ABSTRACT

Krylov Methods for Solving a Sequence of Large Systems of Linear Equations

Huy V. Nguyen, Ph.D.

Advisor: Ronald B. Morgan, Ph.D.

Consider solving a sequence of linear systems

$$A_{(i)}x^{(i)} = b^{(i)}, i = 1, 2, ...$$

where $A_{(i)} \in \mathbb{C}^{n\times n}$ and $b^{(i)} \in \mathbb{C}^n$ using some variations of Krylov subspace methods, like GMRES. For a single system $Ax = b$, it is well-known that the eigenvectors of the coefficient matrix A can be used to speed up the convergence of GMRES by deflating the corresponding eigenvalues.

In this dissertation, we propose a deflation-based algorithm that utilizes the eigenvalue and eigenvector information obtained from one system to improve the convergence of GMRES for solving the subsequent systems. When the change in the system is small enough, the algorithm will REUSE the eigenvectors from the previous system to deflate the small eigenvalues from the new system via a projection to speed up convergence. When the change is significant enough that projection loses effectiveness, the algorithm will RECYCLE the eigenvectors from the previous system by adding them to the new Krylov subspace, thus improving them so that they can be suitable candidates for deflation once again. If the system has changed too much, or the new system is completely unrelated to the previous system, the algorithm will REGENERATE a new set of eigenvectors to help with deflation.

Krylov Methods for Solving a Sequence of Large Systems of Linear Equations

by

Huy V. Nguyen, B.S., M.S.

A Dissertation

Approved by the Department of Mathematics

_____

Lance L. Littlejohn, Ph.D., Chairperson

Submitted to the Graduate Faculty of
Baylor University in Partial Fulfillment of the
Requirements for the Degree
of
Doctor of Philosophy

Approved by the Dissertation Committee

_____

Ronald B. Morgan, Ph.D., Chairperson

_____

John M. Davis, Ph.D.

_____

Joseph Kuehl, Ph.D.

_____

Frank H. Mathis, Ph.D.

_____

Qin Sheng, Ph.D.

Accepted by the Graduate School
August 2015

_____

J. Larry Lyon, Ph.D., Dean

Page bearing signatures is kept on file in the Graduate School.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank the Department of Mathematics at Baylor University for the generous financial support, and for the invaluable opportunity to learn from some of the very best in the field. Special thanks to Dr. Lance Littlejohn and Dr. Qin Sheng for introducing me to the possibility of a career in mathematics, and for inspiring me to become a math major as an undergraduate.

To Dr. Ronald B. Morgan, words cannot express my deepest gratitude. You have been more than a mentor, an advisor, and a teacher to me. Thank you for giving me the privilege to learn from you, for showing me how to aim higher, work harder, and persevere; thank you for your countless number of advice and your inexhaustible amount of patience; and most importantly, thank you for believing in me. I could not have done this without you.

To my previous teaching mentors – Dr. Ed Oxford and Dr. Jonatan Lenells – thank you for your guidance during my early teaching career. To the members of the dissertation committee, thank you for your extremely helpful feedback and suggestions. To Dr. Mark Sepanski, Dr. John Davis, Dr. Klaus Kirsten, and Dr. Robert Kirby, thank you for challenging me to push myself beyond the edge of my abilities. To the wonderful secretaries – Mrs. Dees, Mrs. Massey, and Mrs. Salinas – thank you for all of your hard work and your kindness.

Lastly, I would like to thank my family – especially my mother, my brother, and my grandparents – for their love and constant support. To all of my friends and the brothers of Kappa Kappa Psi - Beta Alpha, thank you for making my journey at Baylor University an absolutely extraordinary experience.

CHAPTER ONE

Introduction

Differential Equations are arguably some of the greatest tools we have to model our natural world, from physics, to neuroscience, to finance, and to various fields of engineering. The solutions to these equations offer great insights into how each process works. However, finding exact, analytic solutions to differential equations is often difficult, and sometimes impractical if we need to use them for any actual computations. In those cases, we turn to numerical solutions. The idea is to convert a continuous problem into a discrete problem whose solutions can be good approximations to the analytic solutions of the continuous problem. This is called a *discretization* process. There are many different ways to discretize a differential equation such as Finite Difference or Finite Elements. In most cases, they end up requiring the solution of a system of linear equations, sometimes many systems of linear equations. In order to achieve high accuracy, a large number data points are usually used, which results in very large systems of linear equations.

There are many methods developed to solve systems of linear equations: from direct solvers like Gaussian Elimination to classical iterative solvers like Jacobi method. For very large systems of linear equations, some of the most popular choices belong to a class of iterative methods called Krylov subspace methods, including: CG, BGC, FOM, GMRES, QMR, BICGSTAB, TFQMR... In Chapter Two, we mainly focus on variations of GMRES – the Generalized Minimum Residual Method. We will describe the algorithm for GMRES and include some convergence analysis to show how eigenvalues affect the convergence of GMRES for solving linear equations. We discuss the negative impact of having small eigenvalues and demonstrate how to address that issue via two deflation-based approaches: GMRES-E and GMRES-DR.

We also briefly show two procedures to compute eigenvalues and eigenvectors while solving the linear equations.

In Chapter Three, we show an approach of reusing eigenvectors from one system to speed up the convergence of GMRES for solving systems with the same coefficient matrix and multiple right-hand sides called GMRES-Proj. When the matrix and the right-hand side are both changing from one system to the next, we use an updated version of GMRES-E that has the capability to recycle eigenvectors from a previous system to make them more suitable for deflation in the current system. We call this GMRES-E(recycled).

We combine GMRES-Proj, GMRES-E(recycled), and GMRES-DR into a new algorithm, GMRES-RRR, that can efficiently solve a sequence of linear systems, switching between methods to take full advantage of what each method does best: GMRES-Proj to REUSE the eigenvectors, GMRES-E(recycled) to RECYCLE the eigenvectors, and GMRES-DR to REGENERATE new eigenvectors. We also provide some guidelines to help the user choose the switching conditions. We provide numerical results to demonstrate the effectiveness of the new algorithm.

In Chapter Four, we compare GMRES-RRR to one of the more standard approaches of subspace recycling, GCRO-DR, to show that GMRES-RRR can perform just as competitively, and sometimes with a slight advantage due to its speed and simple implementation.

