrix} \gamma_{k+1}^{(1)}e_{k+1} \\ \delta_{k+2}^{(1)} \end{bmatrix} \text{ by (2.22)}.$$

Therefore, (3.16) becomes

$$Ar_k = \phi_k\left(\gamma_{k+1}^{(1)}v_{k+1} + \delta_{k+2}^{(1)}v_{k+2}\right), \quad \psi_k := \|Ar_k\|_2 = \|r_k\|\sqrt{\left(\gamma_{k+1}^{(1)}\right)^2 + \left(\delta_{k+2}^{(1)}\right)^2}.$$

**Case 2** If $\operatorname{rank}(L_k) = k-1$, then $Ar_k = Ar_{k-1}$ and $\psi_k := \|Ar_k\| = \|Ar_{k-1}\|$.

■

### 3.3.3 Matrix Norms

If $A$ is symmetric and nonsingular, then $\kappa_2(A) = \frac{\max|\lambda_i|}{\min|\lambda_i|} = \|A\|_2\|A^{-1}\|_2$. However, if $A$ is symmetric and singular, then $\kappa_2(A) = \frac{\max|\lambda_i|}{\min_{\lambda_i\neq 0}|\lambda_i|} = \|A\|_2\|A^{\dagger}\|_2$. In both cases, we say that $A$ is ill-conditioned if the smallest nonzero eigenvalue in magnitude is tiny relative to the largest eigenvalue in magnitude. Consequently, an ill-conditioned matrix has a large condition number and $\kappa_2(A) \gg \|A\|_2$. For a well-conditioned matrix such as $\operatorname{diag}(1,3)$, we may have $\kappa_2(A) = \|A\|_2$.

As in MINRES and Lemma 2.32, the following is used to estimate $\|A\|_2$ in MINRES-QLP:

$$\|A\|_2 \geq \max\left\{\left\|\underline{T_1}e_1\right\|_2, \ldots, \left\|\underline{T_k}e_k\right\|_2\right\}, \quad \text{where} \quad \left\|\underline{T_i}e_i\right\|_2 = \left\|\begin{bmatrix} \beta_i \\ \alpha_i \\ \beta_{i+1} \end{bmatrix}\right\|.$$

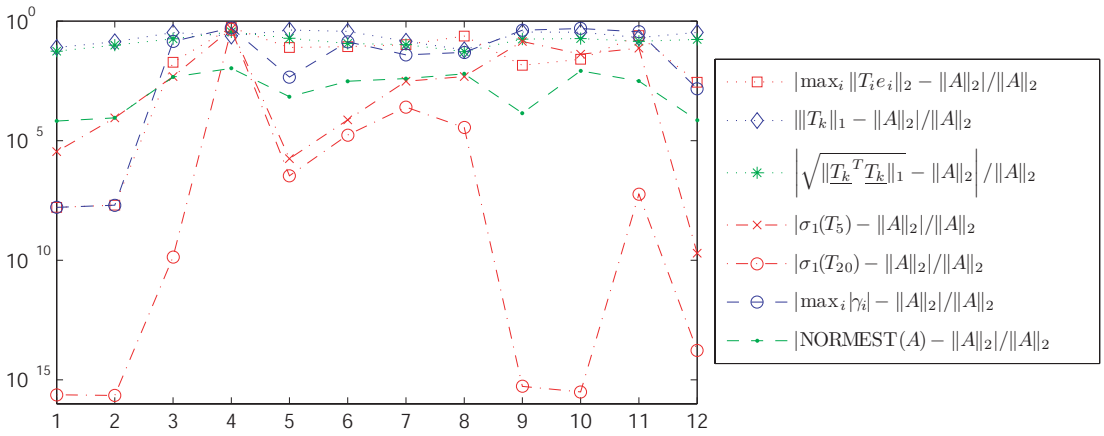


FIGURE 3.5 *Estimating $\|A\|_2$ and $\|A\|_F$ using different methods in MINRES-QLP. These 12 test cases show that provides a good estimate of the order of matrix norms. This figure can be reproduced by* `testminresQLPNormA4`.

If we define $\mathcal{A}_2^{(1)} := \left\|\underline{T_1} e_1\right\|_2$, then

$$\mathcal{A}_2^{(k)} := \max\{\mathcal{A}_2^{(k-1)}, \left\|\underline{T_k} e_k\right\|_2\} \tag{3.18}$$

is monotonically increasing and is thus an improving estimate for $\|A\|_2$ as $k$ increases. By the property of QLP decomposition in (3.6), we could easily extend (3.18) to

$$\mathcal{A}_2^{(k)} := \max\{\mathcal{A}_2^{(k-1)}, \left\|\underline{T_k} e_k\right\|_2, \gamma_{k-2}, \gamma_{k-1}, |\gamma_k|\}. \tag{3.19}$$

Some other schemes inspired by Larsen [66, section A.6.1] and Higham [58] follow:

1. $\|T_k\|_1 \geq \|T_k\|_2$
2. $\sqrt{\|\underline{T_k}^T \underline{T_k}\|_1} \geq \|T_k\|_2$
3. $\|T_j\|_2 \leq \|T_k\|_2$ for small $j = 5, 20$
4. MATLAB function NORMEST($A$), which is based on the power method

Figure 3.5 plots estimates of $\|T_k\|_2$ ($\leq \|A\|_2$ by Lemma 2.31) for 12 matrices from the Florida matrix collection [108], whose sizes $n$ vary from 25 to 3002. In particular, scheme 3 above with $j = 20$ gives significantly more accurate estimates than other schemes for the 12 matrices we tried. However, the choice of $j$ is not always clear and the scheme certainly adds to the cost of MINRES-QLP. Hence we propose incorporating it into MINRES-QLP (or other Lanzcos based iterative methods) only if very accurate $\|A\|_2$ is needed. Otherwise (3.19) uses quantities readily available from MINRES-QLP and gives us satisfactory estimates for the order of $\|A\|_2$.

### 3.3.4 Matrix Condition Numbers

We again apply the property of QLP decomposition in (3.6) to estimate $\kappa_2(T_k)$, which is a lower bound for $\kappa_2(A)$:

$$\kappa_2(A) \geq \kappa_2(T_k) \approx \frac{\max_k \gamma_k}{\min_k \gamma_k}. \tag{3.20}$$

### 3.3.5 Solution Norms

Since $\|x_k\| = \|V_k P_k u_k\| = \|u_k\|$, we can estimate $\|x_k\|$ by computing $\chi_k := \|u_k\|$. However, the last two components of $u_k$ change in $u_{k+1}$ (and a new component $\mu_{k+1}^{(1)}$ is added). We therefore maintain

$$\xi = \|u_k(1:k-2)\| \quad = \|x_{k-2}^{(2)}\| \quad \text{cf. (3.9)}$$

by updating its previous value and then using it according to

$$\xi \leftarrow \left\| \begin{bmatrix} \xi \\ \mu_{k-2}^{(3)} \end{bmatrix} \right\|, \qquad \chi_k = \|x_k\| = \left\| \begin{bmatrix} \xi \\ \mu_{k-1}^{(2)} \\ \mu_k^{(1)} \end{bmatrix} \right\|.$$

Thus $\xi$ increases monotonically but we cannot guarantee that $\|x_k\|$ and its recurred estimate $\chi_k$ are increasing, and indeed they do not in some examples (e.g., see Figures 1.3 and 3.6).

The following example illustrates the regularizing effect of MINRES-QLP with the stopping condition $\chi_k \leq$ maxxnorm.

**Example 5.** *For $k \geq 18$ in Figure 3.6, we observe the following numerical values:*

$$\chi_{18} = \left\| \begin{bmatrix} 2.51 & 3.87 \times 10^{-11} & 1.38 \times 10^{2} \end{bmatrix}^T \right\| = 1.38 \times 10^{2},$$
$$\chi_{19} = \left\| \begin{bmatrix} 2.51 & -8.00 \times 10^{-10} & -1.52 \times 10^{2} \end{bmatrix}^T \right\| = 1.52 \times 10^{2},$$
$$\chi_{20} = \left\| \begin{bmatrix} 2.51 & 1.62 \times 10^{-10} & -1.62 \times 10^{6} \end{bmatrix}^T \right\| = 1.62 \times 10^{6} > \text{maxxnorm} = 10^{4}.$$

*Because the last value exceeds* maxxnorm*, MINRES-QLP regards the last diagonal element of $L_k$ as a singular value to be ignored (in the spirit of truncated SVD solutions). It discards the last component of $u_{20}$ and updates*

$$\chi_{20} \leftarrow \left\| \begin{bmatrix} 2.51 & 1.62 \times 10^{-10} & 0 \end{bmatrix}^T \right\| = 2.51.$$

The full truncation strategy used in the implementation is justified by the fact that $x_k = W_k u_k$ with $W_k$ orthogonal. When $\|x_k\|$ becomes large, the last element of $u_k$ is treated as zero. If $\|x_k\|$ is still large, the second-to-last element of $u_k$ is treated as zero. If $\|x_k\|$ is *still* large, the third-to-last element of $u_k$ is treated as zero.

### 3.3.6 Projection of Right-hand Side onto Krylov Subspaces

In least-squares problems, sometimes projections of the right-hand side vector $b$ onto $\mathcal{K}_k(A, b)$ are required [91]. We can derive a simple recurrence relation for $\|Ax_k\|$:

$$\|Ax_k\| = \|AV_k y_k\| = \|V_{k+1}\underline{T_k} y_k\| \approx \|Q_k \underline{T_k} y_k\| = \|\underline{R_k} y_k\| = \|t_k\|.$$

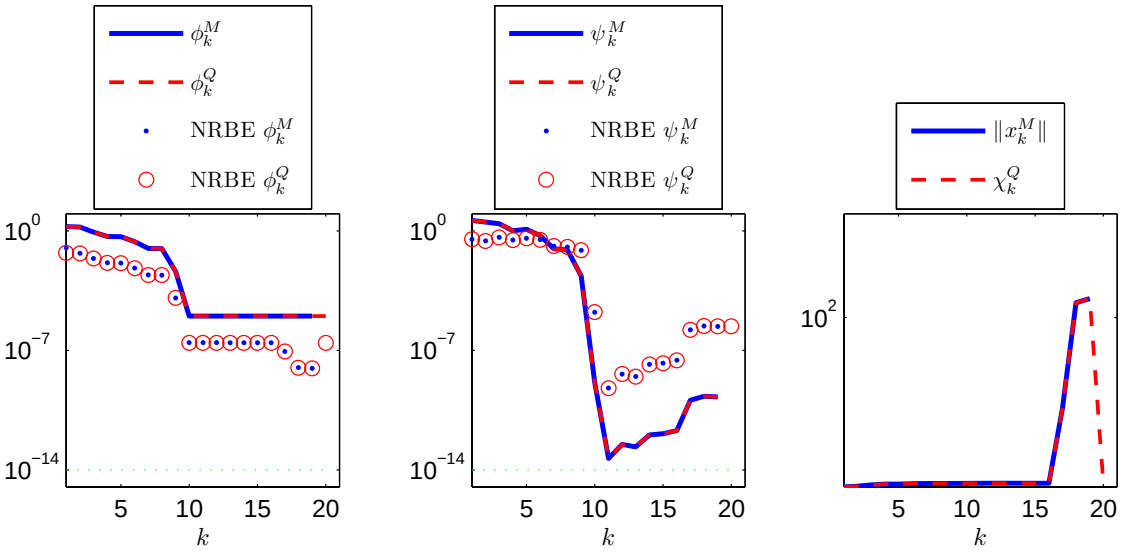


FIGURE 3.6 *Recurred $\|r_k\|$, $\|Ar_k\|$, and $\|x_k\|$ for MINRES and MINRES-QLP. The matrix $A$ (ID* 1177 *from the Florida matrix collection) is positive semidefinite, and $b$ is randomly generated with $\|b\| \simeq 1.7$. Both solvers could have achieved essentially the TEVD solution of $Ax \simeq b$ at iteration* 11*. However, the stringent tolerance $tol = 10^{-14}$ on the recurred normwise relative backward errors (NRBE) $\phi_k$ and $\psi_k$ prevents them from stopping "in time". MINRES ends with an exploding solution, yet MINRES-QLP manages to bring the exploding solution back to the TEVD solution at iteration* 20*—like a magical touch—see Example 5 for numerical details.* ***Left:*** *$\phi_k^M$ and $\phi_k^Q$ are recurred $\|r_k\|$ of MINRES and MINRES-QLP respectively, and their NRBE.* ***Middle:*** *$\psi_k^M$ and $\psi_k^Q$ are recurred $\|Ar_k\|$ and and their NRBE.* ***Right:*** *$\|x_k^M\|$ are norms of solution estimates from MINRES and $\chi_k^Q$ = recurred $\|x_k\|$ from MINRES-QLP with $maxnorm = 10^4$. This figure can be reproduced by* `testminresQLP27(2)`*.*

Therefore,

$$\omega_k := \|Ax_k\|_2 = \left\| \begin{bmatrix} \omega_{k-1} \\ \tau_k \end{bmatrix} \right\|, \qquad \omega_0 = 0. \tag{3.21}$$

## 3.4 Preconditioned MINRES and MINRES-QLP

It is often asked: How to construct a preconditioner for a linear system so that the same problem is solved with fewer iterations? Previous work on preconditioning the symmetric solvers CG, SYMMLQ, or MINRES includes [76, 40, 29, 34, 74, 86, 73, 51, 52, 6, 105].

We have the same question for singular symmetric equations $Ax = b$, and for symmetric least-squares problems $Ax \approx b$.

In all cases, two-sided preconditioning is generally needed in order to preserve symmetry. We can still solve compatible singular systems, but we will no longer obtain the minimum-length solution. For incompatible systems, preconditioning alters the "least squares" norm. In this case we must work with larger equivalent systems that are compatible.

We consider each case in turn, using a positive definite preconditioner $M = CC^T$ with MINRES and MINRES-QLP to solve symmetric compatible systems $Ax = b$. Implicitly, we are solving equivalent symmetric systems $C^{-1}AC^{-T}y = C^{-1}b$, where $C^Tx = y$. As usual, it is possible to arrange the algebra in terms of $M$ itself, so without loss of generality we can assume $C = M^{\frac{1}{2}}$, where $M = VDV^T$ (its eigensystem) and $M^{\frac{1}{2}} = VD^{\frac{1}{2}}V^T$.

As the preconditioned conjugate-gradient method is often abbreviated PCG, we denote the preconditioned MINRES algorithms as PMINRES and PMINRES-QLP respectively.

### 3.4.1 Derivation

Let $\widetilde{A} := M^{-\frac{1}{2}} A M^{-\frac{1}{2}}$ and $\widetilde{b} := M^{-\frac{1}{2}} b$. Given the linear system $Ax = b$, we derive PMINRES by applying MINRES to the equivalent problem

$$\widetilde{A}\widetilde{x} = \widetilde{b}, \qquad M^{\frac{1}{2}} x = \widetilde{x}. \tag{3.22}$$

#### Preconditioned Lanczos Process

Let $\widetilde{v}_k$ denote the Lanczos vectors of $\mathcal{K}(\widetilde{A}, \widetilde{b})$. For notational convenience, we define $\widetilde{v}_0 = 0$, and as before, $\widetilde{\beta}_1 \widetilde{v}_1 = \widetilde{b}$. For $k = 1, 2, \ldots$ we define

$$z_k = \widetilde{\beta}_k M^{\frac{1}{2}} \widetilde{v}_k, \qquad q_k = \widetilde{\beta}_k M^{-\frac{1}{2}} \widetilde{v}_k, \qquad \text{so that} \quad M q_k = z_k. \tag{3.23}$$

Then

$$\widetilde{\beta}_k = \|\widetilde{\beta}_k \widetilde{v}_k\| = \|M^{-\frac{1}{2}} z_k\| = \|z_k\|_{M^{-1}} = \|q_k\|_M = \sqrt{q_k^T z_k},$$

where the square root is well-defined because $M$ is positive definite, and the Lanczos iteration is

$$\begin{aligned}
\widetilde{p}_k &= \widetilde{A}\widetilde{v}_k = M^{-\frac{1}{2}} A M^{-\frac{1}{2}} \widetilde{v}_k = \frac{1}{\widetilde{\beta}_k} M^{-\frac{1}{2}} A q_k, \\
\widetilde{\alpha}_k &= \widetilde{v}_k^T \widetilde{p}_k = \frac{1}{\widetilde{\beta}_k^2} q_k^T A q_k, \\
\widetilde{\beta}_{k+1} \widetilde{v}_{k+1} &= M^{-\frac{1}{2}} A M^{-\frac{1}{2}} \widetilde{v}_k - \widetilde{\alpha}_k \widetilde{v}_k - \widetilde{\beta}_k \widetilde{v}_{k-1}.
\end{aligned}$$

Multiplying the last equation by $M^{\frac{1}{2}}$ we get

$$\begin{aligned}
z_{k+1} = \widetilde{\beta}_{k+1} M^{\frac{1}{2}} \widetilde{v}_{k+1} &= A M^{-\frac{1}{2}} \widetilde{v}_k - \widetilde{\alpha}_k M^{\frac{1}{2}} \widetilde{v}_k - \widetilde{\beta}_k M^{\frac{1}{2}} \widetilde{v}_{k-1} \\
&= \frac{1}{\widetilde{\beta}_k} A q_k - \frac{\widetilde{\alpha}_k}{\widetilde{\beta}_k} z_k - \frac{\widetilde{\beta}_k}{\widetilde{\beta}_{k-1}} z_{k-1}.
\end{aligned} \tag{3.24}$$

The last expression involving consecutive $z_j$'s replaces the three-term recurrence in $\widetilde{v}_j$'s. In addition, we need to solve a linear system $M q_k = z_k$ (3.23) in each iteration.

#### PMINRES

Applying reflectors $\widetilde{Q}_k$ to $\underline{\widetilde{T_k}}$ and $\widetilde{\beta}_1 e_1$, we have

$$\underline{\widetilde{R_k}} = \widetilde{Q}_k \underline{\widetilde{T_k}} = \begin{bmatrix}
\widetilde{\gamma}_1^{(2)} & \widetilde{\delta}_2^{(2)} & \widetilde{\epsilon}_3^{(1)} & \\
& \ddots & \ddots & \widetilde{\epsilon}_k^{(1)} \\
& & \ddots & \widetilde{\delta}_k^{(2)} \\
& & & \widetilde{\gamma}_k^{(2)} \\
& & & 0
\end{bmatrix}, \qquad \begin{bmatrix} \widetilde{t}_k \\ \widetilde{\phi}_k \end{bmatrix} = \widetilde{\beta}_1 \widetilde{Q}_k e_1 = \begin{bmatrix} \widetilde{\tau}_1 \\ \vdots \\ \vdots \\ \widetilde{\tau}_k \\ \widetilde{\phi}_k \end{bmatrix}, \tag{3.25}$$

which defines the subproblem $\min_{y\in\mathbb{R}^k} \left\|\underline{\widetilde{R}_k} y - \widetilde{t}_k\right\|$. Changing basis of the subproblem from $\widetilde{V}_k$ to $\widetilde{D}_k = \widetilde{V}_k \widetilde{R}_k^{-1}$, where the $k$th column of $\widetilde{V}_k$ is $\widetilde{v}_k$ and $\widetilde{x}_k = \widetilde{D}_k \widetilde{t}_k$, we have the following recurrence for the $k$th column of $\widetilde{D}_k$ and $\widetilde{x}_k$:

$$\widetilde{d}_k = \frac{1}{\widetilde{\gamma}_k^{(2)}} \left( \widetilde{v}_k - \widetilde{\delta}_k^{(2)} \widetilde{d}_{k-1} - \widetilde{\epsilon}_k^{(1)} \widetilde{d}_{k-2} \right), \qquad \widetilde{x}_k = \widetilde{x}_{k-1} + \widetilde{\tau}_k \widetilde{d}_k.$$

Multiplying the above two equations by $M^{-\frac{1}{2}}$ on the left and defining $d_k = M^{-\frac{1}{2}} \widetilde{d}_k$, we can update the solution of our original problem by

$$d_k = \frac{1}{\widetilde{\gamma}_k^{(2)}} \left( \frac{1}{\widetilde{\beta}_k} q_k - \widetilde{\delta}_k^{(2)} d_{k-1} - \widetilde{\epsilon}_k^{(1)} d_{k-2} \right), \qquad x_k = M^{-\frac{1}{2}} \widetilde{x}_k = x_{k-1} + \widetilde{\tau}_k d_k.$$

We list the algorithm in Table 3.4.