In Chapter Five, we discuss some further considerations based on previous work. We also mention some challenges that GMRES-E(recycled) may face, and offer some suggestions for improvement.

CHAPTER TWO

Preliminaries

2.1 Krylov Methods for Solving Linear Equations

There are many well-known methods for computing the solution of a system of linear equations of the form $Ax = b$. For a reasonably small matrix, a direct solver, such as Gaussian Elimination, can be applied; however, as the size of the matrix increases, the computational cost also increases. When using a direct solver becomes impractical, one must employ an iterative method, where one chooses and initial guess, computes an approximation, checks the accuracy, and repeats if necessary. Some of the classical iterative methods include Jacobi, Gauss-Seidel, and Successive-Over-Relaxation Methods [26].

Each of these methods has its own disadvantages and limitations, such as slow convergence often seen in Jacobi method. The negative impacts of these disadvantages become even more significant as the size of the systems increases. To avoid the some of the shortcomings of classical iterative methods, especially for very large systems, many new methods were developed. Some of the most popular choices belong to a class of iterative methods called Krylov Subspace Methods, named after the Russian mathematician, Alexei Krylov.

The basis of these methods comes from the idea that the approximate solution to the system of linear equations $Ax = b$ can be found in a *Krylov subspace* of the form

$$\mathcal{K}_m(A, b) = span\{b, Ab, A^2b, A^3b, ..., A^{m-1}b\} \tag{2.1}$$

which is sometimes referred to as a *search subspace* of dimension m. The most simplistic algorithm for solving $Ax = b$ using a Krylov Method can be described in Algorithm 2.1.

Algorithm 2.1 Basic Krylov Method for solving $Ax = b$

1: Choose a value for m.
2: Form the Krylov Subspace of dimension m: $\mathcal{K}_m(A, b)$.
3: Extract the approximate solution $\hat{x}$ from the subspace.
4: Check for convergence, if satisfied then stop, else increase m and go to 2.

As the size of the subspace, m, increases, the cost to build the subspace increases by at least $O(m^2n)$, and the memory cost increases by $O(mn)$ [26]. One way to address this issue is to fix a relatively small value for m and use a restarted version, as described in Algorithm 2.2.

Let x_0 be the initial guess, and the initial residual vector is $r_0 = b - Ax_0$, for implementation purposes, instead of solving $Ax = b$, we will solve the recast problem:

$$b - Ax_0 = r_0$$

$$Ax - Ax_0 = r_0$$

$$A(x - x_0) = r_0$$

which results in a Krylov subspace of the form

$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, A^2r_0, ..., A^{m-1}r_0\} \tag{2.2}$$

from which $x_m \in \mathcal{K}_m(A, r_0)$, the approximate solution to the recast problem $A(x - x_0) = r_0$, can be extracted and used to find $\hat{x} = x_0 + x_m$, the approximate solution to the original problem $Ax = b$.

Algorithm 2.2 Restarted Krylov Method for Solving $Ax = b$

1: Choose an initial guess x_0 and a value for m.
2: Compute $r_0 = b - Ax_0$.
3: Form the Krylov Subspace $\mathcal{K}_m(A, r_0)$.
4: Extract x_m from $\mathcal{K}_m(A, r_0)$ and compute $\hat{x} = x_0 + x_m$.
5: Check for convergence, if satisfied, then stop, else set $x_0 = \hat{x}$, and go to 2.

The details on how to form the Krylov subspace and how to extract the approximate solution will be given in the following sections.

2.1.1 Building Krylov Subspaces: The Arnoldi Algorithm

The Arnoldi algorithm [1] builds an orthogonal basis for the Krylov subspace, $\mathcal{K}_m(A, r_0)$. We will use the Modified Gram-Schmidt version, which is more reliable than the basic version in the presence of round-off error [26].

Algorithm 2.3 Arnoldi with Modified Gram-Schmidt

1: Choose a vector v_1 of norm 1.
2: **for** $j = 1, 2, ..., m$ **do**
3: Compute $w = Av_j$
4: **for** $i = 1, 2, ..., j$ **do**
5: $h_{ij} = (w, v_i)$
6: $w = w - h_{ij}v_i$
7: **end for**
8: $h_{j+1,j} = ||w||$. If $h_{j+1,j} = 0$, Stop.
9: $v_{j+1} = w/h_{j+1,j}$
10: **end for**

This algorithm produces an orthonormal basis $V_m = [v_1, v_2, ..., v_m]$ for $\mathcal{K}_m(A, r_0)$ and an upper Hessenberg matrix $H_m \in \mathbb{C}^{m\times m}$, whose entries below the first subdiagonal are zeros. The non-zero entries are the scalars $h_{i,j}$ from the Arnoldi iterations. Let $\underline{H}_m \in \mathbb{C}^{(m+1)\times m}$ be given by

$$\underline{H}_m = \begin{pmatrix} H_m \\ \\ h_{m+1,m}e_m^T \end{pmatrix}$$

The resulting matrices satisfy the following relations:

$$AV_m = V_{m+1}\underline{H}_m \tag{2.3}$$

$$H_m = V_m^* A V_m \tag{2.4}$$

which are sometimes referred to as the *Arnoldi relations*.

The following Figure 2.1 illustrates the structures and sizes of the matrices produced by the Arnoldi iterations.

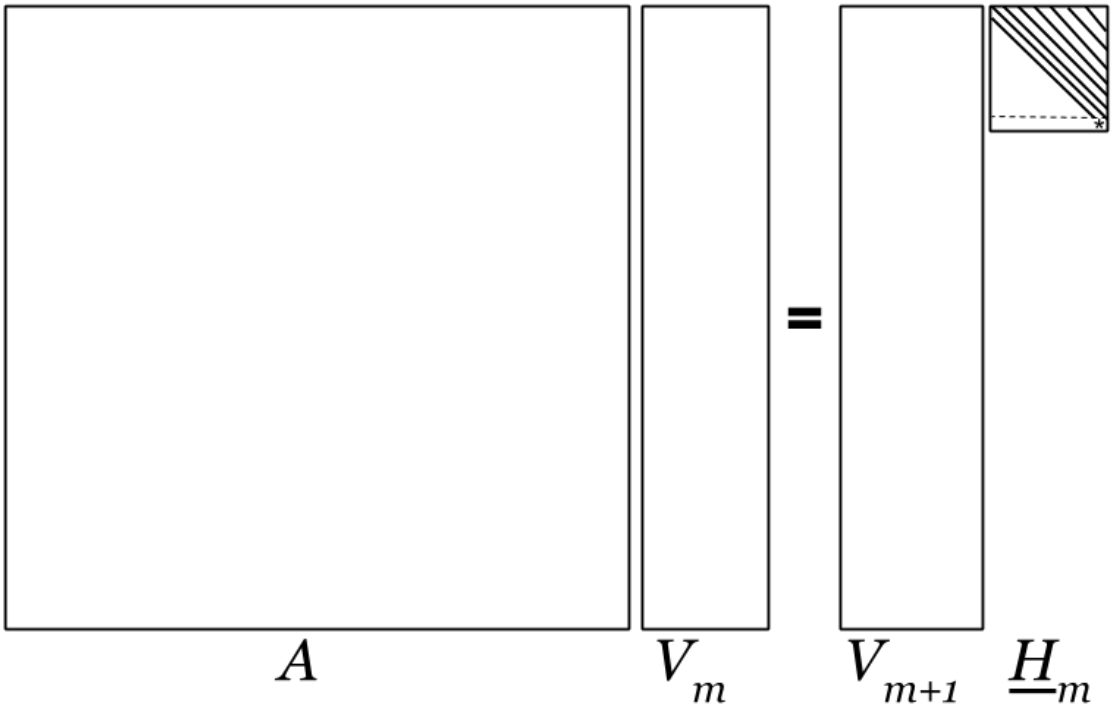


Figure 2.1: Visualizing the Arnoldi relations.

2.1.2 Extracting the Solution: Orthogonal Projections

Next, the method needs to find a vector $x_m \in \mathcal{K}_m(A, r_0)$ such that the norm of the residual vector will be minimized. In order to achieve this, we will utilize the concept of orthogonal projections.

Definition 2.1. Given any vector $b \in \mathbb{C}^n$ and a subspace $\mathcal{S}$, $\hat{b} \in \mathcal{S}$ is an orthogonal projection of b onto $\mathcal{S}$ if and only if $(b - \hat{b}) \perp \mathcal{S}$.

Proposition 2.1. *If $\hat{b}$ is orthogonal projection of b onto $\mathcal{S}$, then*

$$||b - \hat{b}|| = \min_{s \in \mathcal{S}} ||b - s|| \tag{2.5}$$

for all $s \in \mathcal{S}$.

Proof. Let s be any vector of $\mathcal{S}$. Consider the norm squared

$$||b - s||^2 = ||b - \hat{b} + \hat{b} - s||^2$$

For cleanliness, let $u = b - \hat{b}$ and $v = \hat{b} - s$. Note that since $u \perp \mathcal{S}$ and $v \in \mathcal{S}$, we have $u^*v = v^*v = 0$.

$$\begin{aligned} ||b - s||^2 &= ||u + v||^2 \\ &= (u + v)^*(u + v) \\ &= u^*u + u^*v + v^*u + v^*v \\ &= ||u||^2 + 0 + 0 + ||v||^2 \\ &= ||b - \hat{b}||^2 + ||\hat{b} - s||^2 \end{aligned}$$

Since $||\hat{b} - s||^2 \geq 0$, we have

$$||b - s||^2 \geq ||b - \hat{b}||^2, \text{ for all } s \in \mathcal{S}$$

In other word,

$$||b - \hat{b}|| = \min_{s \in \mathcal{S}} ||b - s||$$

□

By the above proposition, in order to minimize the norm of the residual vector $r = r_0 - Ax_m$, the vector Ax_m must be the orthogonal projection of r_0 onto the space $A\mathcal{K}_m(A, r_0)$, which also means $r \perp A\mathcal{K}_m(A, r_0)$. Since the columns of V_m form an orthonormal basis for $\mathcal{K}_m(A, r_0)$, we can write $x_m = V_m d$ for some $d \in \mathbb{C}^m$. Also, recall from the Arnoldi Algorithm 2.3 that $v_1 = r_0/||r_0||$, we have:

$$\begin{aligned} r \perp A\mathcal{K}_m(A, r_0) \\ (AV_m)^*r = 0 \\ (AV_m)^*(r_0 - Ax_m) = 0 \\ (V_{m+1}\underline{H}_m)^*Ax_m = (V_{m+1}\underline{H}_m)^*r_0 \\ \underline{H}_m^*V_{m+1}^*V_{m+1}\underline{H}_m d = \underline{H}_m^*V_{m+1}^*||r_0||v_1 \\ \underline{H}_m^*\underline{H}_m d = \underline{H}_m^*||r_0||e_1 \end{aligned}$$

Let $\beta = ||r_0||$, solving the normal equation

$$\underline{H}_m^* \underline{H}_m d = \underline{H}_m^* ||r_0|| e_1$$

is equivalent to solving

$$\min_{d \in \mathbb{C}^m} ||\beta e_1 - \underline{H}_m d||$$

It follows that in order to find $x_m \in \mathcal{K}_m(A, r_0)$ that minimizes the norm of the residual vector, the method will need to find $d \in \mathbb{C}^m$, the solution to the $(m+1)$ by m least-square problem $\min_{d \in \mathbb{R}^m} ||\beta e_1 - \underline{H}_m d||$, which can be solved using an orthogonal factorization of H. This is a much smaller problem compared to the original problem as illustrated in Figure 2.1.Furthermore, since $\underline{H}_m$ is already upper-Hessenberg, it is rather inexpensive to transform it to upper-triangular form [26].