TABLE 3.4
*Algorithm **PMINRES**. Preconditioned MINRES.*

**PMINRES**$(A, b, M, \sigma, \text{maxit}) \to x, \widetilde{\phi}, \widetilde{\psi}$
$z_0 = 0, \quad z_1 = b, \quad$ Solve $Mq_1 = z_1, \quad \widetilde{\beta}_1 = \sqrt{b^T q_1}$
$\widetilde{\delta}_1^{(1)} = 0, \quad d_0 = d_{-1} = x_0 = 0, \quad \widetilde{c}_0 = -1, \quad \widetilde{s}_0 = 0, \quad k = 1$
**while** no stopping condition is true
  $p_k = Aq_k, \quad \widetilde{\alpha}_k = \frac{1}{\widetilde{\beta}_k^2} q_k^T p_k, \quad z_{k+1} = \frac{1}{\widetilde{\beta}_k} p_k - \frac{\widetilde{\alpha}_k}{\widetilde{\beta}_k} z_k - \frac{\widetilde{\beta}_k}{\widetilde{\beta}_{k-1}} z_{k-1}$
  Solve $Mq_{k+1} = z_{k+1}, \quad \widetilde{\beta}_{k+1} = \sqrt{q_{k+1}^T z_{k+1}}$
  $\widetilde{\delta}_k^{(2)} = \widetilde{c}_{k-1} \widetilde{\delta}_k^{(1)} + \widetilde{s}_{k-1} \widetilde{\alpha}_k, \quad \widetilde{\gamma}_k^{(1)} = \widetilde{s}_{k-1} \widetilde{\delta}_k^{(1)} - \widetilde{c}_{k-1} \widetilde{\alpha}_k,$
  $\widetilde{\epsilon}_{k+1}^{(1)} = \widetilde{s}_{k-1} \widetilde{\beta}_{k+1}, \quad \widetilde{\delta}_{k+1}^{(1)} = -\widetilde{c}_{k-1} \widetilde{\beta}_{k+1}, \quad$ **SymOrtho**$(\widetilde{\gamma}_k^{(1)}, \widetilde{\beta}_{k+1}) \to \widetilde{c}_k, \widetilde{s}_k, \widetilde{\gamma}_k^{(2)}$
  $\widetilde{\tau}_k = \widetilde{c}_k \widetilde{\phi}_{k-1}, \quad \widetilde{\phi}_k = \widetilde{s}_k \widetilde{\phi}_{k-1}, \quad \widetilde{\psi}_{k-1} = \widetilde{\phi}_{k-1} \sqrt{\left(\widetilde{\gamma}_k^{(1)}\right)^2 + \left(\widetilde{\delta}_{k+1}^{(1)}\right)^2}$
  **if** $\widetilde{\gamma}_k^{(2)} \neq 0$,
    $d_k = \frac{1}{\widetilde{\gamma}_k^{(2)}} \left( \frac{1}{\widetilde{\beta}_k} q_k - \widetilde{\delta}_k^{(2)} d_{k-1} - \widetilde{\epsilon}_k^{(1)} d_{k-2} \right), \quad x_k = x_{k-1} + \widetilde{\tau}_k d_k$
  **end**
  $k \leftarrow k + 1$
**end**
$x = x_k, \quad \widetilde{\phi} = \widetilde{\phi}_k, \quad \widetilde{\psi}_k = \widetilde{\phi}_k \sqrt{\left(\widetilde{\gamma}_{k+1}^{(1)}\right)^2 + \left(\widetilde{\delta}_{k+2}^{(1)}\right)^2}$

### PMINRES-QLP

PMINRES-QLP can be derived very similarly. See Table 3.5. The additional work is to apply right reflectors $\widetilde{P}_k$ to $\widetilde{R}_k$, and the new subproblem bases are $\widetilde{W}_k := \widetilde{V}_k \widetilde{P}_k$, with $\widetilde{x}_k = \widetilde{W}_k \widetilde{u}_k$. Multiplying the new basis and solution estimate by $M^{-\frac{1}{2}}$ on the left, we obtain

$$W_k := M^{-\frac{1}{2}} \widetilde{W}_k = M^{-\frac{1}{2}} \widetilde{V}_k \widetilde{P}_k, \tag{3.26}$$

$$x_k = M^{-\frac{1}{2}} \widetilde{x}_k = M^{-\frac{1}{2}} \widetilde{W}_k \widetilde{u}_k = W_k \widetilde{u}_k = x_{k-2}^{(2)} + \widetilde{\mu}_{k-1}^{(2)} w_{k-1}^{(3)} + \widetilde{\mu}_k^{(1)} w_k^{(2)}. \tag{3.27}$$

TABLE 3.5
*Algorithm **PMINRES-QLP**. Preconditioned MINRES-QLP.*

**PMINRES-QLP**$(A, b, M, \sigma, \text{maxit}) \to x, \widetilde{\phi}, \widetilde{\psi}, \widetilde{\chi}, \widetilde{\mathcal{A}}, \widetilde{\kappa}, \widetilde{\omega}$

$z_0 = 0, \qquad z_1 = b, \qquad$ Solve $Mq_1 = z_1, \qquad \widetilde{\beta}_1 = \sqrt{b^T q_1}$

$w_0 = w_{-1} = 0, \qquad x_{-2} = x_{-1} = x_0 = 0$

$\widetilde{c}_{01} = \widetilde{c}_{02} = \widetilde{c}_{03} = -1, \qquad \widetilde{s}_{01} = \widetilde{s}_{02} = \widetilde{s}_{03} = 0, \quad \widetilde{\phi}_0 = \widetilde{\beta}_1, \quad \widetilde{\tau}_0 = \widetilde{\omega}_0 = \widetilde{\chi}_{-2} = \widetilde{\chi}_{-1} = \widetilde{\chi}_0 = 0$

$\widetilde{\delta}_1^{(1)} = \widetilde{\gamma}_{-1} = \widetilde{\gamma}_0 = \widetilde{\eta}_{-1} = \widetilde{\eta}_0 = \widetilde{\eta}_1 = \widetilde{\vartheta}_{-1} = \widetilde{\vartheta}_0 = \widetilde{\vartheta}_1 = \widetilde{\mu}_{-1} = \widetilde{\mu}_0 = 0, \qquad \widetilde{\kappa} = 1, \qquad k = 1$

**while** no stopping condition is true

`//preconditioned Lanczos Step`

$p_k = Aq_k, \qquad \widetilde{\alpha}_k = \frac{1}{\widetilde{\beta}_k^2} q_k^T p_k, \qquad z_{k+1} = \frac{1}{\widetilde{\beta}_k} p_k - \frac{\widetilde{\alpha}_k}{\widetilde{\beta}_k} z_k - \frac{\widetilde{\beta}_k}{\widetilde{\beta}_{k-1}} z_{k-1}$

Solve $Mq_{k+1} = z_{k+1}, \qquad \widetilde{\beta}_{k+1} = \sqrt{q_{k+1}^T z_{k+1}}$

**if** $k = 1, \qquad \widetilde{\rho}_k = \sqrt{\widetilde{\alpha}_k^2 + \widetilde{\beta}_{k+1}^2}$ **else** $\widetilde{\rho}_k = \sqrt{\widetilde{\alpha}_k^2 + \widetilde{\beta}_k^2 + \widetilde{\beta}_{k+1}^2}$ **end**

`//last left orthogonalization on the middle two entries in` $\underline{T_k} e_k$

$\widetilde{\delta}_k^{(2)} = \widetilde{c}_{k-1,1} \widetilde{\delta}_k^{(1)} + \widetilde{s}_{k-1,1} \widetilde{\alpha}_k, \qquad \widetilde{\gamma}_k^{(1)} = \widetilde{s}_{k-1,1} \widetilde{\delta}_k^{(1)} - \widetilde{c}_{k-1,1} \widetilde{\alpha}_k$

`//last left orthogonalization to produce the first two entries in` $\underline{T_{k+1}} e_{k+1}$

$\widetilde{\epsilon}_{k+1}^{(1)} = \widetilde{s}_{k-1,1} \widetilde{\beta}_{k+1}, \qquad \widetilde{\delta}_{k+1}^{(1)} = -\widetilde{c}_{k-1,1} \widetilde{\beta}_{k+1}$

`//current left orthogonalization and first right orthogonalization`

**SymOrtho**$(\widetilde{\gamma}_k^{(1)}, \widetilde{\beta}_{k+1}) \to \widetilde{c}_{k1}, \widetilde{s}_{k1}, \widetilde{\gamma}_k^{(2)}, \qquad$ **SymOrtho**$(\widetilde{\gamma}_{k-2}^{(5)}, \widetilde{\epsilon}_k^{(1)}) \to \widetilde{c}_{k2}, \widetilde{s}_{k2}, \widetilde{\gamma}_{k-2}^{(6)}$

$\widetilde{\delta}_k^{(3)} = \widetilde{s}_{k2} \widetilde{\vartheta}_{k-1}^{(1)} - \widetilde{c}_{k2} \widetilde{\delta}_k^{(2)}, \qquad \widetilde{\gamma}_k^{(3)} = -\widetilde{c}_{k2} \widetilde{\gamma}_k^{(2)}, \qquad \widetilde{\eta}_k^{(1)} = \widetilde{s}_{k2} \widetilde{\gamma}_k^{(2)}$

$\widetilde{\vartheta}_{k-1}^{(2)} = \widetilde{c}_{k2} \widetilde{\vartheta}_{k-1}^{(1)} + \widetilde{s}_{k2} \widetilde{\delta}_k^{(2)}$

`//second right orthogonalization to zero out` $\widetilde{\delta}_k^{(3)}$

**SymOrtho**$(\widetilde{\gamma}_{k-1}^{(4)}, \widetilde{\delta}_k^{(3)}) \to \widetilde{c}_{k3}, \widetilde{s}_{k3}, \widetilde{\gamma}_{k-1}^{(5)}, \qquad \widetilde{\vartheta}_k^{(1)} = \widetilde{s}_{k3} \widetilde{\gamma}_k^{(3)}, \qquad \widetilde{\gamma}_k^{(4)} = -\widetilde{c}_{k3} \widetilde{\gamma}_k^{(3)}$

`//update rhs, residual norms, matrix norms and condition no.,` $\|Ax_k\|$

$\widetilde{\tau}_k = \widetilde{c}_{k1} \widetilde{\phi}_{k-1}, \qquad \widetilde{\phi}_k = \widetilde{s}_{k1} \widetilde{\phi}_{k-1}, \qquad \widetilde{\psi}_{k-1} = \widetilde{\phi}_{k-1} \sqrt{(\widetilde{\gamma}_k^{(1)})^2 + (\widetilde{\delta}_{k+1}^{(1)})^2}$

**if** $k = 1, \qquad \widetilde{\gamma}_{\min} = \widetilde{\gamma}_1$ **else** $\widetilde{\gamma}_{\min} \leftarrow \min\{\widetilde{\gamma}_{\min}, \widetilde{\gamma}_{k-2}^{(6)}, \widetilde{\gamma}_{k-1}^{(5)}, |\widetilde{\gamma}_k^{(4)}|\}$ **end**

$\widetilde{\mathcal{A}}_2^{(k)} = \max\{\widetilde{\mathcal{A}}_2^{(k-1)}, \widetilde{\rho}_k, \widetilde{\gamma}_{k-2}^{(6)}, \widetilde{\gamma}_{k-1}^{(5)}, |\widetilde{\gamma}_k^{(4)}|\}, \qquad \widetilde{\kappa} \leftarrow \widetilde{\mathcal{A}}_2^{(k)} / \widetilde{\gamma}_{\min}, \qquad \widetilde{\omega}_k = \sqrt{\widetilde{\omega}_k^2 + \widetilde{\tau}_k^2}$

`//update` $w_k$, $x_k$ `and solution norm`

$w_k^{(1)} = -(\widetilde{c}_{k2} / \widetilde{\beta}_k) q_k + \widetilde{s}_{k2} w_{k-2}^{(3)}, \qquad w_{k-2}^{(4)} = (\widetilde{s}_{k2} / \widetilde{\beta}_k) q_k + \widetilde{c}_{k2} w_{k-2}^{(3)}$

**if** $k > 2$,

$w_k^{(2)} = \widetilde{s}_{k3} w_{k-1}^{(2)} - \widetilde{c}_{k3} w_k^{(1)}, \qquad w_{k-1}^{(3)} = \widetilde{c}_{k3} w_{k-1}^{(2)} + \widetilde{s}_{k3} w_k^{(1)}$

$\widetilde{\mu}_{k-2}^{(3)} = (\widetilde{\tau}_{k-2} - \widetilde{\mu}_{k-3}^{(3)} \widetilde{\vartheta}_{k-2}^{(1)}) / \widetilde{\gamma}_{k-2}^{(6)}$

**end**

**if** $k > 1, \qquad \widetilde{\mu}_{k-1}^{(2)} = (\widetilde{\tau}_{k-1} - \widetilde{\eta}_{k-1} \widetilde{\mu}_{k-3}^{(3)} - \widetilde{\vartheta}_{k-1}^{(2)} \widetilde{\mu}_{k-2}^{(3)}) / \widetilde{\gamma}_{k-1}^{(5)}$ **end**

**if** $\widetilde{\gamma}_k^{(2)} \neq 0, \qquad \widetilde{\mu}_k^{(1)} = (\widetilde{\tau}_k - \widetilde{\eta}_k^{(1)} \widetilde{\mu}_{k-2}^{(3)} - \widetilde{\vartheta}_k^{(1)} \widetilde{\mu}_{k-1}^{(2)}) / \widetilde{\gamma}_k^{(4)}$ **else** $\widetilde{\mu}_k^{(1)} = 0$ **end**

$x_{k-2} = x_{k-3} + \widetilde{\mu}_{k-2}^{(3)} w_{k-2}^{(3)}, \qquad \widetilde{\chi}_{k-2} = \sqrt{(\widetilde{\chi}_{k-3})^2 + (\widetilde{\mu}_{k-2}^{(3)})^2}$

$x_k = x_{k-2} + \widetilde{\mu}_{k-1}^{(2)} w_{k-1}^{(3)} + \widetilde{\mu}_k^{(1)} w_k^{(2)}, \qquad \widetilde{\chi}_k = \sqrt{(\widetilde{\chi}_{k-2})^2 + (\widetilde{\mu}_{k-1}^{(2)})^2 + (\widetilde{\mu}_k^{(1)})^2}$

$k \leftarrow k + 1$

**end**

$x = x_k, \qquad \widetilde{\phi} = \widetilde{\phi}_k, \qquad \widetilde{\psi} = \widetilde{\phi}_k \sqrt{(\widetilde{\gamma}_{k+1}^{(1)})^2 + (\widetilde{\delta}_{k+2}^{(1)})^2}, \qquad \widetilde{\chi} = \widetilde{\chi}_k, \qquad \widetilde{\mathcal{A}} = \widetilde{\mathcal{A}}_2^{(k)}, \qquad \widetilde{\omega} = \widetilde{\omega}_k$

To summarize, if $A$ and $C$ are nonsingular and the eigenvalues of $C^{-1}AC^{-T}$ are more clustered than those of $A$, and if systems $Mq = b$ with $M = CC^T$ are easy to solve, then we could expect preconditioned CG-type methods to converge to $x^\dagger$ more efficiently than without preconditioning.

The requirement of positive-definite preconditioners $M$ in PMINRES and PMINRES-QLP may seem unnatural for a problem with indefinite $A$ since we cannot achieve $M^{-\frac{1}{2}}AM^{-\frac{1}{2}} \approx I$. However, as shown in [40], we can achieve $M^{-\frac{1}{2}}AM^{-\frac{1}{2}} \approx \left[\begin{smallmatrix} I & \\ & -I \end{smallmatrix}\right]$ using an approximate block-LDL$^{\mathrm{T}}$ factorization $A \approx LDL^T$ to get $M = L|D|L^T$, where $D$ is indefinite with blocks of order 1 and 2, and $|D|$ has the same eigensystem as $D$ except negative eigenvalues are changed in sign.

Otherwise, SQMR [38] could work directly with an indefinite preconditioner (such as $LDL^T$).

### 3.4.2 Preconditioning Singular $Ax = b$

For singular compatible systems $Ax = b$, MINRES-QLP finds the minimum-length solution (see Theorem 3.1). If $M$ is nonsingular, the preconditioned system $\widetilde{A}\widetilde{x} = \widetilde{b}$ (3.22) is also compatible with minimum-length solution $\widetilde{x}$. The unpreconditioned solution $x = M^{-\frac{1}{2}}\widetilde{x}$ is a solution to $Ax = b$, but is not necessarily a minimum-length solution.

**Example 6.**

1. *If* $A = \left[\begin{smallmatrix} 2 & 1 & \\ 1 & 0 & \\ & & 1 \end{smallmatrix}\right]$, $b = \left[\begin{smallmatrix} 2 \\ 1 \\ 0 \end{smallmatrix}\right]$, $M = \left[\begin{smallmatrix} 1 & \sqrt{2} & \\ \sqrt{2} & 1 & \\ & & 1 \end{smallmatrix}\right]$, *then CG on* $Ax = b$ *converges in* 2 *iterations to* $x^\dagger = \left[\begin{smallmatrix} 1 \\ 1 \\ 0 \end{smallmatrix}\right]$. *But* $MAM = \left[\begin{smallmatrix} 2 & 2 & \\ 2 & 0 & \\ & & 1 \end{smallmatrix}\right]$ *and CG converges in* 1 *iteration to* $y = (MAM)^\dagger Mb = \left[\begin{smallmatrix} 1 \\ \frac{1}{\sqrt{2}} \\ 0 \end{smallmatrix}\right]$. *Then* $z = My = \left[\begin{smallmatrix} 1 & \sqrt{2} & \\ \sqrt{2} & 1 & \\ & & 1 \end{smallmatrix}\right] \left[\begin{smallmatrix} 1 \\ \frac{1}{\sqrt{2}} \\ 0 \end{smallmatrix}\right] = \left[\begin{smallmatrix} 1 \\ 1 \\ 0 \end{smallmatrix}\right] = x^\dagger$.

2. *Let* $B = \left[\begin{smallmatrix} 0.20146 & 0.71637 \\ 0.87843 & 0.74523 \\ 0.98696 & 0.94299 \\ 0.40047 & 0.21457 \end{smallmatrix}\right]$, $A = BB^T = \left[\begin{smallmatrix} 0.55377 & 0.71083 & 0.87437 & 0.23439 \\ 0.71083 & 1.3270 & 1.5697 & 0.51169 \\ 0.87437 & 1.5697 & 1.8633 & 0.59758 \\ 0.23439 & 0.51169 & 0.59758 & 0.20641 \end{smallmatrix}\right]$, *and* $b = \left[\begin{smallmatrix} 2.3734 \\ 4.1192 \\ 4.9050 \\ 1.5501 \end{smallmatrix}\right]$. *The matrix* $A$ *is of rank 2 and* $Ax = b$ *is a compatible system. The minimum-length solution is* $x^\dagger = \left[\begin{smallmatrix} 0.79657 & 1.0375 & 1.2740 & 0.34481 \end{smallmatrix}\right]^T$. *By binormalization (see section 3.5.2), we construct the diagonal matrix* $D = \mathrm{diag}(\begin{smallmatrix} 1.0590 & 0.61899 & 0.52000 & 1.6378 \end{smallmatrix})$. *The minimum-length solution of the diagonally preconditioned problem* $DADy = Db$ *is* $y^\dagger = \left[\begin{smallmatrix} 1.2452 & 1.2851 & 1.2834 & 1.2872 \end{smallmatrix}\right]^T$. *It follows that* $x = Dy^\dagger = \left[\begin{smallmatrix} 1.3187 & 0.79543 & 0.66738 & 2.1081 \end{smallmatrix}\right]^T$ *is a solution of* $Ax = b$ *but* $x \neq x^\dagger$.

### 3.4.3 Preconditioning Singular $Ax \approx b$

We propose the following techniques for obtaining minimum-residual, but not necessarily minimum-length, solutions of singular incompatible problems. We assume $A = A^T$ throughout this section.

#### Augmented system

When $A$ is singular, the augmented system $\left[\begin{smallmatrix} I & A \\ A & \end{smallmatrix}\right] \left[\begin{smallmatrix} r \\ x \end{smallmatrix}\right] = \left[\begin{smallmatrix} b \\ 0 \end{smallmatrix}\right]$ is also singular. However, the system is always compatible, and thus preconditioning always gives us a solution $\left[\begin{smallmatrix} r \\ x \end{smallmatrix}\right]$. Note that $x$ is not necessarily the minimum-length solution $x^\dagger$ for the original problem $\min \|Ax - b\|$, meaning $x = x^\dagger + x_{\mathcal{N}}$ for some possibly nonzero $x_{\mathcal{N}} \in \mathcal{N}(A)$, but $r = b - Ax = b - Ax^\dagger$ is unique.

#### A Giant KKT System

The minimum-length least-squares problem $\min \|x\|_2^2$ subject to $\min \|Ax - b\|_2^2$ is equivalent to $\min \left[\begin{smallmatrix} r \\ x \end{smallmatrix}\right]^T \left[\begin{smallmatrix} 0 & 0 \\ 0 & I \end{smallmatrix}\right] \left[\begin{smallmatrix} r \\ x \end{smallmatrix}\right]$ subject to $\left[\begin{smallmatrix} I & A \\ A & 0 \end{smallmatrix}\right] \left[\begin{smallmatrix} r \\ x \end{smallmatrix}\right] = \left[\begin{smallmatrix} b \\ 0 \end{smallmatrix}\right]$, which is an equality-contrained quadratic program.

The corresponding KKT (Karush-Kuhn-Tucker) system [75, section 16.1] is both symmetric and compatible:

$$\begin{bmatrix} & & I & A \\ & \pm I & A & \\ I & A & & \\ A & & & \end{bmatrix} \begin{bmatrix} r \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b \\ 0 \end{bmatrix}. \tag{3.28}$$

Although this is still a singular system, the upper-left $3 \times 3$ block-submatrix is nonsingular and therefore $\begin{bmatrix} r \\ x \\ y \end{bmatrix}$ is unique and a preconditioner applied to the KKT system would give $x$ as the minimum-length solution of our original problem.

### Regularization

When the numerical rank of a given matrix is ill-determined, such as with ill-conditioned weighted least-squares problems $\min_x \|D^{\frac{1}{2}}Ax - b\|$ with $D$ diagonal positive definite but ill-conditioned [14], we may want to *regularize* the problem [30, 54]. The regularized least-squares problem is really different from the original problem; it minimizes $\min \|Ax - b\|_2^2 + \|\delta x\|_2^2$:

$$\min \left\| \begin{bmatrix} A \\ \delta I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2^2, \tag{3.29}$$

where $\delta > 0$ is a *small* parameter. The matrix $\begin{bmatrix} A \\ \delta I \end{bmatrix}$ is of full-rank and always better conditioned.

Alternatively, we could transform (3.29) into the following symmetric and compatible systems before applying preconditioning techniques:

**Normal equation:**

$$(A^TA + \delta^2 I)x = A^Tb. \tag{3.30}$$

**Augmented system:**

$$\begin{bmatrix} I & A \\ A^T & -\delta^2 I \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}. \tag{3.31}$$

**A two-layered problem:** If we eliminate $v$ from $\begin{bmatrix} I & A^TA \\ A^TA & -\delta^2 A^TA \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ A^Tb \end{bmatrix}$, we obtain (3.30). Thus $x$ from the solution of the following system (with $A = A^T$) is also a solution of our regularized problem (3.29):

$$\begin{bmatrix} I & A^2 \\ A^2 & -\delta^2 A^2 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ Ab \end{bmatrix}. \tag{3.32}$$

This is equivalent to the two-layered formulation (4.3) in Bobrovnikova and Vavasis [14] (with $A_1 = A$, $A_2 = D_1 = D_2 = I$, $b_1 = b$, $b_2 = 0$, $\delta_1 = 1$, $\delta_2 = \delta^2$). A key property is that $x \to x^\dagger$ as $\delta \to 0$.

**A KKT-like system:** If we define $y = -Av$ and $r = b - Ax - \delta^2 y$, then we can show (by

eliminating $r$ and $y$ from the following system) that $x$ from the solution of

$$\begin{bmatrix} & & I & A \\ & -I & A^T & \\ I & A & \delta^2 I & \\ A^T & & & \end{bmatrix} \begin{bmatrix} r \\ x \\ y \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b \\ 0 \end{bmatrix} \tag{3.33}$$

is also a solution of (3.32) and thus of (3.29). The upper-left $3 \times 3$ block-submatrix of (3.33) is nonsingular, and the correct limiting behavior occurs: $x \to x^\dagger$ as $\delta \to 0$. In fact, (3.33) reduces to (3.28).

## 3.5 General Preconditioners

Construction of preconditioners is very problem-dependent. If we do not know in advance much about the structure of matrix $A$, then we could only consider general methods such as diagonal preconditioning and incomplete Cholesky factorization. These methods require access to the nonzero elements of $A_{ij}$. (They are not applicable if $A$ exists only as an operator for returning the product $Ax$.)

For a comprehensive survey of preconditioning techniques, see Benzi [8].

### 3.5.1 Diagonal Preconditioning

If $A$ has entries that are very large and different in magnitude, we can perform diagonal preconditioning to make the matrix better conditioned. Further, if $A$ is diagonally dominant and nonsingular, we can define $D = \text{diag}(d_1, \ldots, d_n)$, where

$$d_j = 1/\sqrt{|a_{jj}|}, \qquad j = 1, \ldots, n. \tag{3.34}$$

Instead of solving $Ax = b$, we solve $DADy = Db$, where $DAD$ is still diagonally dominant and nonsingular with all entries $\leq 1$ in magnitude, and $x = Dy$.

More generally, if $A$ is not diagonally dominant and possibly singular, we can safeguard division-by-zero errors by defining

$$d_j(\delta) = 1/\max\{\delta, \sqrt{|a_{jj}|}, \max_{i \neq j} |a_{i,j}|\}, \qquad j = 1, \ldots, n \tag{3.35}$$

for some parameter $\delta > 0$.

**Example 7.**

1. *If* $A = \begin{bmatrix} -1 & 10^{-8} & & \\ 10^{-8} & 1 & 10^4 & \\ & 10^4 & 0 & \\ & & & 0 \end{bmatrix}$, *then* $\kappa_2(A) \approx 10^4$. *Let* $\delta = 1$, $D = \text{diag}(1, 10^{-2}, 10^{-2}, 1)$ *in (3.35).* *Then* $DAD = \begin{bmatrix} -1 & 10^{-10} & & \\ 10^{-10} & 10^{-4} & 1 & \\ & 1 & 0 & \\ & & & 0 \end{bmatrix}$ *and* $\kappa_2(DAD) \approx 1$.
2. $A = \begin{bmatrix} 10^{-4} & 10^{-8} & & \\ 10^{-8} & 10^{-4} & 10^{-8} & \\ & 10^{-8} & 0 & \\ & & & 0 \end{bmatrix}$ *contains mostly very small entries, and* $\kappa_2(A) \approx 10^{10}$. *Let* $\delta = 10^{-8}$

*and* $D = \operatorname{diag}(10^2, 10^2, 10^8, 10^8)$. *Then* $DAD = \begin{bmatrix} 1 & 10^{-4} & & \\ 10^{-4} & 1 & 10^2 & \\ & 10^2 & 0 & \\ & & & 0 \end{bmatrix}$ *and* $\kappa_2(DAD) \approx 10^2$. *(The choice of* $\delta$ *makes a critical difference in this case: with* $\delta = 1$, *we have* $D = I$.*)*

### 3.5.2 Binormalization (BIN)

Livne and Golub [70] scale a symmetric matrix by a series of $k$ diagonal matrices on both sides until all rows and columns of the scaled matrix have unit 2-norm:

$$DAD = D_k \cdots D_1 A D_1 \cdots D_k. \tag{3.36}$$

**Example 8.** *If* $A = \begin{bmatrix} 10^{-8} & 1 & \\ 1 & 10^{-8} & 10^4 \\ & 10^4 & 0 \end{bmatrix}$, *then* $\kappa_2(A) \approx 10^{12}$. *With just one sweep of BIN, we obtain* $D = \operatorname{diag}(8.1 \times 10^{-3}, 6.6 \times 10^{-5}, 1.5)$, $DAD \approx \begin{bmatrix} 6.5\times10^{-1} & 5.3\times10^{-1} & 0 \\ 5.3\times10^{-1} & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ *and* $\kappa_2(DAD) \approx 2.6$ *even though the rows and columns have not converged to one in the two-norm. In contrast, diagonal scaling (3.35) defined by* $\delta = 1$ *and* $D = \operatorname{diag}(1, 10^{-4}, 10^{-4})$ *reduces the condition number to approximately* $10^4$.

### 3.5.3 Incomplete Cholesky Factorization

For a sparse symmetric positive definite matrix $A$, we could compute a preconditioner by the incomplete Cholesky factorization that preserves the sparsity pattern of $A$. This is known as *IC0* in the literature. Sometimes there exists a permutation $P$ such that the IC0 factor of $PAP^T$ is more sparse than that of $A$.

When $A$ is semidefinite or indefinite, IC0 may not exist, but a simple variant that may work is incomplete Cholesky-infinity factorization [116, section 5].

# Chapter 4

# Numerical Experiments on Symmetric Systems

We compare the computational results of MINRES-QLP and various other Krylov subspace methods to solutions computed directly by the eigenvalue decomposition (EVD) and the truncated eigenvalue decompositions (TEVD) of $A$. Let $A = U\Lambda U^T$, where $\Lambda := \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Then we have

$$
\begin{aligned}
x^{EVD} &:= \sum_{|\lambda_i|>0} \frac{1}{\lambda_i} u_i u_i^T b, \\
x^{TEVD} &:= \sum_{|\lambda_i|>c\|A\|_2\varepsilon} \frac{1}{\lambda_i} u_i u_i^T b, \text{ where } c \text{ is some positive real parameter,} \\
\|A\|_2^{EVD} &= \|A\|_2^{TEVD} = \max|\lambda_i|, \\
\kappa_2^{EVD}(A) &= \frac{\max|\lambda_i|}{\min\limits_{|\lambda_i|>0}|\lambda_i|}, \qquad \kappa_2^{TEVD}(A) = \frac{\max|\lambda_i|}{\min\limits_{|\lambda_i|>c\|A\|_2\varepsilon}|\lambda_i|}.
\end{aligned}
$$

We note that $c$ in TEVD is often set to 1 and sometimes set to a moderate positive number such as 10 or 20; it helps to define a "cut-off" point relative to the largest eigenvalue of $A$. For example, with matrix ID 1239 (section 4.1) we noticed that all eigenvalues are between 0.1 and 5 in magnitude except for two: $-3.72 \times 10^{-15}$ and $-1.68 \times 10^{-15}$, just slightly bigger than $\|A\|_2 \varepsilon \approx 10^{-16}$. We expect TEVD to work better when the two small eigenvalues are excluded.

Table 4.1 reviews the key features of software implementations available as MATLAB files. Note that MATLAB MINRES and MATLAB SYMMLQ are MATLAB's implementation of MINRES and SYMMLQ repectively. We also reset `iteration` of MATLAB SYMMLQ to the length of its `resvec` output. Lacking the correct stopping condition for singular problems, MATLAB SYMMLQ works more than necessary and then selects the smallest residual norm from all computed iterates; it would sometimes report that the method did not converge while the selected estimate appeared to be reasonably accurate.

MATLAB's implementation of MINRES and SYMMLQ incorporate *local reorthogonalization* of the Lanczos vector $v_2$, which could enhance the accuracy of the computations if $b$ is close to an eigenvector of $A$ [69]:

$$
\begin{aligned}
&\text{Second Lanczos step: } \beta_1 v_1 = b, \text{ and } q_2 := \beta_2 v_2 = Av_1 - \alpha_1 v_1, \qquad (4.1)\\
&\text{Initial local reorthogonalization: } q_2 \leftarrow q_2 - (v_1^T q_2) v_1.
\end{aligned}
$$

The computations in this chapter were performed on a Windows XP machine with a 3.2GHz Intel Pentium D Processor 940 and 3GB RAM.

TABLE 4.1
*Different* MATLAB *implementations of various Krylov subspace methods from SOL [97] and The MathWorks.*

| MATLAB filename | Algorithm | Stopping Conditions ($\beta_1 := \|b\| = \|r_0\|$) |
|---|---|---|
| EVD | Eigenvalue decomposition | |
| TEVD | Truncated eigenvalue decomposition | |
| MATLAB 7 `PCG` | CG in Table 2.7 | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| MATLAB 7 `PCGI` | CGI in Table 2.8 | $\frac{\|r_k\|}{\mathcal{A}\|x_k\|+\beta_1} < \text{tol}$, $\mathcal{A} = \texttt{NORMEST}(A)$ |
| MATLAB 7 `SYMMLQ` | SYMMLQ in Table 2.10 | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| SOL `SYMMLQ` | SYMMLQ in Table 2.10 | $\frac{\phi_k}{\mathcal{A}_F^{(k)}\chi_k+\beta_1} < \text{tol}$ |
| SOL `SYMMLQ3` | SYMMLQ in Table 2.10 with stabilized reflectors in Table 2.9 | $\frac{\phi_k}{\mathcal{A}_2^{(k)}\chi_k+\beta_1} < \text{tol}$ |
| MATLAB 7 `MINRES` | MINRES in Table 2.11 | $\frac{\phi_k}{\beta_1} < \text{tol}$ |
| SOL `MINRES` | MINRES in Table 2.11 | $\frac{\phi_k}{\mathcal{A}_F^{(k)}\|x_k\|+\beta_1} < \text{tol}$ |
| SOL `MINRES69` | MINRES in Table 2.11 with stabilized reflectors in Table 2.9 | $\frac{\phi_k}{\mathcal{A}_2^{(k)}\|x_k\|+\beta_1} < \text{tol}$ $\frac{\psi_k}{\mathcal{A}_2^{(k)}\phi_k} < \text{tol}$ |
| `MINRES-QLP43` | MINRES-QLP in Table 3.1 with stabilized reflectors in Table 2.9 | $\frac{\phi_k}{\mathcal{A}_2^{(k)}\|x_k\|+\beta_1} < \text{tol}$ $\frac{\psi_k}{\mathcal{A}_2^{(k)}\phi_k} < \text{tol}$ |
| MATLAB `GMRES`(10) | GMRES in Table 2.16 with restart 10 | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| MATLAB 7 `LSQR` | LSQR in Table 2.18 | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| SOL `LSQR` | LSQR in Table 2.18 with right-orthogonalization to recur $\|x_k\|$ | $\frac{\phi_k}{\mathcal{A}_F^{(k)}\|x_k\|+\beta_1} < \text{tol}$ |
| MATLAB 7 `BiCG` | BiCG | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| MATLAB 7 `BiCGSTAB` | BiCGSTAB | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| MATLAB 7 `QMR` | QMR | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |
| `SQMR` | SQMR | $\frac{\|r_k\|}{\beta_1} < \text{tol}$ |

Tests were performed with each solver on four types of problem:

1. symmetric linear systems,
2. mildly incompatible symmetric systems (meaning $\|r\|$ is rather small with respect to $\|b\|$),
3. symmetric (and singular) least-squares problems, and
4. compatible Hermitian systems.

For a compatible system, we generate a random right-hand side vector $b$ that is in the range of the test matrix ($b := Ay$, $y_i \sim U(0,1)$). For a least-squares problem, we generate a random right-hand side vector $b$ that is *not* in the range of the test matrix ($b_i \sim U(0,1)$ often suffices).

We could say from the results that the Lanczos-based methods have built-in regularization features [62], often matching the TEVD solutions very well.

## 4.1 A Singular Indefinite System

In this example, $A$ is indefinite and singular of order $n = 3002$; it is available from University of Floria sparse matrix collection [108] (matrix ID 1239, contributed by Gould, Hu, & Scott). We set $b = Ae$, where $e$ is the vector of all 1's giving a compatible system. MATLAB PCG (without preconditioner) terminates early because of the indefiniteness, but PCGI, which is MATLAB PCG with stopping conditions changed as we defined in CGI, works competitively. All variations of SYMMLQ and MINRES converge to the TEVD solution, although they stop at different iterations due to different stopping conditions: MATLAB MINRES and SYMMLQ use $\frac{r_k}{\beta_1} \leq \text{tol}$ and run more iterations than the other solvers; PCGI and MINRES-SOL69 use $\frac{\|r_k\|}{\mathcal{A}_2\|x_k\|+\beta_1} \leq \text{tol}$, where $\mathcal{A}_2$ estimates $\|A\|_2$; MINRES-SOL uses $\frac{\|r_k\|}{\mathcal{A}_F\|x_k\|+\beta_1} \leq \text{tol}$, where $\mathcal{A}_F$ estimates $\|A\|_F$.

EVD (or TEVD) took about 6 minutes to produce the solution, while all iterative methods together took less than 1 second.

We plot residual and solution norms in Figure 4.1 at each iteration from the most competitive solvers PCGI, MINRES-SOL69, and SYMMLQ-SOL3. MINRES-SOL69 and MINRES-QLP43 achieve the minimum-residual norm (by design) and also the smallest solution norm over most iterations. We observe a spike in PCGI's solution norms and more ups and downs in PCGI and SYMMLQ's residual norms, although they do not prevent the methods from converging accurately in this case. To reproduce this example, run `testminresQLP27(9)`.

```
Matrix ID = 1239, title  = Gould, Hu, & Scott:
n     =  3002,   maxit =   500,   nnz    =  9000,   nnz / n = 3,   numerical rank = 3000
shift = 0.00e+00,  tol = 1e-09, maxxnorm = 1e+09,  maxcond  = 1e+14, ||b|| = 1.0e+02.
No. of positive eigenvalue(s)    = 2000: between  2.26e-01 and  4.25e+00.
No. of almost zero eigenvalue(s) =    2: between -3.53e-15 and  3.43e-15.
No. of negative eigenvalue(s)    = 1000: between -1.10e+00 and -4.64e-01.

EVD.               || AV - VD ||_2 / ||A||_2 =  1.60e-14.  || V'V - I ||_2 =  4.45e-14
TEVD.              || AV - VD ||_2 / ||A||_2 =  1.60e-14.  || V'V - I ||_2 =  4.45e-14
Matlab 7 PCG.      Some scalar quantities became too small or too large.
Matlab 7 PCGI.     Converged to TOL within MAXIT iterations.
```

```
Matlab 7 SYMMLQ.    Converged to TOL within MAXIT iterations.
SYMMLQ SOL.         Reasonable accuracy achieved, given eps.
SYMMLQ SOL3.        Reasonable accuracy achieved, given eps.
Matlab 7 MINRES.    Converged to TOL within MAXIT iterations.
MINRES SOL.         A solution to Ax = b was found, given rtol.
MINRES SOL69.       A solution to (poss. singular) Ax = b found, given rtol.
MINRES QLP43.       A solution to (poss. singular) Ax = b found, given rtol.
                    trancond = 1e+007, Mitn = 54, Qitn = 0.
Matlab 7 LSQR.      Converged to TOL within MAXIT iterations.
LSQR SOL.           Ax - b is small enough, given atol, btol.
Matlab 7 GMRES(10). Converged to TOL within MAXIT iterations.
SQMR.               ||r_k|| < TOL within MAXIT iterations.
Matlab 7 QMR.       Converged to TOL within MAXIT iterations.
Matlab 7 BICG.      Converged to TOL within MAXIT iterations
Matlab 7 BICGSTAB.  Converged to TOL within MAXIT iterations.
```

| Method | A*v | x(1) | ‖x‖ direct | ‖e‖ =‖x−xTEVD‖ | ‖r‖ direct | ‖Ar‖ direct | ‖A‖ | K(A) |
|---|---|---|---|---|---|---|---|---|
| EVD. | -- | -4.257e+00 | 5.498e+01 | 4.7e+00 | 1.9e-12 | 7.4e-12 | 4.2e+00 | 1.2e+15 |
| TEVD. | -- | 2.679e-01 | 5.478e+01 | 0.0e+00 | 1.9e-12 | 7.4e-12 | 4.2e+00 | 1.9e+01 |
| Matlab 7 PCG. | 2 | 2.714e-01 | 6.303e+01 | 2.3e+01 | 2.4e+01 | 2.7e+01 | -- | -- |
| Matlab 7 PCGI. | 39 | 2.679e-01 | 5.478e+01 | 3.6e-05 | 1.7e-05 | 2.9e-05 | -- | -- |
| Matlab 7 SYMMLQ. | 59 | 2.679e-01 | 5.478e+01 | 1.3e-07 | 9.4e-08 | 2.1e-07 | -- | -- |
| SYMMLQ SOL. | 32 | 2.679e-01 | 5.478e+01 | 2.2e-04 | 1.1e-04 | 2.0e-04 | 1.3e+01 | 6.7e+00 |
| SYMMLQ SOL3. | 41 | 2.679e-01 | 5.478e+01 | 1.5e-05 | 1.6e-05 | 5.3e-05 | 3.9e+00 | 6.7e+00 |
| Matlab 7 MINRES. | 57 | 2.679e-01 | 5.478e+01 | 2.6e-07 | 8.9e-08 | 5.7e-08 | -- | -- |
| MINRES SOL. | 46 | 2.679e-01 | 5.478e+01 | 5.1e-06 | 2.0e-06 | 4.7e-06 | 1.0e+02 | 6.7e+00 |
| MINRES SOL69. | 54 | 2.679e-01 | 5.478e+01 | 5.2e-07 | 2.1e-07 | 2.5e-07 | 3.9e+00 | 4.6e+01 |
| MINRES QLP43. | 54 | 2.679e-01 | 5.478e+01 | 5.2e-07 | 2.1e-07 | 2.5e-07 | 4.2e+00 | 9.6e+00 |
| Matlab 7 LSQR. | 118 | 2.679e-01 | 5.478e+01 | 2.2e-07 | 6.9e-08 | 4.4e-08 | -- | -- |
| LSQR SOL. | 98 | 2.679e-01 | 5.478e+01 | 3.1e-06 | 9.9e-07 | 1.6e-06 | 1.9e+01 | 2.0e+02 |
| Matlab 7 GMRES(10). | 76 | 2.679e-01 | 5.478e+01 | 2.8e-07 | 9.2e-08 | 5.4e-08 | -- | -- |
| SQMR. | 55 | 2.679e-01 | 5.478e+01 | 4.3e-07 | 2.3e-07 | 2.7e-07 | -- | -- |
| Matlab 7 QMR. | 116 | 2.679e-01 | 5.478e+01 | 2.6e-07 | 8.9e-08 | 5.7e-08 | -- | -- |
| Matlab 7 BICG. | 120 | 2.679e-01 | 5.478e+01 | 1.3e-07 | 9.4e-08 | 2.1e-07 | -- | -- |
| Matlab 7 BICGSTAB. | 252 | 2.679e-01 | 5.478e+01 | 1.6e-07 | 9.9e-08 | 8.6e-08 | -- | -- |