2.1.3 Convergence of Krylov Methods: Eigenvalues and Polynomial

In order to monitor the convergence of Krylov Methods for solving linear systems, we look at the norm of the residual vector r given by

$$r = r_0 - Ax_m$$

where $x_m \in span\{r_0, Ar_0, A^2 r_0, A^3 r_0, ..., A^{m-1} r_0\}$ is the approximate solution to the recast problem $A(x - x_0) = r_0$, which can be written as a linear combination of the vectors in the Krylov Subspace

$$x_m = \sum_{i=0}^{m-1} c_i A^i r_0$$

Let $p(\alpha) = \sum_{i=0}^{m-1} c_i \alpha^i$ be a polynomial of degree $m-1$ or less, then

$$x_m = p(A) r_0$$

Define $q(\alpha) = 1 - \alpha p(\alpha)$, then

$$r = r_0 - Ax_m = r_0 - Ap(A)r_0 = (I - Ap(A))r_0 = q(A)r_0$$

where $q(\alpha)$ is a polynomial of degree m or less and $q(0) = 1$. Suppose A has eigenvalues $\{\lambda_1, \lambda_2, ..., \lambda_n\}$ and eigenvectors $\{z_1, z_2, ..., z_n\}$, the right-hand side r_0 can be expanded in terms of the eigenvectors

$$r_0 = \sum_{j=1}^{n} \beta_j z_j$$

As a result, the residual vector can be viewed as

$$r = q(A)r_0 = \sum_{j=1}^{n} \beta_j q(A) z_j = \sum_{j=1}^{n} \beta_j q(\lambda_j) z_j \tag{2.6}$$

In order for the norm of the residual vector to be small, $q(\alpha)$ must be small at each λ_j for $j = 1, 2, ..., n$. Since $q(0) = 1$, this is essentially using $q(\alpha)$ to interpolate $n+1$ points. Hence, when the degree of $q(\alpha)$ is n, the method is guaranteed to converge because $q(\alpha)$ will be able to go through each λ_j, i.e. $q(\lambda_j) = 0$ for all j, thus makes $||r|| = 0$. However, having such a high-degree polynomial corresponds to building a large Krylov subspace, which can be impractical. As a result, for the most part, m must remain relatively small, which means certain matrices will have spectra with favorable properties that would result in fast convergence, and certain spectra would cause slow, or no convergence. Some of the main causes for slow convergence include: matrices that have very (relatively) small eigenvalues, and matrices that have both positive and negative eigenvalues

The formulation of the residual vector in (2.6) gives insight to how the eigenvalues of A affect the convergence of Krylov methods, which provides guidelines to some of the later methods that seek to improve the convergence.

2.1.4 GMRES - Generalized Minimum Residual Method

There are many variations of Krylov subspace methods for solving a system of linear equations. For instance, in the Arnoldi algorithm, instead of Gram-Schmidt, some methods use Householder transforms to create the orthonormal basis. Instead of restarting, some methods use a truncation approach to avoid running out of

storage [26]. Different implementations result in different methods. For the scope of this dissertation, we will focus on restarted variations of the Generalized Minimum Residual Method (GMRES for short) that use the Modified Gram-Schmidt version of the Arnoldi Algorithm.

Algorithm 2.4 Restarted GMRES

1: Choose x_0.
2: Compute $r_0 = b - Ax_0$. Let $v_1 = r_0/||r_0||$, and $\beta = ||r_0||$.
3: **for** $j = 1, 2, ..., m$ **do**
4: Compute $w = Av_j$
5: **for** $i = 1, 2, ..., j$ **do**
6: $h_{ij} = (w, v_i)$
7: $w = w - h_{ij}v_i$
8: **end for**
9: $h_{j+1,j} = ||w||$. If $h_{j+1,j} = 0$, set $m = j$ and go to 12.
10: $v_{j+1} = w/h_{j+1,j}$
11: **end for**
12: Define the $(m + 1) \times m$ Hessenberg matrix $\underline{H}_m$ whose entries are the h_{ij} from the Arnoldi iteration.
13: Compute d that minimizes $||\beta e_1 - \underline{H}_m d||$.
14: Update the approximate solution $\hat{x} = x_0 + V_m d$
15: Check for convergence. If satisfied, then Stop, else set $x_0 = \hat{x}$ and go to 2.

Example 2.1. Effect of Small Eigenvalues - To demonstrate the ill-effect of small eigenvalues on restarted GMRES. Consider solving a 500 by 500 linear system with $A = tridiag(-1, 2, -1)$ and b is a random vector using GMRES(25) and GMRES(400).

$$\begin{pmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{499} \\ x_{500} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{499} \\ b_{500} \end{pmatrix}$$

We choose this particular matrix because it has many relatively small eigenvalues. Table 2.1 shows the smallest 10 eigenvalues compared to the 2 largest in magnitude. The condition number is $\kappa = \frac{\lambda_{500}}{\lambda_1} \approx 1.02 \times 10^5$.

Table 2.1: Smallest ten and largest two eigenvalues of A

Sorted Eigenvalues of A in Order of Magnitude	
λ_1	0.00003932084757
λ_2	0.0001572818442
λ_3	0.0003538783514
λ_4	0.000629102639
λ_5	0.0009829438849
λ_6	0.001415388176
λ_7	0.001926418508
λ_8	0.002516014786
λ_9	0.003184153828
λ_{10}	0.003930809361
$\vdots$	$\vdots$
λ_{499}	3.999842718
λ_{500}	3.999960679

Recall how the residual vector can be expressed as a polynomial $q(\alpha)$ of degree m (2.6). Since $q(0)$ must be 1 and $q(\lambda_i)$ needs to be small for each i, having λ_1 so close to 0 while the rest of the eigenvalues so spread out requires the polynomial to have a high enough degree in order to be effective. Since the degree of the polynomial is fixed in restarted methods, even though GMRES(25) will find the best degree 25 polynomial to minimize the norm of the residual vector at the end of each cycle, it is simply not good enough, which results in a very slow convergence rate. On the other hand, GMRES(400) uses a large enough subspace that it is able to reach convergence. Figure 2.2 demonstrate the difference between the convergence of GMRES(25) and GMRES(400). However, $m = 400$ relatively close to the actual size of the original problem, $n = 500$, which can be impractical especially if n is very large.