## 4.2 Two Laplacian Systems

### 4.2.1 An Almost Compatible System

Our first example is an $n = 400$ symmetric singular indefinite linear system with $A$ being a Laplacian matrix, i.e. a symmetric block-tridiagonal matrix with each block equal to a tridiagonal

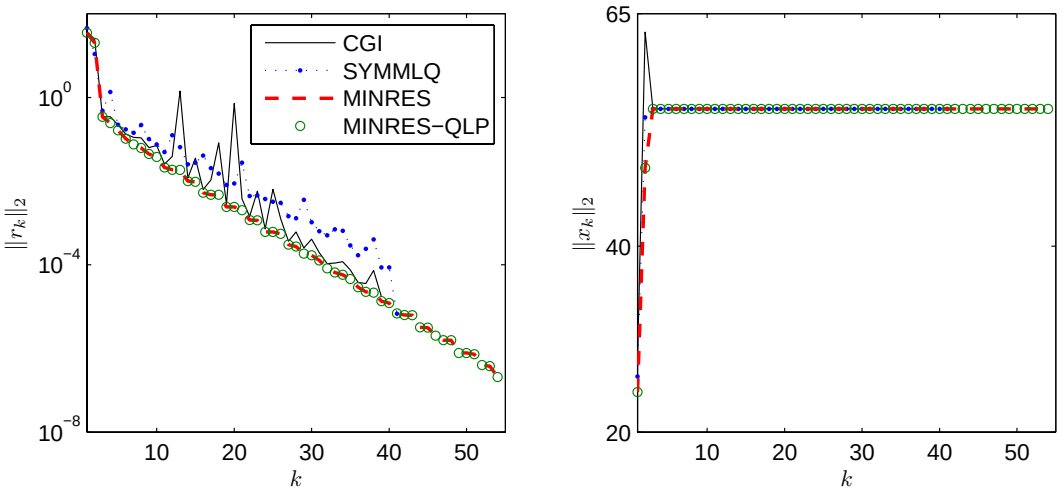


FIGURE 4.1 *Indefinite and singular $Ax = b$, with $A$ from University of Florida sparse matrix collection (Matrix ID 1239) and $b = Ae$. For details, see section 4.1. To reproduce this figure, run* `testminresQLP27(9)`.

matrix $T$ of order $N = 20$ with all nonzeros equal to 1:

$$A = \begin{bmatrix} T & T & & \\ T & T & \ddots & \\ & \ddots & \ddots & T \\ & & T & T \end{bmatrix}_{n \times n}, \quad T = \begin{bmatrix} 1 & 1 & & \\ 1 & 1 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 1 \end{bmatrix}_{N \times N}.$$

The right-hand side $b = Ay + 10^{-10}z$ (with $y_i$ and $z_i \sim U(0,1)$) has a very small incompatible component. MINRES-SOL gives a larger solution than MINRES-QLP. This example has a residual norm of about $1.7 \times 10^{-10}$, so it is not clear whether to classify it as a linear system or a least-squares problem. To the credit of PCGI and MATLAB SYMMLQ, they think it is a linear system and return good solutions. LSQR converges but with more than twice the number of iterations of MINRES-QLP. The other solvers fall short in some way. To reproduce this example, run `testminresQLP27(24)`.

The termination message for MINRES-QLP shows that the first 424 iterations were in standard "MINRES mode", with a transfer to "MINRES-QLP mode" for the last 278 iterations.

```
Title  = FINITE ELEMENT PROBLEM. LAPLACIAN ON A 20 BY 20 GRID.
n     =   400,   maxit =  1200,   nnz    =  3364,   nnz / n = 9,   numerical rank = 361
shift = 0.00e+00,  tol = 1e-15, maxxnorm = 1e+02,  maxcond  = 1e+15, ||b|| = 8.7e+01.
No. of positive eigenvalue(s)    =  205: between  6.10e-02 and  8.87e+00.
No. of almost zero eigenvalue(s) =   39: between -2.36e-15 and  2.65e-15.
No. of negative eigenvalue(s)    =  156: between -2.91e+00 and -6.65e-02.

EVD.               || AV - VD ||_2 / ||A||_2 =  3.84e-15.   || V'V - I ||_2 =  1.12e-14
TEVD.              || AV - VD ||_2 / ||A||_2 =  3.81e-15.   || V'V - I ||_2 =  1.12e-14
Matlab 7 PCG.      Some scalar quantities became too small or too large.
```

```
Matlab 7 PCGI.      Iterated MAXIT times but did not converge.
Matlab 7 SYMMLQ.    Iterated MAXIT times but did not converge.
SYMMLQ SOL.         xnorm has exceeded maxxnorm.
SYMMLQ SOL3.        xnorm has exceeded maxxnorm.
Matlab 7 MINRES.    Iterated MAXIT times but did not converge.
MINRES SOL.         The iteration limit was reached.
MINRES SOL69.       xnorm has exceeded maxxnorm.
MINRES QLP43.       xnorm has exceeded maxxnorm. trancond = 1e+007, Mitn = 424, Qitn = 278.
Matlab 7 LSQR.      Converged to TOL within MAXIT iterations.
LSQR SOL.           The least-squares solution is good enough, given atol.
Matlab 7 GMRES(10). Iterated MAXIT times but did not converge.
SQMR.               q^T A q = 0.
Matlab 7 QMR.       Iterated MAXIT times but did not converge.
Matlab 7 BICG.      Iterated MAXIT times but did not converge.
Matlab 7 BICGSTAB.  Iterated MAXIT times but did not converge.
```

| Method | A*v | x(1) | ‖x‖ direct | ‖e‖ =‖x−xTEVD‖ | ‖r‖ direct | ‖Ar‖ direct | ‖A‖ | K(A) |
|---|---|---|---|---|---|---|---|---|
| EVD. | -- | -2.986e+04 | 5.300e+05 | 5.3e+05 | 2.3e-09 | 9.8e-09 | 8.9e+00 | 2.9e+17 |
| TEVD. | -- | 3.892e-01 | 1.147e+01 | 0.0e+00 | 1.7e-10 | 2.7e-12 | 8.9e+00 | 1.5e+02 |
| Matlab 7 PCG. | 2 | 1.974e-01 | 1.025e+01 | 5.5e+00 | 1.2e+01 | 5.1e+01 | -- | -- |
| Matlab 7 PCGI. | 1200 | 3.892e-01 | 1.147e+01 | 4.8e-09 | 3.1e-10 | 1.8e-09 | -- | -- |
| Matlab 7 SYMMLQ. | 1200 | 3.892e-01 | 1.147e+01 | 9.5e-10 | 1.9e-09 | 6.0e-09 | -- | -- |
| SYMMLQ SOL. | 652 | 1.049e+01 | 9.511e+01 | 9.5e+01 | 1.3e+02 | 9.7e+02 | 1.2e+02 | 1.2e+01 |
| SYMMLQ SOL3. | 652 | -3.277e+03 | 4.753e+05 | 4.8e+05 | 2.2e+02 | 1.6e+03 | 8.6e+00 | 1.2e+01 |
| Matlab 7 MINRES. | 1200 | 5.577e+02 | 1.071e+05 | 1.1e+05 | 1.4e+03 | 8.4e+03 | -- | -- |
| MINRES SOL. | 1200 | 5.965e+03 | 4.105e+05 | 4.1e+05 | 3.8e+04 | 2.7e+05 | 1.8e+02 | 1.8e+01 |
| MINRES SOL69. | 701 | -9.515e-02 | 7.121e+01 | 7.0e+01 | 1.7e-10 | 3.7e-12 | 8.6e+00 | 2.8e+14 |
| MINRES QLP43. | 702 | 3.892e-01 | 1.147e+01 | 4.5e-12 | 1.7e-10 | 9.4e-12 | 8.7e+00 | 2.1e+14 |
| Matlab 7 LSQR. | 1762 | 3.892e-01 | 1.147e+01 | 2.6e-13 | 1.7e-10 | 2.1e-13 | -- | -- |
| LSQR SOL. | 1758 | 3.892e-01 | 1.147e+01 | 2.7e-13 | 1.7e-10 | 2.8e-13 | 1.6e+02 | 8.0e+03 |
| Matlab 7 GMRES(10). | 1200 | 3.915e-01 | 1.145e+01 | 4.0e-01 | 2.6e-02 | 5.7e-03 | -- | -- |
| SQMR. | 399 | 3.892e-01 | 1.147e+01 | 2.4e-08 | 2.1e-08 | 4.4e-08 | -- | -- |
| Matlab 7 QMR. | 2400 | 3.892e-01 | 1.147e+01 | 4.6e-09 | 1.7e-10 | 2.0e-11 | -- | -- |
| Matlab 7 BICG. | 2400 | 3.892e-01 | 1.147e+01 | 4.8e-09 | 3.1e-10 | 1.8e-09 | -- | -- |
| Matlab 7 BICGSTAB. | 4800 | 3.892e-01 | 1.147e+01 | 5.9e-09 | 1.7e-10 | 4.9e-13 | -- | -- |

### 4.2.2 A Least-Squares Problem

This example is a clear-cut least-squares problem with $A$ again the Laplace matrix in the last example, while $b = 10\times\text{rand}(n, 1)$. The residual norm is about 16. MINRES gives a least-squares solution. MINRES-QLP is the only solver that matches the solution of TEVD. All the other solvers are not satisfactory in performance. To reproduce this example, run `testminresQLP27(25)`.

```
Title  = FINITE ELEMENT PROBLEM. LAPLACIAN ON A 20 BY 20 GRID.
n     =   400,   maxit =   500,   nnz    =  3364,   nnz / n = 9,   numerical rank = 361
shift = 0.00e+00,  tol = 1e-14, maxxnorm = 1e+04,  maxcond  = 1e+14, ||b|| = 1.2e+02.
No. of positive eigenvalue(s)    =  205: between  6.10e-02 and  8.87e+00.
No. of almost zero eigenvalue(s) =   39: between -2.36e-15 and  2.65e-15.
No. of negative eigenvalue(s)    =  156: between -2.91e+00 and -6.65e-02.

EVD.               || AV - VD ||_2 / ||A||_2 =  3.84e-15.   || V'V - I ||_2 =  1.12e-14
TEVD.              || AV - VD ||_2 / ||A||_2 =  3.81e-15.   || V'V - I ||_2 =  1.12e-14
Matlab 7 PCG.      Some scalar quantities became too small or too large.
Matlab 7 PCGI.     Iterated MAXIT times but did not converge.
Matlab 7 SYMMLQ.   Iterated MAXIT times but did not converge.
SYMMLQ SOL.        xnorm has exceeded maxxnorm.
SYMMLQ SOL3.       xnorm has exceeded maxxnorm.
Matlab 7 MINRES.   Iterated MAXIT times but did not converge.
MINRES SOL.        The iteration limit was reached.
MINRES SOL69.      xnorm has exceeded maxxnorm.
MINRES QLP43.      xnorm has exceeded maxxnorm.
                   trancond = 1e+007, Mitn = 347, Qitn = 36.
Matlab 7 LSQR.     Iterated MAXIT times but did not converge.
LSQR SOL.          The iteration limit has been reached.
Matlab 7 GMRES(10). Iterated MAXIT times but did not converge.
SQMR.              Iterated MAXIT times but did not converge.
Matlab 7 QMR.      Iterated MAXIT times but did not converge.
Matlab 7 BICG.     Iterated MAXIT times but did not converge.
Matlab 7 BICGSTAB. Iterated MAXIT times but did not converge.

Method             A*v  x(1)      ||x||     ||e||      ||r||   ||Ar||   ||A||  K(A)
                                   direct  =||x-xTEVD|| direct  direct
EVD.                -- -2.984e+15 5.300e+16 5.3e+16   2.3e+02 9.5e+02 8.9e+00 2.9e+17
TEVD.               -- -8.750e+00 1.426e+02 0.0e+00   1.7e+01 4.8e-12 8.9e+00 1.5e+02
Matlab 7 PCG.        1  1.402e+00 1.708e+01 1.4e+02   6.0e+01 2.6e+02  --      --
Matlab 7 PCGI.     500  1.402e+00 1.708e+01 1.4e+02   6.0e+01 2.6e+02  --      --
Matlab 7 SYMMLQ.   500  2.735e-01 1.517e+01 1.4e+02   6.0e+01 2.9e+02  --      --
SYMMLQ SOL.        228 -6.961e+02 9.667e+03 9.7e+03   1.1e+04 6.0e+04 6.8e+01 8.5e+00
SYMMLQ SOL3.       228 -6.961e+02 9.667e+03 9.7e+03   1.1e+04 6.0e+04 7.6e+00 8.5e+00
Matlab 7 MINRES.   500 -1.981e+14 2.875e+16 2.9e+16   2.1e+02 1.2e+03  --      --
MINRES SOL.        500  2.493e+14 3.619e+16 3.6e+16   1.8e+02 9.8e+02 1.5e+02 1.1e+01
MINRES SOL69.      382 -1.447e+01 8.428e+02 8.3e+02   1.7e+01 1.0e-05 7.6e+00 3.0e+09
MINRES QLP43.      383 -8.750e+00 1.426e+02 4.4e-06   1.7e+01 1.1e-05 8.6e+00 1.2e+10
Matlab 7 LSQR.    1000 -8.750e+00 1.426e+02 3.0e-05   1.7e+01 2.3e-05  --      --
LSQR SOL.         1000 -8.750e+00 1.426e+02 3.3e-05   1.7e+01 9.6e-06 1.2e+02 4.4e+03
Matlab 7 GMRES(10). 500 -7.678e+00 1.088e+02 6.3e+01   1.7e+01 1.6e+00  --      --
SQMR.              500 -9.711e+15 1.409e+18 1.4e+18   7.2e+10 2.5e+11  --      --
Matlab 7 QMR.     1000 -7.300e+00 2.336e+02 1.9e+02   1.7e+01 3.6e+00  --      --
Matlab 7 BICG.    1000  1.402e+00 1.708e+01 1.4e+02   6.0e+01 2.6e+02  --      --
Matlab 7 BICGSTAB. 2000 -1.651e+01 2.368e+02 1.5e+02   2.7e+01 2.8e+01  --      --
```

## 4.3 Hermitian Problems

If $A$ is Hermitian, then $v^H Av$ is real for all complex vectors $v$. In this example, $A$ is the only Hermitian matrix found in Matrix Market as of April 2006, and $b = Az$ with $z_i \sim U(0, 1)$.

Numerically (in double precision), all $\alpha_k = v_k^H A v_k$ turn out to have small imaginary parts in the first few iterations and snowball to have large imaginary parts in later iterations. This would result in a poor estimation of $\|T_k\|_F$ or $\|A\|_F$, and unnecessary error in the Lanczos iteration. Thus we made sure to *typecast* $\alpha_k = \text{real}(v_k^H A v_k)$ in MINRES-QLP and MINRES-SOL.

The matrix is positive definite but not diagonally dominant. Some elements have magnitude of order $\varepsilon$; the other nonzeros are between $2.5 \times 10^{-10}$ and 53.2.

### 4.3.1 Without Preconditioning

The matrix condition number of $A$ is approximately $10^{11}$. All iterative solvers exhibit slow convergence; after $n$ iterations the solution estimates are all different from the TEVD solution. To reproduce this example, run `testminresQLP27(21)`.

```
Title  = Matrix Market, MHD1280B: Alfven Spectra in Magnetohydrodynamics
n     =  1280,   maxit =  1280,   nnz    = 22778,   nnz / n = 18,   numerical rank = 1280
shift = 0.00e+00,  tol = 1e-12, maxxnorm = 1e+09,  maxcond  = 1e+14, ||b|| = 9.3e+01.
No. of positive eigenvalue(s)    = 1280: between  1.48e-11 and  7.03e+01.
No. of almost zero eigenvalue(s) =    0
No. of negative eigenvalue(s)    =    0

EVD.               || AV - VD ||_2 / ||A||_2 =  7.45e-15.   || V'V - I ||_2 =  2.40e-14
TEVD.              || AV - VD ||_2 / ||A||_2 =  7.45e-15.   || V'V - I ||_2 =  2.40e-14
Matlab 7 PCG.      Iterated MAXIT times but did not converge.
Matlab 7 PCGI.     Iterated MAXIT times but did not converge.
Matlab 7 SYMMLQ.   Iterated MAXIT times but did not converge.
SYMMLQ SOL.        The iteration limit was reached.
SYMMLQ SOL3.       The iteration limit was reached.
Matlab 7 MINRES.   Iterated MAXIT times but did not converge.
MINRES SOL.        The iteration limit was reached.
MINRES SOL69.      The iteration limit was reached.
MINRES QLP45.      The iteration limit was reached.
                   trancond = 1e+007, Mitn = 1280, Qitn = 0.
Matlab 7 LSQR.     Iterated MAXIT times but did not converge.
LSQR SOL.          The iteration limit has been reached.
Matlab 7 GMRES(10). Iterated MAXIT times but did not converge.
SQMR.              Iterated MAXIT times but did not converge.
Matlab 7 QMR.      Iterated MAXIT times but did not converge.
Matlab 7 BICG.     Iterated MAXIT times but did not converge.
Matlab 7 BICGSTAB. Iterated MAXIT times but did not converge.

Method             A*v  x(1)      ||x||      ||e||     ||r||   ||Ar||  ||A||   K(A)
                                   direct  =||x-xTEVD|| direct  direct
EVD.                --  9.501e-01 2.069e+01  0.0e+00  7.1e-13 3.6e-11 7.0e+01 4.8e+12
```

```
TEVD.                --  9.501e-01 2.069e+01  0.0e+00  7.1e-13 3.6e-11 7.0e+01 4.8e+12
Matlab 7 PCG.      1280  9.501e-01 1.860e+01  8.7e+00  9.6e-05 1.2e-04  --      --
Matlab 7 PCGI.     1280  9.501e-01 1.860e+01  8.7e+00  9.6e-05 1.2e-04  --      --
Matlab 7 SYMMLQ.   1280  9.501e-01 1.848e+01  9.3e+00  2.7e-02 6.0e-02  --      --
SYMMLQ SOL.        1280  9.484e-01 1.854e+01  9.2e+00  2.2e-01 3.9e+00 1.1e+03 2.1e+02
SYMMLQ SOL3.       1280  9.501e-01 1.860e+01  8.7e+00  1.6e-04 4.2e-04 7.0e+01 2.1e+02
Matlab 7 MINRES.   1280  9.501e-01 1.855e+01  8.8e+00  1.0e-05 9.7e-07  --      --
MINRES SOL.        1280  9.501e-01 1.855e+01  8.8e+00  1.0e-05 1.6e-06 1.0e+03 2.8e+02
MINRES SOL69.      1280  9.501e-01 1.854e+01  8.9e+00  1.2e-05 1.9e-06 7.0e+01 3.5e+06
MINRES QLP45.      1280  9.501e-01 1.854e+01  8.9e+00  1.2e-05 1.9e-06 7.0e+01 4.7e+05
Matlab 7 LSQR.     2560  9.501e-01 1.485e+01  1.3e+01  2.8e-02 6.5e-03  --      --
LSQR SOL.          2560  9.501e-01 1.484e+01  1.3e+01  2.8e-02 2.3e-03 1.3e+03 2.0e+05
Matlab 7 GMRES(10).1280  9.501e-01 1.773e+01  9.7e+00  2.1e-03 3.7e-03  --      --
SQMR.              1280  9.501e-01 1.861e+01  8.6e+00  1.2e-04 1.6e-03  --      --
Matlab 7 QMR.      2560  9.501e-01 1.853e+01  8.9e+00  2.6e-05 2.3e-04  --      --
Matlab 7 BICG.     2560  9.501e-01 1.860e+01  8.7e+00  1.1e-04 4.3e-04  --      --
Matlab 7 BICGSTAB. 5120  9.501e-01 1.852e+01  9.0e+00  1.7e-05 1.2e-06  --      --
```

### 4.3.2 With Diagonal Preconditioning

We applied diagonal preconditioning. The matrix condition number of $DAD$ is approximately $10^4$ with $\delta = 1$ in (3.35), the performance of all solvers improve. To reproduce this example, run `testminresQLP27(27)`.

```
Title  = Matrix Market, MHD1280B: Alfven Spectra in Magnetohydrodynamics
n     =  1280,   maxit =  2560,   nnz    = 22778,   nnz / n = 18,   numerical rank = 1280
shift = 0.00e+00,  tol = 1e-12, maxxnorm = 1e+09,  maxcond  = 1e+14, ||b|| = 2.7e+01.
No. of positive eigenvalue(s)    = 1280: between  3.28e-04 and  2.73e+00.
No. of almost zero eigenvalue(s) =    0
No. of negative eigenvalue(s)    =    0

EVD.               || AV - VD ||_2 / ||A||_2 =  7.46e-15.   || V'V - I ||_2 =  6.04e-14
TEVD.              || AV - VD ||_2 / ||A||_2 =  7.46e-15.   || V'V - I ||_2 =  6.04e-14
Matlab 7 PCG.      Converged to TOL within MAXIT iterations.
Matlab 7 PCGI.     Converged to TOL within MAXIT iterations.
Matlab 7 SYMMLQ.   Converged to TOL within MAXIT iterations.
SYMMLQ SOL.        Reasonable accuracy achieved, given eps.
SYMMLQ SOL3.       Reasonable accuracy achieved, given eps.
Matlab 7 MINRES.   Converged to TOL within MAXIT iterations.
MINRES SOL.        A solution to Ax = b was found, given rtol.
MINRES SOL69.      A solution to (poss. singular) Ax = b found, given rtol.
MINRES QLP43.      A solution to (poss. singular) Ax = b found, given rtol.
                   trancond = 1e+007, Mitn = 414, Qitn = 0.
Matlab 7 LSQR.     Iterated MAXIT times but did not converge.
LSQR SOL.          The iteration limit has been reached.
Matlab 7 GMRES(10). Iterated MAXIT times but did not converge.
SQMR.              q^T A q = 0.
Matlab 7 QMR.      Converged to TOL within MAXIT iterations.
```

```
Matlab 7 BICG.      Converged to TOL within MAXIT iterations
Matlab 7 BICGSTAB.  Converged to TOL within MAXIT iterations.

Method              A*v    x(1)     ||x||    ||e||      ||r||   ||Ar||  ||A||    K(A)
                                    direct =||x-xTEVD|| direct  direct
EVD.                 -- 9.501e-01 2.069e+01 0.0e+00   2.1e-13 2.9e-13 2.7e+00 8.3e+03
TEVD.                -- 9.501e-01 2.069e+01 0.0e+00   2.1e-13 2.9e-13 2.7e+00 8.3e+03
Matlab 7 PCG.       467 9.501e-01 2.069e+01 8.5e-10   1.6e-11 2.4e-11  --      --
Matlab 7 PCGI.      351 9.501e-01 2.069e+01 1.6e-06   2.1e-09 2.6e-09  --      --
Matlab 7 SYMMLQ.    465 9.501e-01 2.069e+01 8.7e-10   2.0e-11 2.6e-11  --      --
SYMMLQ SOL.         407 9.501e-01 2.069e+01 5.5e-08   5.8e-10 8.7e-10 3.1e+01 4.9e+00
SYMMLQ SOL3.        491 9.501e-01 2.069e+01 1.1e-10   4.0e-12 5.1e-12 2.1e+00 4.9e+00
Matlab 7 MINRES.    424 9.501e-01 2.069e+01 4.8e-08   2.7e-11 7.0e-12  --      --
MINRES SOL.         365 9.501e-01 2.069e+01 1.5e-06   6.1e-10 5.9e-11 4.0e+01 5.0e+00
MINRES SOL69.       414 9.501e-01 2.069e+01 6.4e-08   7.1e-11 2.4e-11 2.1e+00 1.2e+04
MINRES QLP43.       414 9.501e-01 2.069e+01 6.4e-08   7.1e-11 2.4e-11 2.1e+00 1.9e+03
Matlab 7 LSQR.     5120 9.501e-01 2.068e+01 3.7e-01   1.4e-04 6.1e-06  --      --
LSQR SOL.          5120 9.501e-01 2.068e+01 3.4e-01   1.3e-04 1.8e-05 8.9e+01 2.4e+05
Matlab 7 GMRES(10). 2560 9.501e-01 2.069e+01 6.5e-04   2.2e-07 2.9e-08  --      --
SQMR.               327 9.501e-01 2.069e+01 2.2e-06   1.4e-08 1.2e-08  --      --
Matlab 7 QMR.       912 9.501e-01 2.069e+01 2.5e-08   2.7e-11 1.9e-11  --      --
Matlab 7 BICG.      932 9.501e-01 2.069e+01 2.0e-09   2.5e-11 3.1e-11  --      --
Matlab 7 BICGSTAB. 1732 9.501e-01 2.069e+01 6.9e-08   2.7e-11 1.4e-11  --      --
```

We ran the example again with $\delta = 10^{-10}$ in the diagonal scaling (3.35). The condition number of $DAD$ was then approximately $10^2$, and the number of iterations reduced further.

```
Title  = Matrix Market, MHD1280B: Alfven Spectra in Magnetohydrodynamics
n     =  1280,   maxit =  2560,   nnz    = 22778,   nnz / n = 18,   numerical rank = 1280
shift = 0.00e+00,  tol = 1e-12, maxxnorm = 1e+09,  maxcond  = 1e+14, ||b|| = 2.8e+01.
No. of positive eigenvalue(s)    = 1280: between  3.36e-02 and  2.90e+00.
No. of almost zero eigenvalue(s) =    0
No. of negative eigenvalue(s)    =    0

Methods:
EVD.                || AV - VD ||_2 / ||A||_2 =  8.59e-15.   || V'V - I ||_2 =  2.72e-14
TEVD.               || AV - VD ||_2 / ||A||_2 =  8.59e-15.   || V'V - I ||_2 =  2.72e-14
Matlab 7 PCG.       Converged to TOL within MAXIT iterations.
Matlab 7 PCGI.      Converged to TOL within MAXIT iterations.
Matlab 7 SYMMLQ.    Converged to TOL within MAXIT iterations.
SYMMLQ SOL.         Reasonable accuracy achieved, given eps.
SYMMLQ SOL3.        Reasonable accuracy achieved, given eps.
Matlab 7 MINRES.    Converged to TOL within MAXIT iterations.
MINRES SOL.         A solution to Ax = b was found, given rtol.
MINRES SOL69.       A solution to (poss. singular) Ax = b found, given rtol.
MINRES QLP45.       A solution to (poss. singular) Ax = b found, given rtol.
                    trancond = 1e+007, Mitn = 84, Qitn = 0.
Matlab 7 LSQR.      Converged to TOL within MAXIT iterations.
```

```
LSQR SOL.           Ax - b is small enough, given atol, btol.
Matlab 7 GMRES(10). Converged to TOL within MAXIT iterations.
SQMR.               q^T A q = 0.
Matlab 7 QMR.       Converged to TOL within MAXIT iterations.
Matlab 7 BICG.      Converged to TOL within MAXIT iterations
Matlab 7 BICGSTAB.  Converged to TOL within MAXIT iterations.
```

| Method | A*v | x(1) | ‖x‖ direct | ‖e‖ =‖x-xTEVD‖ | ‖r‖ direct | ‖Ar‖ direct | ‖A‖ | K(A) |
|---|---|---|---|---|---|---|---|---|
| EVD. | -- | 9.501e-01 | 2.069e+01 | 0.0e+00 | 2.5e-13 | 4.0e-13 | 2.9e+00 | 8.6e+01 |
| TEVD. | -- | 9.501e-01 | 2.069e+01 | 0.0e+00 | 2.5e-13 | 4.0e-13 | 2.9e+00 | 8.6e+01 |
| Matlab 7 PCG. | 89 | 9.501e-01 | 2.069e+01 | 7.1e-11 | 2.2e-11 | 3.2e-11 | -- | -- |
| Matlab 7 PCGI. | 73 | 9.501e-01 | 2.069e+01 | 1.2e-08 | 2.1e-09 | 3.4e-09 | -- | -- |
| Matlab 7 SYMMLQ. | 88 | 9.501e-01 | 2.069e+01 | 7.1e-11 | 2.2e-11 | 3.2e-11 | -- | -- |
| SYMMLQ SOL. | 76 | 9.501e-01 | 2.069e+01 | 3.8e-09 | 7.3e-10 | 1.1e-09 | 1.5e+01 | 2.5e+00 |
| SYMMLQ SOL3. | 89 | 9.501e-01 | 2.069e+01 | 4.3e-11 | 1.4e-11 | 2.0e-11 | 2.1e+00 | 2.7e+00 |
| Matlab 7 MINRES. | 87 | 9.501e-01 | 2.069e+01 | 2.5e-10 | 2.2e-11 | 1.4e-11 | -- | -- |
| MINRES SOL. | 78 | 9.501e-01 | 2.069e+01 | 4.3e-09 | 3.6e-10 | 2.5e-10 | 3.2e+01 | 2.5e+00 |
| MINRES SOL69. | 84 | 9.501e-01 | 2.069e+01 | 9.6e-10 | 6.7e-11 | 3.9e-11 | 2.1e+00 | 1.1e+02 |
| MINRES QLP45. | 84 | 9.501e-01 | 2.069e+01 | 9.6e-10 | 6.7e-11 | 3.9e-11 | 2.1e+00 | 2.2e+01 |
| Matlab 7 LSQR. | 916 | 9.501e-01 | 2.069e+01 | 1.1e-10 | 2.4e-11 | 1.4e-11 | -- | -- |
| LSQR SOL. | 862 | 9.501e-01 | 2.069e+01 | 4.2e-09 | 7.2e-10 | 4.0e-10 | 3.8e+01 | 2.3e+03 |
| Matlab 7 GMRES(10). | 144 | 9.501e-01 | 2.069e+01 | 6.5e-10 | 2.5e-11 | 1.4e-11 | -- | -- |
| SQMR. | 69 | 9.501e-01 | 2.069e+01 | 4.7e-08 | 8.7e-09 | 1.2e-08 | -- | -- |
| Matlab 7 QMR. | 176 | 9.501e-01 | 2.069e+01 | 2.5e-10 | 2.2e-11 | 1.4e-11 | -- | -- |
| Matlab 7 BICG. | 178 | 9.501e-01 | 2.069e+01 | 7.1e-11 | 2.2e-11 | 3.2e-11 | -- | -- |
| Matlab 7 BICGSTAB. | 246 | 9.501e-01 | 2.069e+01 | 2.5e-10 | 1.4e-11 | 4.0e-12 | -- | -- |

### 4.3.3 With Binormalization

Ten sweeps of BIN took 1.5 seconds and produced $\kappa(DAD) \approx 100$. The number of iterations of the Hermitian solvers reduced to less than 100. To reproduce this example, run `testminresQLP27(30)`.

```
Title  = Matrix Market, MHD1280B: Alfven Spectra in Magnetohydrodynamics
n     =  1280,   maxit =  2560,   nnz    = 22778,   nnz / n = 18,   numerical rank = 1280
shift = 0.00e+00,  tol = 1e-12, maxxnorm = 1e+09,  maxcond  = 1e+14, ||b|| = 2.5e+01.
No. of positive eigenvalue(s)    = 1280: between  2.27e-02 and  2.10e+00.
No. of almost zero eigenvalue(s) =    0
No. of negative eigenvalue(s)    =    0
```

```
EVD.                || AV - VD ||_2 / ||A||_2 =  6.55e-15.   || V'V - I ||_2 =  2.07e-14
TEVD.               || AV - VD ||_2 / ||A||_2 =  6.55e-15.   || V'V - I ||_2 =  2.07e-14
Matlab 7 PCG.       Converged to TOL within MAXIT iterations.
Matlab 7 PCGI.      Converged to TOL within MAXIT iterations.
Matlab 7 SYMMLQ.    Converged to TOL within MAXIT iterations.
SYMMLQ SOL.         Reasonable accuracy achieved, given eps.
SYMMLQ SOL3.        Reasonable accuracy achieved, given eps.
```

```
Matlab 7 MINRES.    Converged to TOL within MAXIT iterations.
MINRES SOL.         A solution to Ax = b was found, given rtol.
MINRES SOL69.       A solution to (poss. singular) Ax = b found, given rtol.
MINRES QLP43.       A solution to (poss. singular) Ax = b found, given rtol.
                    trancond = 1e+007, Mitn = 85, Qitn = 0.
Matlab 7 LSQR.      Converged to TOL within MAXIT iterations.
LSQR SOL.           Ax - b is small enough, given atol, btol.
Matlab 7 GMRES(10). Converged to TOL within MAXIT iterations.
SQMR.               q^T A q = 0.
Matlab 7 QMR.       Converged to TOL within MAXIT iterations.
Matlab 7 BICG.      Converged to TOL within MAXIT iterations
Matlab 7 BICGSTAB.  Converged to TOL within MAXIT iterations.
```

| Method | A*v | x(1) | ‖x‖ direct | ‖e‖ =‖x−xTEVD‖ | ‖r‖ direct | ‖Ar‖ direct | ‖A‖ | K(A) |
|---|---|---|---|---|---|---|---|---|
| EVD. | -- | 9.501e-01 | 2.069e+01 | 0.0e+00 | 2.2e-13 | 3.0e-13 | 2.1e+00 | 9.3e+01 |
| TEVD. | -- | 9.501e-01 | 2.069e+01 | 0.0e+00 | 2.2e-13 | 3.0e-13 | 2.1e+00 | 9.3e+01 |
| Matlab 7 PCG. | 88 | 9.501e-01 | 2.069e+01 | 6.8e-11 | 1.9e-11 | 2.4e-11 | -- | -- |
| Matlab 7 PCGI. | 74 | 9.501e-01 | 2.069e+01 | 1.0e-08 | 1.3e-09 | 1.6e-09 | -- | -- |
| Matlab 7 SYMMLQ. | 87 | 9.501e-01 | 2.069e+01 | 6.8e-11 | 1.9e-11 | 2.4e-11 | -- | -- |
| SYMMLQ SOL. | 77 | 9.501e-01 | 2.069e+01 | 4.4e-09 | 5.0e-10 | 6.3e-10 | 1.1e+01 | 2.7e+00 |
| SYMMLQ SOL3. | 90 | 9.501e-01 | 2.069e+01 | 1.8e-11 | 5.0e-12 | 6.1e-12 | 1.5e+00 | 2.7e+00 |
| Matlab 7 MINRES. | 86 | 9.501e-01 | 2.069e+01 | 1.9e-10 | 2.3e-11 | 1.7e-11 | -- | -- |
| MINRES SOL. | 77 | 9.501e-01 | 2.069e+01 | 7.1e-09 | 3.8e-10 | 1.9e-10 | 2.7e+01 | 2.7e+00 |
| MINRES SOL69. | 85 | 9.501e-01 | 2.069e+01 | 7.1e-10 | 5.4e-11 | 3.2e-11 | 1.5e+00 | 1.1e+02 |
| MINRES QLP43. | 85 | 9.501e-01 | 2.069e+01 | 7.1e-10 | 5.4e-11 | 3.2e-11 | 1.6e+00 | 2.4e+01 |
| Matlab 7 LSQR. | 826 | 9.501e-01 | 2.069e+01 | 1.2e-10 | 2.4e-11 | 1.3e-11 | -- | -- |
| LSQR SOL. | 784 | 9.501e-01 | 2.069e+01 | 3.1e-09 | 5.4e-10 | 2.7e-10 | 2.8e+01 | 2.2e+03 |
| Matlab 7 GMRES(10). | 150 | 9.501e-01 | 2.069e+01 | 6.6e-10 | 2.3e-11 | 1.3e-11 | -- | -- |
| SQMR. | 68 | 9.501e-01 | 2.069e+01 | 1.0e-07 | 1.3e-08 | 1.1e-08 | -- | -- |
| Matlab 7 QMR. | 174 | 9.501e-01 | 2.069e+01 | 1.9e-10 | 2.3e-11 | 1.7e-11 | -- | -- |
| Matlab 7 BICG. | 176 | 9.501e-01 | 2.069e+01 | 6.8e-11 | 1.9e-11 | 2.4e-11 | -- | -- |
| Matlab 7 BICGSTAB. | 248 | 9.501e-01 | 2.069e+01 | 4.3e-10 | 2.2e-11 | 1.9e-11 | -- | -- |

## 4.4 Effects of Rounding Errors in MINRES-QLP

The recurred residual norms $\phi_k^M$ in MINRES usually approximates the directly computed ones $\|r_k^M\|$ very well, until $\|r_k^M\|$ becomes small. We observe that $\phi_k^M$ continues to decrease in the last few iterations, even though $\|r_k^M\|$ has become stagnant. This is desirable in the sense that the stopping rule will cause termination, although the final solution is not as accurate as predicted.

We present similar plots of MINRES-QLP in the following examples, with the corresponding quantities as $\phi_k^Q$ and $\|r_k^Q\|$. We observe that except in very ill-conditioned least-squares problems, $\phi_k^Q$ approximates $\|r_k^Q\|$ very closely.

1. four singular linear systems: see Figure 4.2.
2. four singular least-squares problems: see Figure 4.3.

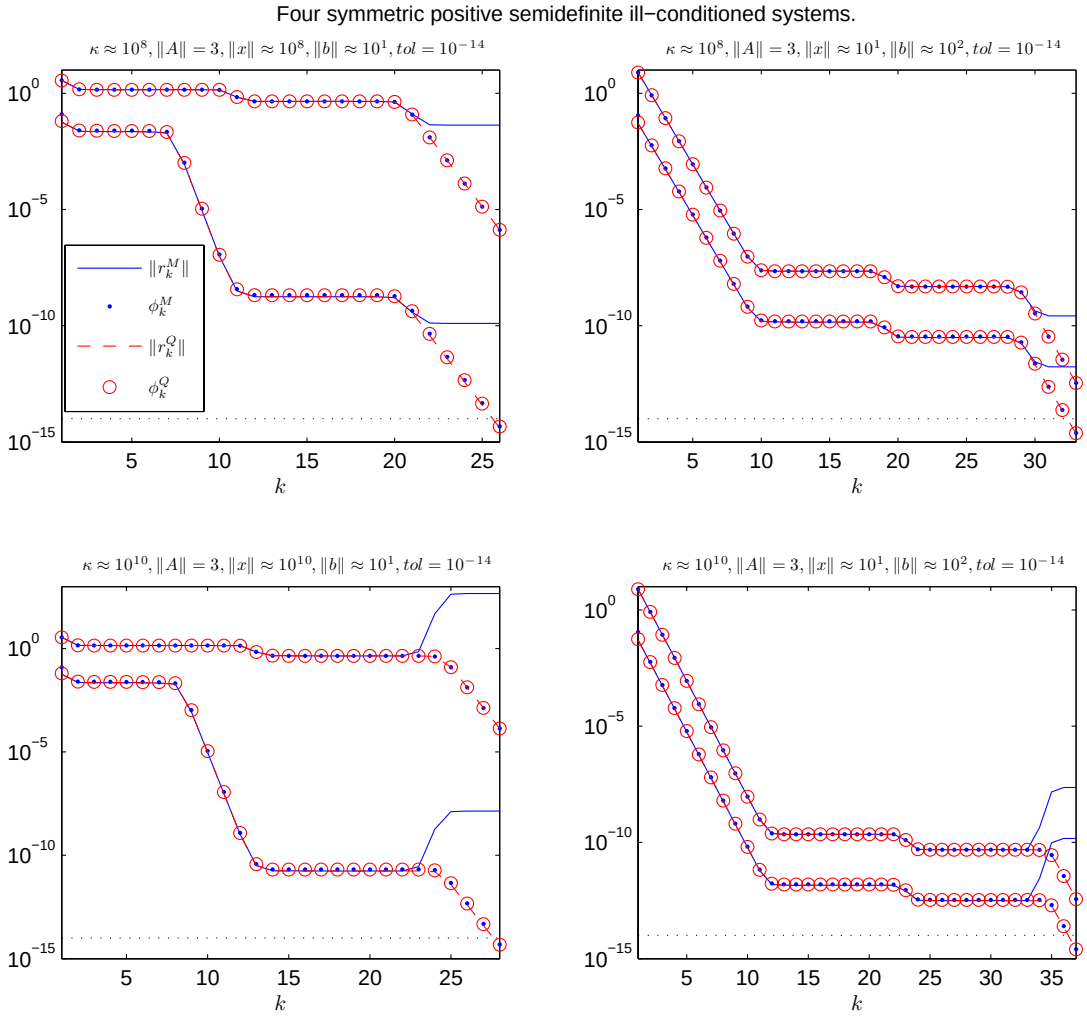


FIGURE 4.2 *Solving $Ax = b$ with symmetric positive semidefinite $A = Q\,\mathrm{diag}([0_5, \eta, 2\eta, 2 : \frac{1}{789} : 3])Q$ of dimension $n = 797$, nullity 5, and norm $\|A\|_2 = 3$, where $Q = I - (2/n)ww^T$ is a Householder matrix generated by $v = [0_5, 1, \ldots, 1]^T$, $w = v/\|v\|$. These plots illustrate and compare the effect of rounding errors in MINRES and MINRES-QLP similar to the nonsingular example in Figure 3.1. The upper part of each plot shows the computed and recurred residual norms, and the lower part shows the computed and recurred normwise relative backward errors (NRBE). MINRES and MINRES-QLP terminate when the recurred NRBE is less than the given* $\mathrm{tol} = 10^{-14}$.