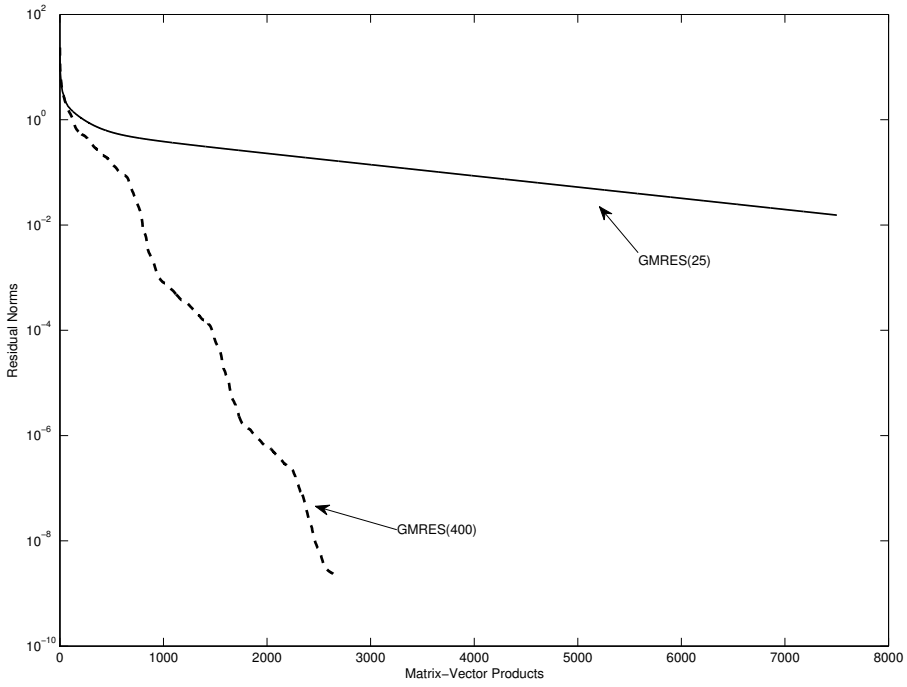


Figure 2.2: Comparing the convergence of GMRES(25) and GMRES(400).

2.2 Krylov Methods for Eigenvalue Problems

Before showing how eigenvalues and eigenvectors can help improving the convergence of GMRES, we will briefly discuss a few approaches to extract the approximate eigenvalues and eigenvectors of a matrix from a Krylov subspace.

2.2.1 Rayleigh-Ritz Procedure

In general, given a matrix $A \in \mathbb{C}^{n\times n}$ and a subspace $\mathcal{K}$ of dimension m, we want to find a vector $y_i \in \mathcal{K}$ and some $\theta_i \in \mathbb{C}$ such that θ_i and y_i are approximations of λ_i and z_i, respectively, where (λ_i, z_i) is an eigenpair of A, i.e. $Az_i = \lambda_i z_i$.

Define the residual of an eigenvector

$$r_i = Ay_i - \theta_i y_i \tag{2.7}$$

In order for (θ_i, y_i) to be a good approximation as an eigenpair of A, the norm of r_i must be small. Using the technique of orthogonal projection, as shown in Proposition 2.1, we see that $||Ay_i - \theta_i y_i||$ is minimized when

$$(Ay_i - \theta_i y_i) \perp \mathcal{K} \tag{2.8}$$

Let $V \in \mathbb{C}^{n\times m}$ be a matrix whose columns form an orthonormal basis of $\mathcal{K}$, then we can write $y_i = Vg_i$ for some $g_i \in \mathbb{C}^m$ and

$$\begin{aligned} V^*(Ay_i - \theta_i y_i) &= 0 \\ V^*AVg_i - \theta_i V^*Vg_i &= 0 \\ V^*AVg_i &= \theta_i g_i \end{aligned}$$

Define $H_m = V^*AV$, it follows that (θ_i, g_i) is an eigenpair of H_m. So, in order to find approximate eigenvectors of A from the subspace $\mathcal{K}$, all we have to do is solve the eigenvalue problem

$$H_m g = \theta g$$

This is a much easier problem since H_m is an m by m matrix, which is relatively small compared to A. Once we have found θ_i's and g_i's, the eigenvalues and eigenvectors of H_m, we can get θ_i's and $y_i = Vg_i$'s, approximate eigenvalues and approximate eigenvectors of A. θ_i's and y_i's are also called Ritz values and Ritz vectors, respectively. This is essentially the idea behind the Rayleigh-Ritz procedure in Algorithm 2.5. When used in conjunction with a Krylov method, such as GMRES, the orthonormal basis, the matrices V and H_m are already available from the Arnoldi iteration, which allows approximate eigenvalues and eigenvectors to be computed without incurring much additional expenses.

Algorithm 2.5 Rayleigh-Ritz Procedure

1: Find an orthonormal basis $\{v_1, v_2, ..., v_m\}$ for $\mathcal{K}$.
2: Let $V = [v_1 v_2 ... v_m]$.
3: Form the matrix $H = V^*AV$
4: Find θ_i's and g_i's, the eigenvalues and eigenvectors of H
5: θ_i's and $y_i = Vg_i$'s are approximate eigenvalues and eigenvectors of A

Without going into much details, we will show that in general, if the subspace $\mathcal{K}$ contains an exact eigenvector, the Rayleigh-Ritz procedure will extract it.

Proposition 2.2. *Suppose z_1 is an eigenvector of A with corresponding eigenvalue λ_1. If $z_1 \in \mathcal{K}$, then one of the Ritz pairs produced by the Rayleigh-Ritz procedure will be (λ_1, z_1).*

Proof. Let $V = [v_1 v_2 ... v_m]$ be the matrix whose columns form an orthonormal basis for $\mathcal{K}$ where $v_1 = z_1$ and $||z_1|| = 1$. Consider the matrix H in the Rayleigh-Ritz procedure

$$H = V^*AV = \begin{pmatrix} v_1^*Av_1 & v_1^*Av_2 & ... & & v_1^*Av_m \\ v_2^*Av_1 & v_2^*Av_2 & v_2^*Av_3 & ... & v_2^*Av_m \\ v_3^*Av_1 & v_3^*Av_2 & v_3^*AV_3 & ... & v_3^*Av_m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_m^*Av_1 & v_m^*Av_2 & ... & ... & v_m^*A_m \end{pmatrix}$$