***Upper left:*** *$\eta = 10^{-8}$ and thus $\kappa(A) \approx 10^8$. Also $b = e$ and therefore $\|x\| \gg \|b\|$. The graphs of MINRES's directly computed residual norms $\|r_k^M\|$ and recurrently computed residual norms $\phi_k^M$ start to differ at the level of $10^{-1}$ starting at iteration 21, while the values $\phi_k^Q \approx \|r_k^Q\|$ from MINRES-QLP decrease monotonically and stop near $10^{-6}$ at iteration 26.*

***Upper right:*** *Again $\eta = 10^{-8}$ but $b = Ae$. Thus $\|x\| = \|e\| = O(\|b\|)$. The MINRES graphs of $\|r_k^M\|$ and $\phi_k^M$ start to differ when they reach a much smaller level of $10^{-10}$ at iteration 30. The MINRES-QLP $\phi_k^Q$'s are excellent approximations of $\phi_k^Q$, with both reaching $10^{-13}$ at iteration 33.*

***Lower left:*** *$\eta = 10^{-10}$ and thus $A$ is even more ill-conditioned than the matrix in the upper plots. Here $b = e$ and $\|x\|$ is again exploding. MINRES ends with $\|r_k^M\| \approx 10^2$, which means no convergence at all, while MINRES-QLP reaches a residual norm of $10^{-4}$.*

***Lower right:*** *$\eta = 10^{-10}$ and $b = Ae$. The final MINRES residual norm $\|r_k^M\| \approx 10^{-8}$, which is satisfactory but not as accurate as $\phi_k^M$ claims at $10^{-13}$. MINRES-QLP again has $\phi_k^Q \approx \|r_k^Q\| \approx 10^{-13}$ at iteration 37.*

*This figure can be reproduced from the Matlab program* `DPtestSing5.m`.

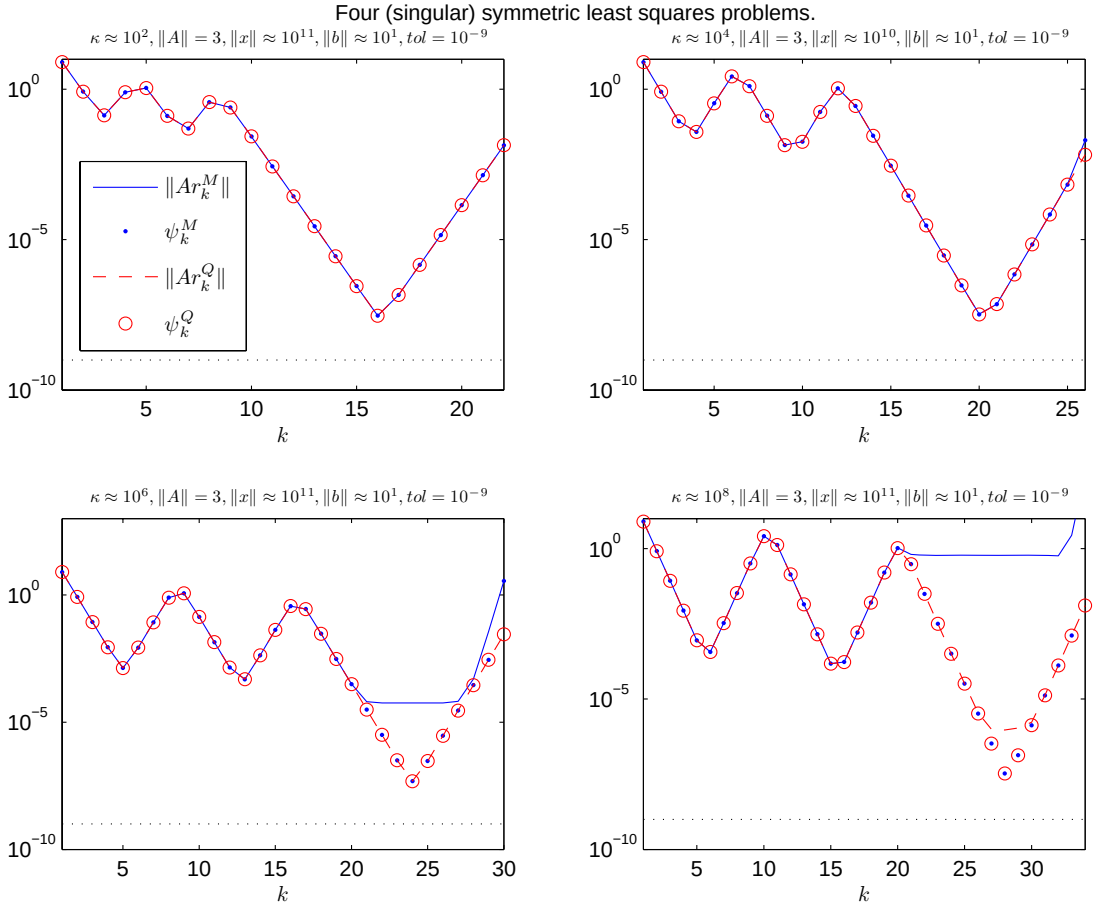


FIGURE 4.3 *Solving $Ax = b$ with symmetric positive semidefinite $A = Q\,\mathrm{diag}([0_5, \eta, 2\eta, 2 : \frac{1}{789} : 3])Q$ of dimension $n = 797$ with $\|A\|_2 = 3$, where $Q = I - (2/n)ee^T$ is a Householder matrix generated by $e = [1, \ldots, 1]^T$.*

***Upper left:*** *$\eta = 10^{-2}$ and thus $\mathrm{cond}(A) \approx 10^2$. Also $b = e$ and therefore $\|x\| \gg \|b\|$. The graphs of MINRES's directly computed $\|Ar_k^M\|$ and recurrently computed $\psi_k^M$, and also $\psi_k^Q \approx \|Ar_k^Q\|$ from MINRES-QLP, match very well throughout the iterations.*

***Upper right:*** *Here, $\eta = 10^{-4}$ and $A$ is more ill-conditioned than the last example. The final MINRES residual norm $\psi_k^M \approx \|Ar_k^M\|$ is slightly larger than the final MINRES-QLP residual norm $\psi_k^Q \approx \|Ar_k^Q\|$. The MINRES-QLP $\psi_k^Q$ are excellent approximations of $\psi_k^Q$, and both reach $10^{-13}$ at iteration 33.*

***Lower left:*** *$\eta = 10^{-6}$ and $\mathrm{cond(A)} \approx 10^6$. MINRES's $\psi_k^M$ and $\|Ar_k^M\|$ differ starting at iteration 21. Eventually, $\|Ar_k^M\| \approx 3$, which means no convergence. MINRES-QLP reaches a residual norm of $\psi_k^Q = \|Ar_k^Q\| = 10^{-2}$.*

***Lower right:*** *$\eta = 10^{-8}$. MINRES performs even worse than last example. MINRES-QLP reaches a minimum $\|Ar_k^Q\| \approx 10^{-7}$ but the solver does not manage to shut down soon enough and ends with a final $\psi_k^Q = \|Ar_k^Q\| = 10^{-2}$. The values of $\psi_k^Q$ and $\|Ar_k^Q\|$ differ only at iterations 27–28.*

*This figure can be reproduced from the Matlab program* `DPtestLSSing3.m`.

# Chapter 5

# Computation of Null Vectors, Eigenvectors, and Singular Vectors

We return now to the original motivating problem described in Chapter 1: that of computing a null vector for an arbitrary matrix $A$. If the nullity of $A$ is one, then the null vector is unique up to a (complex) scalar multiple and we consider the null-vector problem well-posed. Otherwise, it is ill-posed and we are satisfied with a set of orthogonal unit vectors that span $\mathcal{N}(A)$.

## 5.1 Applications

### 5.1.1 Eigenvalue Problem

If an eigenvalue $\hat{\lambda}$ of a given matrix $A$ is known or approximated, the method of *inverse iteration* as defined in (1.2) or *Rayleigh quotient iteration* in (5.1) could be used in conjunction with Krylov subspace solvers [87, 81, 101, 5, 77, 9]. Either scheme involves a sequence of linear systems in the following iteration:

$$(A - \hat{\lambda}I)x_k = v_{k-1}, \qquad v_k = x_k/\|x_k\|, \qquad \hat{\lambda} \leftarrow v_k^T A v_k, \qquad k = 1, \ldots, k_I, \tag{5.1}$$

where the number of iterations $k_I$ would be only 1 or few. The matrix $A - \hat{\lambda}I$ is intentionally singular, and the computed solutions $x_k$ are expected to grow extremely large ($\|x_k\| = O(1/\varepsilon)$, where $\varepsilon$ is the machine precision), so that the normalized vectors $v_k$ would satisfy

$$(A - \hat{\lambda}I)v_k \approx 0 \tag{5.2}$$

and hence $Av_k \approx \hat{\lambda}v_k$ as required.

ARPACK [67] provides alternative Lanczos- and Arnoldi-based approaches for computing several eigenvalues and/or vectors.

### 5.1.2 Singular Value Problem

The singular value problem $Av_i = \sigma_i u_i$, $A^T u_i = \sigma_i v_i$, may be reformulated as an eigenvalue problem, or a null-vector problem when $\sigma_i$ is known:

$$\begin{bmatrix} & A \\ A^T & \end{bmatrix} \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \sigma_i \begin{bmatrix} u_i \\ v_i \end{bmatrix} \qquad \Leftrightarrow \qquad \left( \begin{bmatrix} & A \\ A^T & \end{bmatrix} - \sigma_i I \right) \begin{bmatrix} u_i \\ v_i \end{bmatrix} = 0.$$

AMRES [97] is a special version of MINRES for this purpose. ARPACK [67] operates on the same matrix to obtain several singular values/vectors. PROPACK [65] operates directly on $A$.

### 5.1.3 Generalized, Quadratic, and Polynomial Eigenvalue Problems

Given matrices $A_0, A_1, \ldots, A_d \in \mathbb{C}^{n\times n}$, we want to find an eigenvalue $\lambda \in \mathbb{C}$ and its corresponding eigenvector $x \in \mathbb{C}^n$, $x \neq 0$ such that

$$(A_0 + \lambda A_1 + \cdots + \lambda^d A_d)x = 0. \tag{5.3}$$

This is called the *polynomial eigenvalue problem* [95, 28, 104, 59].

In particular, when $d = 2$, it is called the *quadratic eigenvalue problem*. When $d = 1$, it is called the *generalized eigenvalue problem*. One may also encounter *rectangular* generalized eigenvalue problems [15, 99], where $A_0$ and $A_1$ are rectangular matrices of the same size.

Clearly, when $A_1 = I$, the generalized eigenvalue problem reduces to an eigenvalue problem. However, these two problem classes are very different in nature. An eigenvalue problem has $n$ eigenvalues guaranteed by the roots of the degree-$n$ characteristic polynomial $p_n(\lambda) = \det(A-\lambda I)$. A generalized eigenvalue problem may not have $n$ eigenvalues. For example, if $A_0 = I$ and $A_1 = 0$, then $A_0 + \lambda A_1 = I \neq 0$ for any scalar $\lambda$. Generally speaking, the number of eigenvalues in a generalized eigenvalue problem is given by the nullity of $A_0 + \lambda A_1$.

When $\lambda$ is known in a polynomial eigenvalue problem (5.3), we effectively have a null-vector problem. The corresponding generalized eigenvector may be solved by inverse iteration coupled with CG-type methods [68, 117, 29, 5, 113].

### 5.1.4 Multiparameter Eigenvalue Problem

For this problem problem [4, 1] we want to find scalars $\lambda_1, \ldots, \lambda_d \in \mathbb{C}$ and a corresponding eigenvector $x \in \mathbb{C}^n$, $x \neq 0$ such that

$$(A_0 + \lambda_1 A_1 + \cdots + \lambda_d A_d)x = 0. \tag{5.4}$$

## 5.2 Computing a Single Null Vector

We may abstract problems (1.2) and (5.2) by writing $A$ in place of the (nearly) singular $A - \hat{\lambda} I$, and likewise for the other more general eigenvalue problems. This gives us a null-vector problem (or homogeneous equation) $Av \approx 0$, with $A$ essentially singular.

Chapter 1 already discussed the following iterative methods:

**Inverse iteration:** For a random vector $b$, apply LSQR or MINRES to the least-squares problem $\min_x \|Ax - b\|_2$, $A \in \mathbb{R}^{m\times n}$, $\operatorname{rank}(A) < n$. The normalized vector $v = x/\|x\|$ approximates a null vector of $A$.

**Least-squares approach:** For a random vector $c$, apply LSQR, MINRES or MINRES-QLP to the problem $\min_y \|A^T y - c\|_2$, $A \in \mathbb{R}^{m\times n}$, $\operatorname{rank}(A) < n$. The optimal residual $s = c - A^T y$ satisfies $As = 0$, and the required null vector is $v = s/\|s\|$. Convergence should occur sooner than with inverse iteration, as we have seen in Figures 1.1–1.2.

We compare the vectors involved in the inverse-iteration approach and the least-squares approach in Table 5.1.

TABLE 5.1
*Null vectors from various Krylov subspace methods using the inverse-iteration and matrix-transpose approaches.*

| Method | Stopping Conditions Disabled | Normal Stopping Conditions |
|---|---|---|
| CG | $x_k/\|x_k\|$, or $q_k/\|q_k\|$ | $r_k/\|r_k\|$ |
| SYMMLQ | $x_k^C/\|x_k^C\|$ | $r_k/\|r_k\|$ or $\bar{w}_k$ |
| MINRES | $x_k/\|x_k\|$ | $r_k/\|r_k\|$ |
| MINRES-QLP | $x_k/\|x_k\|$ | $r_k/\|r_k\|$ |
| LSQR | $x_k/\|x_k\|$ | $r_k/\|r_k\|$ |

TABLE 5.2
*Algorithm **MCGLS** [66] for solving a sequence of least-squares problems* $\min \|Ax - b^{(k)}\|$.

**MCGLS** $(A, b^{(1)}, \ldots, b^{(q)}, \text{tol}, \text{maxit}) \to x^{(1)}, \ldots, x^{(q)}$
**for** $k = 1, 2, \ldots, q$
  $r_0^{(k)} = b^{(k)}$
**end**
**for** $k = 1, 2, \ldots, q$
  `//select the kth system as seed`
  $p_0 = s_0 = A^T r_0^{(k)}, \qquad \gamma_0 = \|s_0\|_2^2, \qquad i = 0$
  **while** $\gamma_i >$ *operatornametol* **and** $i \le$ maxit
    `//take a CGLS step`
    $q_i = Ap_i, \qquad \alpha_i = \gamma_i/\|q_i\|_2^2, \qquad x_{i+1}^{(k)} = x_i^{(k)} + \alpha_i p_i, \qquad r_{i+1}^{(k)} = r_i^{(k)} - \alpha_i q_i,$
    $s_{i+1} = A^T r_{i+1}^{(k)}, \qquad \gamma_{i+1} = \|s_{i+1}\|, \qquad \beta_i = \gamma_{i+1}/\gamma_i$
    `//Perform Galerkin projection`
    **for** $j = k+1, k+2, \ldots, q$
      $\eta_j = (q_i^T r_0^{(j)})\alpha_i/\gamma_i, \qquad x_0^{(j)} = x_0^{(j)} + \eta_j p_i, \qquad r_0^{(j)} = r_0^{(j)} - \eta_j q_i,$
    **end**
    $p_{i+1} = s_{i+1} + \beta_i p_i, \qquad i = i + 1$
  **end**
**end**

## 5.3 Computing Multiple Null Vectors

If the nullity of $A$ is bigger than one and we wish to compute $q$ null vectors for some $q$ such that $1 < q \le \text{nullity}(A)$, a first strategy is to apply the least-squares approach repeatedly, solving $q$ singular least-squares problems with different random right-hand sides $c$ (chosen in advance).

### 5.3.1 MCGLS: Least-Squares with Multiple Right-Hand Sides

Larsen [66, Paper A, section 4.2] has proposed MCGLS for this purpose; see Table 5.2 for solving a sequence of least-squares problems $\min \|Ax - b^{(k)}\|$, $k = 1, 2, \ldots, q$. We would apply the method to the problems $y^{(k)} = \arg\min \|A^T y - c^{(k)}\|$ with $q$ random vectors $c^{(k)}$. At the end, we could orthogonalize all the residual vectors $r^{(k)} = c^{(k)} - A^T y^{(k)}$ by the modified Gram-Schmidt process.

TABLE 5.3
*Algorithm **MLSQRnull** for computing multiple orthogonal null vectors.* MATLAB *file:* `NullBasis.m`.

**MLSQRnull**$(A, \text{tol}, \text{maxit}) \rightarrow \text{nullity}, r_1, r_2, \ldots$
**for** $i = 1, 2, \ldots$
  Choose random vector $c$
  Orthogonalize $c$ to $r_1, \ldots, r_{i-1}$
  $c \leftarrow c/\|c\|$
  $y = \textbf{LSQR}(A^T, c, \text{tol}, \text{maxit})$
  $r_i = c - A^T y$
  **if** $\|r_i\| < \text{tol}(\|A\|\|y\| + \|c\|)$  `// ` $A^T y \approx c$ `is compatible`
    $\text{nullity} = i - 1$,  STOP
  **end**
**end**

### 5.3.2 MLSQR: Least-Squares with Multiple Right-Hand Sides

In [82, section 8.7] (see also [13]), it is demonstrated that CGLS and LSQR have comparable numerical performance on well-conditioned least-squares problems, but the latter can be superior on ill-conditioned problems. Thus it is natural to consider using LSQR instead of CGLS in an ill-conditioned least-squares problem with multiple right-hand sides. This was already suggested by Larsen [66, Paper A] with the caution that reorthogonalization (or at least partial reorthogonalization) would be necessary in the Golub-Kahan process.

We should note Björck's [11] band decomposition for least-squares with multiple right-hand sides—a natural extension of the Golub-Kahan bidiagonalization and LSQR.

### 5.3.3 MLSQRnull: Multiple Null Vectors

For our null-vector problem, we do not have to generate all right-hand sides in the beginning and thus we can be more memory efficient. Also, we can generate random right-hand sides in increasingly small subspaces in the following fashion, so that the LSQR iterations might be reduced.

With a slight change of notation, suppose we have obtained the first null vector $r_1$ normalized so that $\|r_1\| = 1$ from the solution $y_1$ of $\min_y \|A^T y - c_1\|$. To obtain the second null vector, we choose a nonzero vector $c_2 \notin \mathcal{R}(A^T)$ and $c_2 \perp r_1$. Then the residual $r_2 = c_2 - A^T y_2$ from the solution $y_2$ of $\min_y \|A^T y - c_2\|$ is orthogonal to $r_1$ because

$$r_1^T r_2 = r_1^T (c_2 - A^T y_2) = r_1^T c_2 - y_2^T A r_1 = r_1^T c_2 = 0.$$

Thus, $r_2$ is a second null vector of $A$. We can proceed to choose $c_3 \notin \mathcal{R}(A^T)$ and make $c_3 \perp r_1, r_2$ by the modified Gram-Schmidt process. Repeat the procedure to get all the null vectors of $A$.

We list the steps as algorithm **MLSQRnull** in Table 5.3.

In practice, to produce a vector not in $R(A^T)$ for a given singular matrix $A$, we simply generate a random vector: the probability of it having no component in $\mathcal{N}(A)$ is zero.

## 5.4 Numerical Experiments on Unsymmetric Systems

### 5.4.1 The PageRank Problem

In this application, we happen to know an exact eigenvalue of an unsymmetric matrix, and we wish to compute the corresponding eigenvector $x$. It is the dominant eigenvector of an $n \times n$ Markov matrix $A$ that arises from PageRank [16] (see [64] for a survey):

$$A := \alpha P + \frac{1-\alpha}{n} e e^T, \qquad \alpha \in (0,1), \qquad Ax = x,$$

where $P$ is sparse, unsymmetric, and column-stochastic (satisfying $P^T e = e$). Note that $A$ is dense and thus not explicitly formed. Also, $A$ is both column-stochastic and irreducible (its underlying graph is strongly connected), even if $P$ is reducible. By the Perron-Frobenius theorem, $A$ has a simple maximal eigenvalue equal to 1. The corresponding right-eigenvector $x$ is non-negative, and when normalized to unit 1-norm it is known as the *stationary probability distribution* of the Markov chain represented by $A^T$. Under the PageRank model, $x_i$ measures the importance of the $i$th web page.

In practice, $P$ could have some zero columns, and then $A$ will not be column-stochastic. In this case, we can define an irreducible column-stochastic matrix $B$ of order $n+1$ and its eigenvector as follows (extending Tomlin [106] to handle zero columns in $P$):

$$B = \begin{bmatrix} \alpha P & \frac{1}{n} e \\ e^T - \alpha(e^T P) & 0 \end{bmatrix}, \qquad Bv = v, \quad \text{where} \quad v = \begin{bmatrix} x \\ \theta \end{bmatrix}. \tag{5.5}$$

In essence, the graph of $B$ has one extra node that links to every other node in the graph of $P$.

When $n$ is extremely large (currently many billions for the whole WWW), perhaps the only practical approach is to apply the classical power method, as in the original paper [16].

In the following numerical experiment, we used $P$ from the `harvard500` web-graph—a collection of 500 hyperlinked Harvard web pages assembled by Moler [71]. We defined $B$ using $\alpha = 0.999$ and computed its eigenvector $v$ by the power method and by our least-squares approach of finding the (essentially unique) null vector of $C := B^T - I$:

$$\min_y \|Cy - b\|, \qquad r = b - Cy, \qquad v = r/\|r\| \qquad x = v(1\!:\!n)/\|v(1\!:\!n)\|_1. \tag{5.6}$$

To improve the performance of LSQR on this problem, we used NBIN [70] to compute diagonal matrices $S$ and $T$, then solved the scaled problem

$$\min_{\bar{y}} \|(SCT)\bar{y} - b\|, \qquad s = S(b - (SCT)\bar{y}), \qquad v = s/\|s\|, \qquad x = v(1\!:\!n)/\|v(1\!:\!n)\|_1. \tag{5.7}$$

Note that we usually cannot use two-sided preconditioning on least-squares problems, and indeed $y \neq T\bar{y}$ above, but we do have $C^T v = 0$ in both cases (and hence $Bv = v$).

The power method required about 650 iterations (each consisting of 1 matrix-vector multiplication) to achieve a final error $\|Bv_k - v_k\| \approx 10^{-12}$, while LSQR took 115 iterations (each requiring two matrix-vector multiplications). Figure 5.1 compares $\|Bv_k - v_k\|$ for the two methods. For reference purposes, Figure 5.2 is a bar-graph of the PageRank $x$.

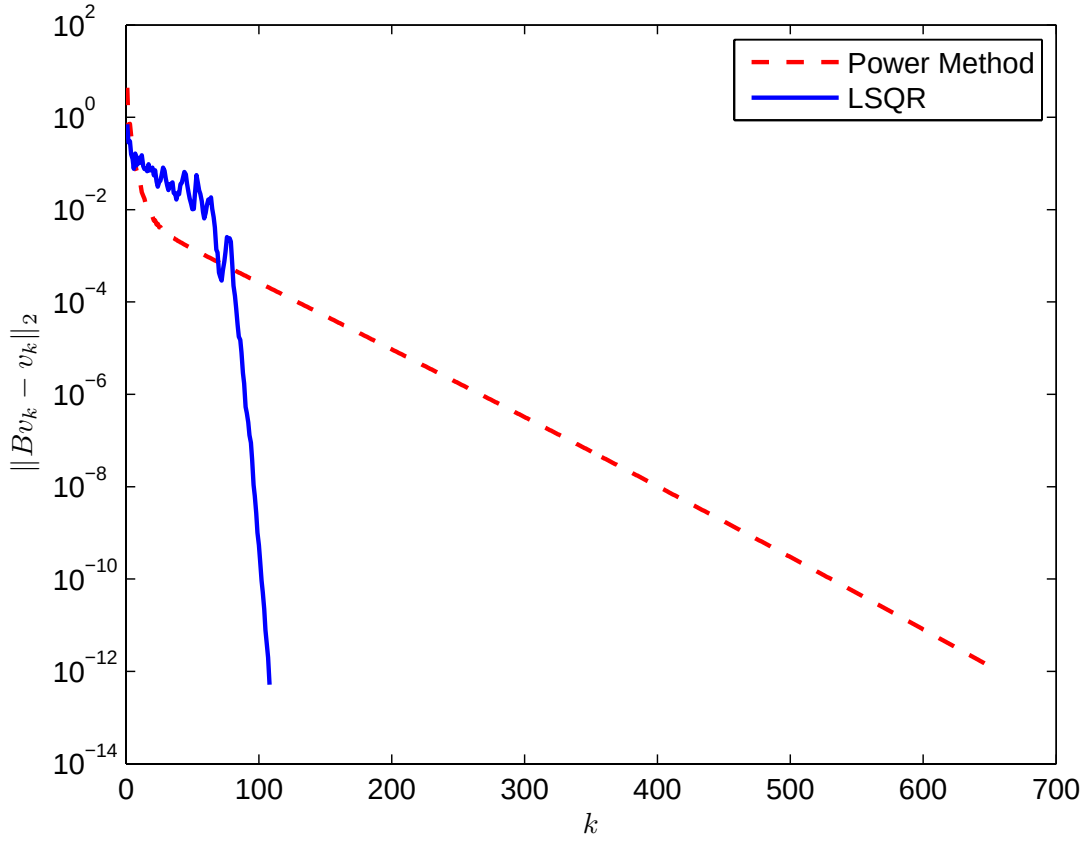


FIGURE 5.1 *Convergence of the power method and the least-squares approach with diagonally preconditioned LSQR (see equation (5.7)) on the* `harvard500` *web matrix, with* $\alpha = 0.999$ *in (5.5).* $\|Bv_k - v_k\|$ *is plotted against iteration number* $k$. *This figure is reproducible by* `PageRank-LSQR_EigsDriverScalingDummyNode4Harvard.m`.

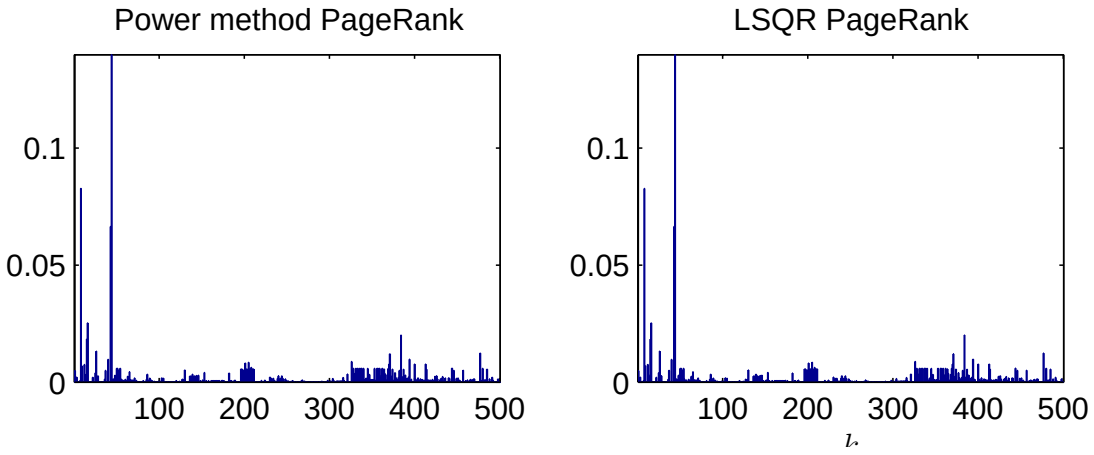


FIGURE 5.2 *PageRank of* `harvard500` *computed by the power method and the least-squares approach with diagonally preconditioned LSQR. Since each solution is very accurate, the figures appear identical. This figure is reproducible by* `PageRankLSQREigsDriverScalingDummyNode4Harvard.m`.

### 5.4.2 PageRank Applied to Citation Data

Research papers are traditionally ranked by number of citations. However, if we model each paper as a web page and the citation relation as hyperlinks, we could compute its PageRank as an alternative literature ranking scheme.

Readers of a paper often follow the bibliography rather closely for further reading (they seldom pick another paper at random!). Thus $\alpha = 0.999$ is a reasonable modeling parameter value. We have obtained the citation data of 531,675 sample papers available from the computer and information science digital library CiteSeer [26] as of August 2002. Figures 5.3 and 5.4 show the convergence and the computed PageRank using the power method and LSQR on problem (5.7). Although LSQR has a rather oscillatory performance on the error measure $\|Bv_k - v_k\|$, it takes only 50% of the matrix-vector products required by the power method to achieve the accuracy $\|Bv_k - v_k\| < 10^{-6}$.

The top 20 papers in descending order of PageRank are as follows:

1. *The UNIX Time-Sharing System*, D. Ritchie and K. Thompson, 1974.
2. *A System for Typesetting Mathematics*, B. Kernighan and L. Cherry, 1975.
3. *Congestion Avoidance in Computer Networks with a Connectionless Network Layer*, R. Jain, K. Ramakrishnan, and D.-M. Chiu, 1997.
4. *Almost Optimal Lower Bounds for Small Depth Circuits*, J. Hastad, 1989.
5. *Relational Queries Computable in Polynomial Time*, N. Immerman, 1986.
6. *Probabilistic Methods in Combinatorics*, J. Spencer, 1974.
7. *Discrepancy in Arithmetic Progressions*, J. Matoušek and J. Spencer, 1996.
8. *Generalized Additive Models*, T. Hastie and R. Tibshirani, 1995.
9. *Why Functional Programming Matters*, J. Hughes, 1984.
10. *Logic Programming with Sets*, G. Kuper, 1990.
11. *Shape and Motion from Image Streams: a Factorization Method*, C. Tomasi and T. Kanade, 1992.
12. *Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management*, S. Kent, 1993.
13. *Deriving Production Rules for Constraint Maintenance*, S. Ceri and J. Widom, 1990.
14. *Reaching Approximate Agreement in the Presence of Faults*, D. Dolev, N. Lynch, S. Pinter, E. Stark, and W. Weihl, 1985.
15. *Hazard Regression*, C. Kooperberg, C. Stone, and Y. Truong, 1994.
16. *Dynamic Perfect Hashing: Upper and Lower Bounds*, M. Dietzfelbinger, A. Karlin, K. Mehlhorn, F. Meyer auf der Heide, H. Rohnert, R. Tarjan, 1990.
17. *On the Length of Programs for Computing Finite Binary Sequences*, G. Chaitin, 1966.
18. *Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers*, D. Balenson, 1993.
19. *Model Selection and Accounting for Model Uncertainty in Linear Regression Models*, A. Raftery, D. Madigan, and J. Hoeting, 1993.
20. *Set-Oriented Production Rules in Relational Database Systems*, J. Widom and S. Finkelstein, 1990.

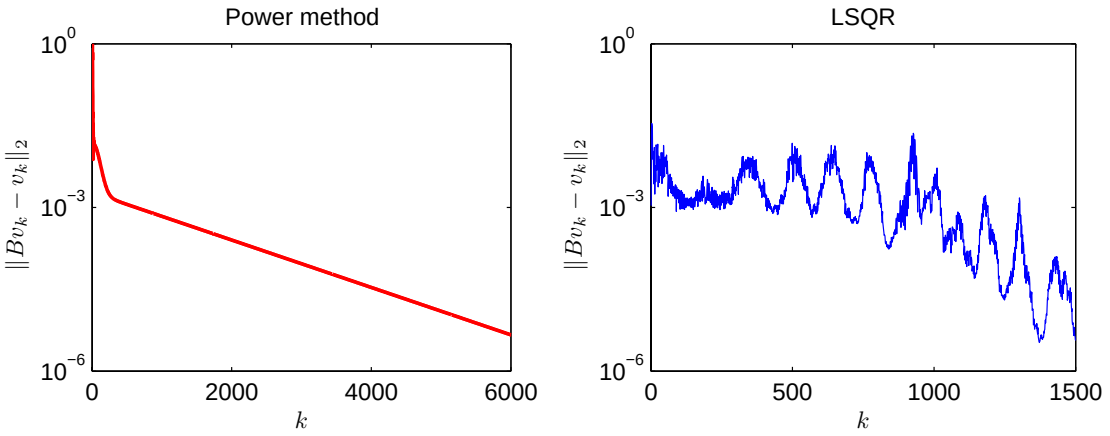


FIGURE 5.3 *Convergence of the power method and the least-squares approach with diagonally preconditioned LSQR (see equation (5.7)) on the CiteSeer citation matrix, with* $\alpha = 0.999$ *in (5.5). This figure is reproducible by* `PageRankLSQR_EigsDriverScalingDummyNode4Cite.m`.

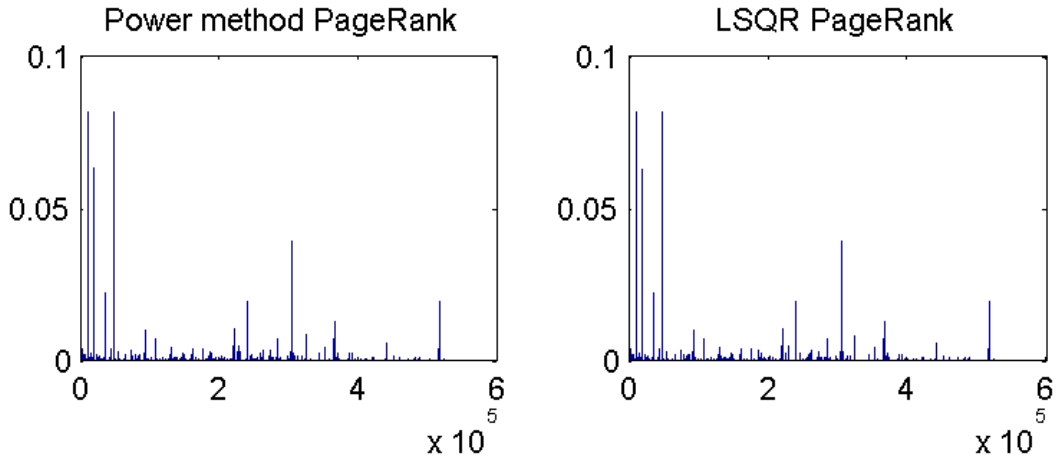


FIGURE 5.4 *Essentially identical PageRank solutions from the methods in the preceding figure. This figure is reproducible by* `PageRankLSQR_EigsDriverScalingDummyNode4Cite.m`.

### 5.4.3 A Multiple Null-Vector Problem from Helioseismology

Here we present a problem that arises from a helioseismology application at HEPL (the Hansen Experimental Physics Laboratory at Stanford) in 2003. Using algorithm `MLSQRnull` in Table 5.3, we computed several null vectors of a dense, square matrix $A$ of order $n = 540,672$ and condition number $\kappa(A) = O(10^4)$. In this case, $A$ is defined by convolution operators for computing $Ax$ and $A^T y$, given inputs $x$ and $y$.

Figure 5.5 shows $\|r_k\|$ and $\|Ar_k\|$ for each LSQR iteration on $\min \|A^T x - c\|$ (where $r_k = c - A^T x_k$), along with the $k$th estimates of $x(1)$ and $\|A\|$, using a particular random vector $c$. The figures for the other null-vectors are similar and thus omitted.

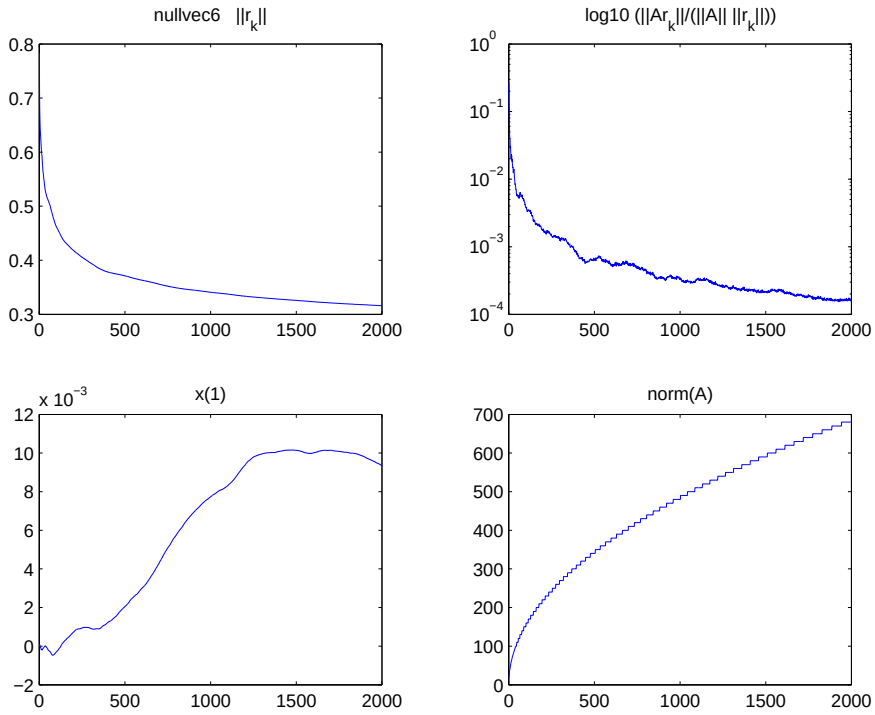


FIGURE 5.5 *An application that arises from helioseismology. $A$ is a large, dense, unsymmetric square matrix of order $n = 540,672$. We compute $x = \texttt{LSQR}(A^T, c)$ for a random vector $c$. The residual vector $r = c - A^T x$ is a null vector of $A$. This experiment was performed on a 2.4GHz Intel Xeon Linux machine with 2GB RAM. It took 5 hours for 2,000 iterations. To reproduce this figure, run* `td_inversion_nullvec6.m`.

# Chapter 6

# Conclusions and Future Work

## 6.1 Summary

Krylov subspace methods find approximate solutions to linear systems $Ax = b$ or least-squares problems $\min \|Ax - b\|$. Each iteration requires one matrix-vector product $Av_k$ (and possibly one product $A^T u_k$) for a certain vector $v_k$ (and $u_k$) generated by the method. This is the mechanism by which $A$ makes itself known. The $k$th approximate solution $x_k$ lies in the Krylov subspace spanned by the vectors $\{b, Ab, A^2b, \ldots, A^{k-1}b\}$ (in most cases) or by the vectors $\{A^Tb, (A^TA)A^Tb, (A^TA)^2A^Tb, \ldots, (A^TA)^{k-1}A^Tb\}$.

Table 6.1 summarizes the main Krylov subspace methods according to problem types. Our solver MINRES-QLP fills a gap by extending the symmetric solver MINRES to the case of singular or ill-conditioned symmetric systems.

TABLE 6.1
*Problem types and algorithms. CGLS applies to the same categories as LSQR. The condition number of $A$ is denoted by $\kappa$.*

| **Problem** | *A full rank or rank-deficient* | | *A unknown rank* |
|---|---|---|---|
| | *compatible*<br>$Ax = b$ | *incompatible*<br>$\min \|Ax - b\|$ | *unknown compatibility*<br>*unknown* $\kappa$ |
| *square*<br>$A = A^*$ | $\pm A \succeq 0$:<br>CG<br>---<br>$\pm A \not\succeq 0$, $\kappa = O(1)$:<br>MINRES<br>SYMMLQ<br>SQMR<br>---<br>$\pm A \not\succeq 0$, $\kappa \gg 1$:<br>MINRES-QLP | *minimum-length solution:*<br>MINRES-QLP<br>LSQR<br>---<br>*least-squares solution:*<br>MINRES<br>SQMR | MINRES-QLP<br>LSQR |
| *square*<br>$A \neq A^*$ | LSQR<br>GMRES<br>QMR<br>Bi-CGSTAB | LSQR | LSQR |
| *rectangular* | LSQR | LSQR | LSQR |

## 6.2 Contributions

This research began with a new approach for null-vector computation, based on least-squares problems and the observation that min $\|Ax - b\|$ gives a residual vector $r = b - Ax$ satisfying $A^T r = 0$.

Applications include eigenvector and singular vector computations, as an *alternative to inverse iteration* when an exact eigenvalue or singular value is known. Iterative solution of the singular least-squares problem converges sooner than when the solver is forced to compute an exploding solution.

The approach extends to computing multiple null vectors, and thus estimating the nullity of a sparse matrix and obtaining an orthogonal null basis (assuming the nullity is relatively small).

Our experimentation with LSQR on singular least-squares problems led us to focus on the behavior of MINRES on singular *symmetric* systems. We realized that MINRES computes the minimum-length solution for singular *compatible* systems $Ax = b$, but not for singular symmetric least-squares problems min $\|Ax - b\|$. The major part of our research became the development of a new algorithm (MINRES-QLP) for this purpose.

MINRES-QLP constructs its $k$th iterate with orthogonal steps: $x_k^Q = (V_k P_k)u_k$. One triangular system $L_k u_k = Q_k(\beta_1 e_1)$ is involved for each $k$, compared to the $n$ systems present in the standard MINRES computation $V_k R_k^{-1}$ (that is, the $n$ lower-triangular systems $R_k^T D_k^T = V_k^T$). Thus MINRES-QLP overcomes the potential instability predicted by the MINRES authors [81] and analyzed by Sleijpen et al. [96]. The additional work and storage are moderate, and maximum efficiency is retained by transferring from MINRES to the MINRES-QLP iterates only when the estimated condition number of $A$ exceeds a specified value.