Consider the entries of the first column of H

$$h_{11} = v_1^* A v_1 = z_1^* A z_1 = z_1^*(\lambda_1 z_1) = \lambda_1 z_1^* z_1 = \lambda_1$$

$$h_{i1} = v_i^* A v_1 = v_i^* A z_1 = v_i^*(\lambda_1 z_1) = \lambda_1 v_i^* v_1 = 0 \text{ for } i = 2, ..., m.$$

So, H actually looks like

$$H = \begin{pmatrix} \lambda_1 & * & * & ... & * \\ 0 & * & * & ... & * \\ 0 & * & * & ... & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & * & * & ... & * \end{pmatrix}$$

Denote the length m vector $e_1 = (\ 1 \quad 0 \quad 0 \quad ... \quad 0\ )^T$, we see that

$$He_1 = \lambda_1 e_1$$

which means λ_1 is an eigenvalue of H with corresponding eigenvector e_1. The Rayleigh-Ritz procedure yields λ_1 as the Ritz value of A with corresponding Ritz vector $Ve_1 = v_1 = z_1$. □

2.2.2 Harmonic Rayleigh-Ritz Procedure

The Rayleigh-Ritz procedure is generally more reliable at extracting exterior eigenvalues [21] but not for interior eigenvalues [11], which we might need in order to speed up the convergence of GMRES. To compute interior eigenvalues, we will use a modified version of the Rayleigh-Ritz procedure that can extract approximate eigenvalues near any given value σ. Let V be the matrix whose columns span the subspace $\mathcal{K}$ from which approximate eigenvectors are to be extracted. The modified procedure solves the eigenvalue problem

$$V^*(A - \sigma I)^* V g = \frac{1}{\widetilde{\theta} - \sigma} V^*(A - \sigma I)^*(A - \sigma I) V g \tag{2.9}$$

for eigenpairs $\left(\frac{1}{\widetilde{\theta}_i - \sigma}, (A-\sigma I)Vg_i\right)$'s of the matrix $(A-\sigma I)^{-1}$. These, in turns, will give the corresponding eigenpairs $\left(\widetilde{\theta}_i, (A-\sigma I)Vg_i\right)$ of the matrix A [11].

Note that $(A-\sigma I)Vg_i$ is equivalent to Vg_i for each i since

$$\begin{aligned}
A\big((A-\sigma I)Vg_i\big) &= \widetilde{\theta}_i(A-\sigma I)Vg_i \\
(A-\sigma I)^{-1}A(A-\sigma I)Vg_i &= \widetilde{\theta}_i(A-\sigma I)^{-1}(A-\sigma I)Vg_i \\
(A-\sigma I)^{-1}(A^2-\sigma A)Vg_i &= \widetilde{\theta}_i Vg_i \\
(A-\sigma I)^{-1}(A-\sigma I)AVg_i &= \widetilde{\theta}_i Vg_i \\
A(Vg_i) &= \widetilde{\theta}_i(Vg_i)
\end{aligned}$$

which shows that Vg_i's are also approximate eigenvectors of A corresponding to $\widetilde{\theta}_i$'s.

This is called the *harmonic Rayleigh-Ritz procedure* for finding interior eigenvalues near σ in [20]. $\widetilde{\theta}_i$'s are the harmonic Ritz values, and Vg_i's are the corresponding harmonic Ritz vectors.

Since the slow convergence of GMRES is often caused by eigenvalues that are very (relatively) small in magnitude, i.e. they are located near 0. In order to compute those for use in deflation, we will apply the harmonic Rayleigh-Ritz procedure with $\sigma = 0$. The problem in (2.9) becomes

$$V^*A^*Vg = \frac{1}{\widetilde{\theta}}V^*A^*AVg \tag{2.10}$$

We will use similar versions of the harmonic Rayleigh-Ritz procedure in the next sections. Furthermore, we will include better formulas to yield more favorable results in terms of accuracy and implementation.

2.3 Improving the Convergence of GMRES using Eigenvectors

As shown in Example 2.1, having unfavorable eigenvalues could have a significantly negative impact on the convergence of restarted GMRES. To address this problem, one approach is to use a preconditioner [2, 3, 6, 9, 10, 25, 29] , which

effectively changes the spectrum and avoids having tough eigenvalues. Another approach is to eliminate – or to deflate – the unfavorable eigenvalues by adding the corresponding approximate eigenvectors into the Krylov subspace [12] [14]. Combinations of preconditioning and deflating are also possible. In this dissertation, we will focus only on deflation by adding approximate eigenvectors into the subspace.

2.3.1 GMRES-E: Restarted GMRES with Eigenvectors

Developed by Ronald B. Morgan in 1995, the restarted GMRES method augmented with eigenvectors (or GMRES-E) attempts to reduce the negative effect of restarting and of having small eigenvalues by adding approximate eigenvectors into the Krylov subspace [12]. Since the approximate eigenvalues and eigenvectors can be computed from the same Krylov subspace generated during GMRES using the Arnoldi method for eigenvalues [1, 23], this can be built into the algorithm without incurring much additional expenses. Furthermore, it can be shown that adding a converged eigenvector to the subspace will effectively eliminate the corresponding eigenvalue from the spectrum, in other word, that eigenvalue is deflated.

The Krylov subspace generated by GMRES-E(m,k) is

$$K_{(m,k)}(A, r_0) = span\{r_0, Ar_0, A^2r_0, ..., A^{m-k-1}r_0, \tilde{y}_1, \tilde{y}_2, ...\tilde{y}_k\} \tag{2.11}$$

where m is the size of the subspace, and $\tilde{y}_1, \tilde{y}_2, ...\tilde{y}_k$ are approximate eigenvectors associated with the k smallest eigenvalues. Even though this subspace does not have the form of a general Krylov subspace like (2.1), it is a Krylov subspace when $\tilde{y}_1, \tilde{y}_2, ...\tilde{y}_k$ are computed as harmonic Ritz vectors [13].

Implementation of GMRES-E. The implementation for GMRES-E is quite simple. Let m be the dimension of the Krylov subspace, and k be the number of approximate eigenvectors used. The algorithm will generate a matrix $W_m \in \mathbb{C}^{n\times(m)}$ whose first $m-k$ columns are orthonormal Arnoldi vectors, and whose last k vectors are the approximate eigenvectors, and a matrix $V_{m+1} \in \mathbb{C}^{n\times(m+1)}$ whose first

$m-k+1$ columns are the Arnoldi vectors, and whose last k columns are formed by orthonormalizing the vectors $A\tilde{y}_i$, for $i = 1, 2, ..., k$ against the previous columns of V_{m+1}, which results in the following Arnoldi recurrence:

$$AW_m = V_{m+1}\underline{H}_m \tag{2.12}$$

where $\underline{H}_m$ is an $(m+1)$ by (m) upper-Hessenberg matrix.