MINRES and MINRES-QLP are readily applicable to Hermitian matrices, once $\alpha_k$ is typecast as a real scalar in finite-precision arithmetic. For both algorithms, we derived recurrence relations for $\|Ar_k\|$ and $\|Ax_k\|$ and used them to formulate new stopping conditions for singular problems.

TEVD or TSVD are commonly known to use rank-$k$ approximations to $A$ to find approximate solutions to min $\|Ax - b\|$ that serve as a form of *regularization*. Krylov subspace methods also have regularization properties [55, 53, 62]. Since MINRES-QLP monitors more carefully the rank of $T_k$, which could be $k$ or $k - 1$, we may say that regularization is a stronger feature in MINRES-QLP, as we have shown in our numerical examples.

## 6.3 Ongoing Work

We hope to study more in depth the error and convergence analysis for MINRES and MINRES-QLP in the fashion of [96]. Specifically, their $k$th iterates $x_k^M = (V_k R_k^{-1})t_k$ and $x_k^Q = (V_k P_k)u_k$ give some hints on better rounding-error properties of MINRES-QLP. The question remains whether MINRES-QLP is capable of delivering the level of accuracy in finite precision as expected from the norm-wise relative backward errors.

Like all solvers, MINRES-QLP is challenged by very ill-conditioned least-squares problems, including the weighted least-squares problems studied in Bobrovnikova and Vavasis [14]. The difficulties are more pronounced for matrices whose numerical rank is ill-determined. Regularization schemes [54], selective and partial reorthogonalization [85, 92, 93, 65] remain helpful. These are certainly active areas of research not restricted to symmetric problems.

Our approach in extending MINRES to MINRES-QLP may be applied to existing iterative algorithms such as GMRES and QMR for unsymmetric singular least-squares problems. Clearly, both methods need efficient estimates of $\|A^T r_k\|$ for singular least-squares problems. If GMRES without restarting is practical (in terms of memory), a modified version GMRES-QLP could compute QLP factors of the Arnoldi Hessenberg matrix at the final iteration $k$:

$$Q_k \underline{H_k} = \underline{R_k} = \begin{bmatrix} R_k \\ 0 \end{bmatrix}, \qquad R_k P_k = L_k.$$

This would probably reveal rank better and thus give a more regularized solution $x_k = V_k P_k u_k$ with $u_k(k) = 0$. But if restarting is necessary after $m$ steps (where $m$ cannot be too large), the Hessenberg matrix $\underline{H_m}$ need not be singular nor ill-conditioned (that is, not rank revealing), in which case QLP factors of $\underline{H_m}$ may not be helpful.

We are also interested in studying the convergence of $\|A\bar{w}_k\|$ for singular $A$ and $b \notin \mathcal{R}(A)$ in SYMMLQ. This has applications in null-vector computations.

As pointed out by Larsen [66], least-squares problems with multiple right-hand sides are less studied than linear systems with multiple right-hand sides. We would like to pursue the design of efficient algorithms for large least-squares problems with multiple right-hand sides, as suggested by Larsen in connection with his PROPACK software [65]. Such algorithms could be applied to large-scale multiple null-vector problems.

LSQR has a connection to partial least squares (PLS) [32]. We can expect similar characteristics of MINRES-QLP in the symmetric case.

Lastly, it would be ideal to estimate upper and lower bounds for the error norm $||x - x_k||$ in MINRES and MINRES-QLP using moments and quadrature techniques following the series of work published by Golub on symmetric positive definite matrices [44, 45, 48].

# Bibliography


[1] M. A. Amer. Constructive solutions for nonlinear multiparameter eigenvalue problems. *Comput. Math. Appl.*, 35(11):83–90, 1998.

[2] M. Arioli, I. Duff, and D. Ruiz. Stopping criteria for iterative solvers. *SIAM J. Matrix Anal. Appl.*, 13(1):138–144, 1992.

[3] W. E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.*, 9:17–29, 1951.

[4] F. V. Atkinson. *Multiparameter Eigenvalue Problems*. Academic Press, New York, 1972.

[5] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems*. SIAM, Philadelphia, PA, 2000.

[6] Z.-Z. Bai and G.-Q. Li. Restrictively preconditioned conjugate gradient methods for systems of linear equations. *IMA J. Numer. Anal.*, 23(4):561–580, 2003.

[7] A. Ben-Israel and T. N. E. Greville. *Generalized Inverses: Theory and Applications*. Springer-Verlag, New York, second edition, 2003.

[8] M. Benzi. Preconditioning techniques for large linear systems: a survey. *J. Comput. Phys.*, 182(2):418–477, 2002.

[9] J. Berns-Müller, I. G. Graham, and A. Spence. Inexact inverse iteration for symmetric matrices. *Linear Algebra Appl.*, 416(2-3):389–413, 2006.

[10] Å. Björck. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, PA, 1996.

[11] Å. Björck. Bidiagonal decomposition and statistical computing. Presented at the 15th International Workshop on Matrices and Statistics, University of Uppsala, Sweden, 2006.

[12] Å. Björck and T. Elfving. Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations. *BIT*, 19(2):145–163, 1979.

[13] Å. Björck, T. Elfving, and Z. Strakoš. Stability of conjugate gradient and Lanczos methods for linear least squares problems. *SIAM J. Matrix Anal. Appl.*, 19(3):720–736, 1998.

[14] E. Y. Bobrovnikova and S. A. Vavasis. Accurate solution of weighted least squares by iterative methods. *SIAM J. Matrix Anal. Appl.*, 22(4):1153–1174, 2001.

[15] D. Boley. Computing rank-deficiency of rectangular matrix pencils. *Systems Control Lett.*, 9(3):207–214, 1987.

[16] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW7: Proceedings of the Seventh International Conference on World Wide Web*, pages 107–117. Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 1998.

[17] P. N. Brown and H. F. Walker. GMRES on (nearly) singular systems. *SIAM J. Matrix Anal. Appl.*, 18(1):37–51, 1997.

[18] J. C. Browne, M. Yalamanchi, K. Kane, and K. Sankaralingam. General parallel computations on desktop grid and p2p systems. In *LCR '04: Proceedings of the 7th Workshop on Languages, Compilers, and Run-Time Support for Scalable Systems*, pages 1–8. ACM Press, New York, NY, USA, 2004.

[19] P. Businger and G. H. Golub. Linear least squares solutions by Householder transformations. *Numer. Math.*, 7:269–276, 1965.

[20] S.-L. Chang and C.-S. Chien. A multigrid-Lanczos algorithm for the numerical solutions of nonlinear eigenvalue problems. *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, 13(5):1217–1228, 2003.

[21] D. Chen and S. Toledo. Combinatorial characterization of the null spaces of symmetric H-matrices. *Linear Algebra Appl.*, 392:71–90, 2004.

[22] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61, 1998.

[23] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.

[24] Y. T. Chen. Iterative methods for linear least-squares problems. Res. Rep. CS-75-04, Department of Computer Science, University of Waterloo, ON, Canada, 1975.

[25] S.-C. Choi, D. L. Donoho, A. G. Flesia, X. Huo, O. Levi, and D. Shi. About Beamlab—a toolbox for new multiscale methodologies. http://www-stat.stanford.edu/~beamlab/, 2002.

[26] CiteSeer.IST Scientific Digital Library. http://citeseer.ist.psu.edu/.

[27] J. Claerbout. Hypertext documents about reproducible research. http://sepwww.stanford.edu.

[28] J.-P. Dedieu and F. Tisseur. Perturbation theory for homogeneous polynomial eigenvalue problems. *Linear Algebra Appl.*, 358:71–94, 2003.

[29] F. A. Dul. MINRES and MINERR are better than SYMMLQ in eigenpair computations. *SIAM J. Sci. Comput.*, 19(6):1767–1782, 1998.

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

[31] L. Eldén. The eigenvalues of the Google matrix. Tech. Rep. LiTHMAT-R-04-01, Linköping University, Linköping, Sweden, 2004.

[32] L. Eldén. Partial least-squares vs. Lanczos bidiagonalization. I. Analysis of a projection method for multiple regression. *Comput. Statist. Data Anal.*, 46(1):11–31, 2004.

[33] D. K. Faddeev and V. N. Faddeeva. *Computational Methods of Linear Algebra*. Translated by Robert C. Williams. W. H. Freeman and Co., San Francisco, 1963.

[34] B. Fischer, A. Ramage, D. J. Silvester, and A. J. Wathen. Minimum residual methods for augmented systems. *BIT*, 38(3):527–543, 1998.

[35] R. Fletcher. Conjugate gradient methods for indefinite systems. In *Numerical Analysis (Proc 6th Biennial Dundee Conf., Univ. Dundee, Dundee, 1975)*, pages 73–89. Lecture Notes in Math., Vol. 506. Springer, Berlin, 1976.

[36] R. W. Freund. A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems. *SIAM J. Sci. Comput.*, 14(2):470–482, 1993.

[37] R. W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-Hermitian linear systems. *Numer. Math.*, 60(3):315–339, 1991.

[38] R. W. Freund and N. M. Nachtigal. A new Krylov-subspace method for symmetric indefinite linear systems. In W. F. Ames, editor, *Proceedings of the 14th IMACS World Congress on Computational and Applied Mathematics*, pages 1253–1256. IMACS, 1994.

[39] K. A. Gallivan, S. Thirumalai, P. Van Dooren, and V. Vermaut. High performance algorithms for Toeplitz and block Toeplitz matrices. In *Proceedings of the Fourth Conference of the International Linear Algebra Society (Rotterdam, 1994)*, volume 241/243, pages 343–388, 1996.

[40] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders. Preconditioners for indefinite systems arising in optimization. *SIAM J. Matrix Anal. Appl.*, 13(1):292–311, 1992.

[41] P. E. Gill, W. Murray, and M. H. Wright. *Numerical Linear Algebra and Optimization. Vol. 1.* Addison-Wesley Publishing Company Advanced Book Program, Redwood City, CA, 1991.

[42] D. Gleich, L. Zhukov, and P. Berkhin. Fast parallel PageRank: A linear system approach. Technical Report YRL-2004-038, Yahoo! Research Labs, 2004.

[43] G. H. Golub. Numerical methods for solving linear least squares problems. *Numer. Math.*, 7:206–216, 1965.

[44] G. H. Golub. Matrix computation and the theory of moments. In *Proceedings of the International Congress of Mathematicians, Vol. 1, 2 (Zürich, 1994)*, pages 1440–1448. Birkhäuser, Basel, 1995.

[45] G. H. Golub. Matrix computation and the theory of moments. *Bull. Belg. Math. Soc. Simon Stevin*, suppl.:1–9, 1996. Numerical analysis (Louvain-la-Neuve, 1995).

[46] G. H. Golub and C. Greif. An Arnoldi-type algorithm for computing PageRank. *BIT Numerical Mathematics*, published online (Springerlink), 2006.

[47] G. H. Golub and W. M. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *J. Soc. Indust. Appl. Math. Ser. B Numer. Anal.*, 2:205–224, 1965.

[48] G. H. Golub and G. Meurant. Matrices, moments and quadrature. II. How to compute the norm of the error in iterative methods. *BIT*, 37(3):687–705, 1997.

[49] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, second edition, 1989.

[50] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[51] M. H. Gutknecht and M. Rozložnik. Residual smoothing techniques: do they improve the limiting accuracy of iterative solvers? *BIT*, 41(1):86–114, 2001.

[52] W. W. Hager. Iterative methods for nearly singular linear systems. *SIAM J. Sci. Comput.*, 22(2):747–766, 2000.

[53] M. Hanke and J. G. Nagy. Restoration of atmospherically blurred images by symmetric indefinite conjugate gradient techniques. *Inverse Problems*, 12(2):157–173, 1996.

[54] P. C. Hansen. Truncated singular value decomposition solutions to discrete ill-posed problems with ill-determined numerical rank. *SIAM J. Sci. Statist. Comput.*, 11(3):503–518, 1990.

[55] P. C. Hansen and D. P. O'Leary. The use of the L-curve in the regularization of discrete ill-posed problems. *SIAM J. Sci. Comput.*, 14(6):1487–1503, 1993.

[56] R. J. Hanson and C. L. Lawson. Extensions and applications of the Householder algorithm for solving linear least squares problems. *Math. Comp.*, 23:787–812, 1969.

[57] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436, 1952.

[58] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, PA, second edition, 2002.

[59] N. J. Higham and F. Tisseur. More on pseudospectra for polynomial eigenvalue problems and applications in control theory. *Linear Algebra Appl.*, 351/352:435–453, 2002.

[60] I. C. F. Ipsen and C. D. Meyer. The idea behind Krylov methods. *Amer. Math. Monthly*, 105(10):889–899, 1998.

[61] E. F. Kaasschieter. Preconditioned conjugate gradients for solving singular systems. *J. Comput. Appl. Math.*, 24(1-2):265–275, 1988.

[62] M. Kilmer and G. W. Stewart. Iterative regularization and MINRES. *SIAM J. Matrix Anal. Appl.*, 21(2):613–628, 1999.

[63] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Research Nat. Bur. Standards*, 45:255–282, 1950.

[64] A. N. Langville and C. D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, Princeton, NJ, 2006.

[65] R. M. Larsen. PROPACK downloadable software. http://soi.stanford.edu/~rmunk/PROPACK/index.html.

[66] R. M. Larsen. *Efficient Algorithms for Helioseismic Inversion*. PhD thesis, Dept of Computer Science, University of Aarhus, 1998.

[67] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide*. SIAM, Philadelphia, PA, 1998.

[68] A. Y. T. Leung. Inverse iteration for the quadratic eigenvalue problem. *J. Sound Vibration*, 124(2):249–267, 1988.

[69] J. G. Lewis. *Algorithms for Sparse Matrix Eigenvalue Problems*. PhD thesis, Dept of Computer Science, Stanford University, 1976.

[70] O. E. Livne and G. H. Golub. Scaling by binormalization. *Numer. Algorithms*, 35(1):97–120, 2004.

[71] C. B. Moler. *Numerical Computing with MATLAB*. SIAM, Philadelphia, PA, 2004.

[72] R. B. Morgan. Computing interior eigenvalues of large matrices. *Linear Algebra Appl.*, 154/156:289–309, 1991.

[73] M. F. Murphy, G. H. Golub, and A. J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.*, 21(6):1969–1972, 2000.

[74] M. G. Neytcheva and P. S. Vassilevski. Preconditioning of indefinite and almost singular finite element elliptic equations. *SIAM J. Sci. Comput.*, 19(5):1471–1485, 1998.

[75] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, second edition, 2006.

[76] Y. Notay. Solving positive (semi)definite linear systems by preconditioned iterative methods. In *Preconditioned Conjugate Gradient Methods (Nijmegen, 1989)*, volume 1457 of *Lecture Notes in Math.*, pages 105–125. Springer, Berlin, 1990.

[77] Y. Notay. Combination of Jacobi-Davidson and conjugate gradients for the partial symmetric eigenproblem. *Numer. Linear Algebra Appl.*, 9(1):21–44, 2002.

[78] C. C. Paige. Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix. *J. Inst. Math. Appl.*, 18(3):341–349, 1976.

[79] C. C. Paige. Krylov subspace processes, Krylov subspace methods, and iteration polynomials. In *Proceedings of the Cornelius Lanczos International Centenary Conference (Raleigh, NC, 1993)*, pages 83–92. SIAM. Philadelphia, PA, 1994.

[80] C. C. Paige, M. Rozložník, and Z. Strakoš. Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES. *SIAM J. Matrix Anal. Appl.*, 28(1):264–284, 2006.

[81] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[82] C. C. Paige and M. A. Saunders. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8(1):43–71, 1982.

[83] C. C. Paige and M. A. Saunders. algorithm 583; LSQR: Sparse linear equations and least-squares problems. *ACM Trans. Math. Software*, 8(2):195–209, 1982.

[84] B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM, Philadelphia, PA, 1998.

[85] B. N. Parlett and D. S. Scott. The Lanczos algorithm with selective orthogonalization. *Math. Comp.*, 33(145):217–238, 1979.

[86] W.-q. Ren and J.-x. Zhao. Iterative methods with preconditioners for indefinite systems. *J. Comput. Math.*, 17(1):89–96, 1999.

[87] A. Ruhe and T. Wiberg. The method of conjugate gradients used in inverse iteration. *Nordisk Tidskr. Informationsbehandling (BIT)*, 12:543–554, 1972.

[88] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, PA, second edition, 2003.

[89] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, 1986.

[90] M. A. Saunders. Solution of sparse rectangular systems using LSQR and Craig. *BIT*, 35(4):588–604, 1995.

[91] M. A. Saunders. Computing projections with LSQR. *BIT*, 37(1):96–104, 1997.

[92] H. D. Simon. Analysis of the symmetric Lanczos algorithm with reorthogonalization methods. *Linear Algebra Appl.*, 61:101–131, 1984.

[93] H. D. Simon. The Lanczos algorithm with partial reorthogonalization. *Math. Comp.*, 42(165):115–142, 1984.

[94] M. Sipser. *Introduction to the Theory of Computation*. PWS Publishing Company, Boston, MA, 1997.

[95] G. L. G. Sleijpen, A. G. L. Booten, D. R. Fokkema, and H. A. Van der Vorst. Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT*, 36(3):595–633, 1996.

[96] G. L. G. Sleijpen, H. A. Van der Vorst, and J. Modersitzki. Differences in the effects of rounding errors in Krylov solvers for symmetric indefinite linear systems. *SIAM J. Matrix Anal. Appl.*, 22(3):726–751, 2000.

[97] Systems Optimization Laboratory (SOL), Stanford University, downloadable software: AMRES, CGLS, LSQR, LUMOD, MINRES, MINRES-QLP, PDCO, PDSCO, SYMMLQ. http://www.stanford.edu/group/SOL/software.html.

[98] P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 10(1):36–52, 1989.

[99] G. W. Stewart. Perturbation theory for rectangular matrix pencils. *Linear Algebra Appl.*, 208/209:297–301, 1994.

[100] G. W. Stewart. The QLP approximation to the singular value decomposition. *SIAM J. Sci. Comput.*, 20(4):1336–1348, 1999.

[101] D. B. Szyld and O. B. Widlund. Applications of conjugate gradient type methods to eigenvalue calculations. In *Advances in Computer Methods for Partial Differential Equations, III (Proc. Third IMACS Internat. Sympos., Lehigh Univ., Bethlehem, Pa., 1979)*, pages 167–173. IMACS, New Brunswick, N.J., 1979.

[102] R. C. Thompson. Principal submatrices. IX. Interlacing inequalities for singular values of submatrices. *Linear Algebra and Appl.*, 5:1–12, 1972.

[103] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B*, 58(1):267–288, 1996.

[104] F. Tisseur. Backward error and condition of polynomial eigenvalue problems. In *Proceedings of the International Workshop on Accurate Solution of Eigenvalue Problems (University Park, PA, 1998)*, volume 309, pages 339–361, 2000.

[105] K.-C. Toh, K.-K. Phoon, and S.-H. Chan. Block preconditioners for symmetric indefinite linear systems. *Internat. J. Numer. Methods Engrg.*, 60(8):1361–1381, 2004.

[106] J. A. Tomlin. A new paradigm for ranking pages on the world wide web. In *Proceedings of the World Wide Web conference 2003 (WWW2003)*, pages 350–355, May 2003.

[107] L. N. Trefethen and D. Bau, III. *Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.

[108] University of Florida sparse matrix collection. http://www.cise.ufl.edu/research/sparse/matrices/.

[109] H. A. Van der Vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 13(2):631–644, 1992.

[110] H. A. Van der Vorst. *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press, Cambridge, 2003.

[111] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Rev.*, 38(1):49–95, 1996.

[112] S. Varadhan, M. W. Berry, and G. H. Golub. Approximating dominant singular triplets of large sparse matrices via modified moments. *Numer. Algorithms*, 13(1-2):123–152, 1996.

[113] C.-h. Yu and O. Axelsson. A process for solving a few extreme eigenpairs of large sparse positive definite generalized eigenvalue problem. *J. Comput. Math.*, 18(4):387–402, 2000.

[114] H.-G. Yu and G. Nyman. A spectral transform minimum residual filter diagonalization method for interior eigenvalues of physical systems. *J. Chem. Phys.*, 110(23):11133–11140, 1999.

[115] J. Y. Yuan. Numerical methods for generalized least squares problems. In *Proceedings of the Sixth International Congress on Computational and Applied Mathematics (Leuven, 1994)*, volume 66, pages 571–584, 1996.

[116] Y. Zhang. Solving large-scale linear programs by interior-point methods under the MATLAB environment. *Optim. Methods Softw.*, 10(1):1–31, 1998.

[117] T. S. Zheng, W. M. Liu, and Z. B. Cai. A generalized inverse iteration method for solution of quadratic eigenvalue problems in structural dynamic analysis. *Comput. & Structures*, 33(5):1139–1143, 1989.