The minimum residual solution can be calculated the same way as for standard GMRES. Let $\beta = ||r_0||$, find $d \in \mathbb{C}^m$ that minimizes $||\beta e_1 - \underline{H}_m d||$, then compute the approximate solution

$$\hat{x} = x_0 + W_m d \tag{2.13}$$

In order to extract the approximate eigenvectors from the subspace, since we want to find good approximations to the smallest eigenvalues that are responsible for the slow convergence of restarted GMRES, we will use a harmonic version of the Rayleigh-Ritz procedure in [13] to solve the reduced $(m+k)$ by $(m+k)$ generalized eigenvalue problem

$$W^*A^*Wg_i = \frac{1}{\widetilde{\theta}_i}W^*A^*AWg_i \tag{2.14}$$

Using the Arnoldi recurrence (2.12), the (2.14) problem becomes

$$H_m^*g_i = \frac{1}{\widetilde{\theta}_i}\underline{H}_m^*\underline{H}_m g_i \tag{2.15}$$

which makes the implementation a bit easier since the entries of $\underline{H}_m$ have already been computed from the Arnoldi process. Another possibly better way [20] is to solve

$$(H_m + h_{m+1,m}^2 H_m^{-*}e_m e_m^*)g_i = \widetilde{\theta}_i g_i \tag{2.16}$$

Only the g_i's associated with the k largest values of $\frac{1}{\widetilde{\theta}_i}$'s (or k smallest harmonic Ritz values $\widetilde{\theta}_i$'s) are needed, which in turn give the approximate eigenvectors,

Algorithm 2.6 GMRES-E

1: Compute $r_0 = b - Ax_0$. Run one cycle of GMRES(m) and compute k harmonic Ritz vectors: $\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k$.
2: Generate the Arnoldi basis V_{m-k+1}, and the matrix $\underline{H}_{m-k}$.
3: Let $W(:, 1 : m-k) = V_{m-k}$, and $W(:, m-k+1 : m) = [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k]$
4: **for** $j = m-k+1 : m$ **do**
5: Compute $w = Aw_j$
6: **for** $i = 1, 2, ..., j$ **do**
7: $h_{ij} = (w, v_i)$
8: $w = w - h_{ij}v_i$
9: **end for**
10: $h_{j+1,j} = ||w||$.
11: $v_{j+1} = w/h_{j+1,j}$
12: **end for**
13: Let $\beta = ||r_0||$. Compute d that minimizes $||\beta e_1 - \underline{H}_m d||$.
14: Update the approximate solution $\hat{x} = x_0 + W_m d$
15: Check for convergence, if satisfied, then Stop, else set $x_0 = \hat{x}$, and $r_0 = b - A\hat{x}$.
16: Solve $(H_m + h^2_{m+1,m} H_m^{-*} e_m e_m^*) g_i = \widetilde{\theta}_i g_i$ for g_i's.
17: Form the new approximate eigenvectors $\tilde{y}_i = W_m g_i$, then go to Step 2.

in the form of harmonic Ritz vectors $\tilde{y}_i = W_m g_i$, that can be added to the subspace to help with deflation.

Analysis of GMRES-E. The reason GMRES-E works so well is due to the fact that not only that the subspace (2.11) is a Krylov subspace, it also contains smaller Krylov subspaces with each $\tilde{y}_i$ as the starting vector

$$span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i, ..., A^{m-1}\tilde{y}_i\}$$

which allows each approximate eigenvector to be improved as the algorithm progresses until they are accurate enough to deflate the corresponding eigenvalue. Let the approximate solution to the linear system is $\hat{x} = x_0 + W_m d$ and the approximate eigenvectors are $\tilde{y}_i = W g_i$ for $i = 1, 2, ..., k$ as generated by GMRES-E(m,k). Suppose the residual vector of the linear equations is given by

$$r = b - A\hat{x}$$

and the residuals of the eigenvectors (sometimes called the *eigenresidual vectors*)

are defined as

$$r_i = A\tilde{y}_i - \widetilde{\theta}_i\tilde{y}_i, \text{ for each } i = 1, ..., k$$

then r and r_i's are multiples of each other. In other word, $r_i = \gamma_i r$ for some $\gamma \in \mathbb{R}$ [13]. We use the residual norms of the eigenvectors (or *eigenresidual norms*) to monitor the accuracy of the approximate eigenvectors.

Proposition 2.3. *The subspace generated by GMRES-E is a Krylov subspace. i.e.*

$$span\{r_0, Ar_0, A^2r_0, ..., A^{m-k-1}r_0, \tilde{y}_1, \tilde{y}_2, ...\tilde{y}_k\} = span\{s, As, A^2s, ..., A^{m-1}s\}$$

for some vector s. Furthermore, it contains Krylov subspaces with each $\tilde{y}_i$ as the starting vector.

Proof. For simplicity, we will show that $span\{r_0, Ar_0, \tilde{y}_i\} = span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i\}$, where r_0 is the new residual obtained from the end of the previous cycle

$$r_0 = r = b - A\hat{x}$$

Since the residual vectors are multiples of each other, we have

$$r_i = A\tilde{y}_i - \widetilde{\theta}_i\tilde{y}_i = \gamma_i r_0 \text{ for some } \gamma_i \in \mathbb{R}$$

We can write the residual of the linear system as

$$r_0 = \frac{1}{\gamma_i}A\tilde{y}_i - \frac{\widetilde{\theta}_i}{\gamma_i}\tilde{y}_i$$

which is a linear combination of $\tilde{y}_i$ and $A\tilde{y}_i$, so

$$r_0 \in span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i\}$$

Next,

$$Ar_0 = \frac{1}{\gamma_i}A^2\tilde{y}_i - \frac{\widetilde{\theta}_i}{\gamma_i}A\tilde{y}_i$$

which is a linear combination of $A\tilde{y}_i$ and $A^2\tilde{y}_i$, so

$$Ar_0 \in span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i\}$$

Hence,

$$span\{r_0, Ar_0, \tilde{y}_i\} \subset span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i\}$$

Now consider

$$A\tilde{y}_i = \widetilde{\theta}_i\tilde{y}_i + \gamma_i r_0$$

which is a linear combination of $\tilde{y}_i$ and r_0, so

$$A\tilde{y}_i \in span\{r_0, Ar_0, \tilde{y}_i\}$$

Next,

$$\begin{aligned} A^2\tilde{y}_i &= \widetilde{\theta}_i A\tilde{y}_i + \gamma_i Ar_0 \\ &= \widetilde{\theta}_i(\widetilde{\theta}_i\tilde{y}_i + \gamma_i r_0) + \gamma_i Ar_0 \\ &= \widetilde{\theta}_i^2\tilde{y}_i + \widetilde{\theta}_i\gamma_i r_0 + \gamma_i Ar_0 \end{aligned}$$

which is a linear combination of r_0, Ar_0, and $\tilde{y}_i$, so

$$A^2\tilde{y}_i \in span\{r_0, Ar_0, \tilde{y}_i\}$$

Hence,

$$span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i\} \subset span\{r_0, Ar_0, \tilde{y}_i\}$$

It follows from the double inclusion that

$$span\{r_0, Ar_0, \tilde{y}_i\} = span\{\tilde{y}_i, A\tilde{y}_i, A^2\tilde{y}_i\}$$

which is a Krylov subspace with $\tilde{y}_i$ as the starting vector. Furthermore, this is true for each $i = 1, 2, ..., k$, and the proof can be extended for larger values of m. □

As the algorithm progresses, it will keep improving the accuracy of each eigenvectors, and once they're accurate enough they can effectively deflate the corresponding small eigenvalues.

Proposition 2.4. *Suppose the Krylov subspace generated by GMRES-E contains an exact eigenvector z_1 of A, i.e. $\mathcal{K}_{(m,1)}(A, r_0) = span\{r_0, Ar_0, ..., A^{m-2}r_0, z_1\}$, then GMRES converges as if the eigenvalue λ_1 isn't in the spectrum, in other word, λ_1 is deflated.*

Proof. Suppose A has eigenvalues $\{\lambda_1, ..., \lambda_n\}$ and eigenvectors $\{z_1, ..., z_n\}$. The vector r_0 can be expanded in terms of the eigenvectors

$$r_0 = \sum_{j=1}^{n} \beta_j z_j$$

Let $x_m \in \mathcal{K}_m(A, r_0)$ be the approximate solution extracted from the Krylov subspace generated by GMRES-E.

$$x_m = c_1 z_1 + c_2 r_0 + c_3 A r_0 + ... + c_m A^{m-2} r_0$$

Define a polynomial $q(\alpha) = I - c_2\alpha - c_3\alpha^2 - ... - c_m\alpha^{m-1}$

The residual vector can be written as

$$\begin{aligned} r = r_0 - Ax_m &= r_0 - c_1 A z_1 - c_2 A r_0 - c_3 A^2 r_0 - ... - c_m A^{m-1} r_0 \\ &= q(A) r_0 - c_1 \lambda_1 z_1 \\ &= \beta_1 q(\lambda_1) z_1 + \beta_2 q(\lambda_2) z_2 + ... + \beta_n q(\lambda_n) z_n - c_1 \lambda_1 z_1 \end{aligned}$$

Since the solution minimizes the residual norm, any choice we make for the coefficients c_i's will be at most as good as what the method comes up with. Pick $c_1 = \dfrac{\beta_1 q(\lambda_1)}{\lambda_1}$, the residual vector is now

$$r = \sum_{k=2}^{n} \beta_i q(\lambda_i) z_i$$

which means the method only needs to find a polynomial $q(\alpha)$ that is small over $[\lambda_2, \lambda_n]$, i.e. λ_1 no longer affects the convergence of GMRES. As a result, if λ_1 was the main reason for the slow convergence of GMRES, once it is deflated, GMRES will converge at a much faster rate. Similar results can be shown for adding more

than one eigenvectors into the subspace, which effectively deflates more than one eigenvalues.

Numerical results. The following Example 2.2 demonstrates the effect of deflation via adding approximate eigenvectors into the subspace to improve convergence of GMRES.

Example 2.2. Consider the previous Example 2.1 of solving a 500 by 500 linear system with $A = tridiag(-1, 2, -1)$ and b is a random vector, where GMRES(25) converges very slowly. We will solve the system using GMRES-E(25,10), i.e. using 15 Arnoldi vectors and 10 approximate eigenvectors to maintain the same overall size of the Krylov subspace. As a results, the method converges significantly faster. It is quite amazing how simply deflating out 10 small eigenvalues while keeping the size of the Krylov subspace small results in such a great improvement, even better than using a much large subspace, GMRES(400).

2.3.2 GMRES-DR: GMRES with Deflated Restarting

Algebraically equivalent to GMRES-E [14], but instead of adding harmonic Ritz vectors into the Krylov subspace after generating the Arnoldi vectors, GMRES-DR adds the approximate eigenvectors into the subspace before generating the Arnoldi vectors. Similar to GMRES-E, GMRES-DR also uses harmonic Ritz vectors.

Implementation of GMRES-DR. On the first cycle, GMRES-DR uses standard GMRES(m) to generate V_{m+1} and $\underline{H}_m$ with the Arnoldi iteration, and to compute the approximate solution to the linear system as usual. The method then uses the harmonic Rayleigh-Ritz procedure to compute k smallest eigenpairs $(\widetilde{\theta}_i, \tilde{g}_i)$ of the matrix $H_m + h_{m+1,m}^2 H_m^{-*} e_m e_m^*$, where $\widetilde{\theta}_i$'s are the harmonic Ritz values of A [11, 18, 20] with corresponding harmonic Ritz vectors $\tilde{y}_i = V_m \tilde{g}_i$, for $i = 1, 2, ..., k$.

At the beginning of the next cycle, the deflation of small eigenvalues is done by projecting the residual vector over the newly generated set of harmonic Ritz vectors.

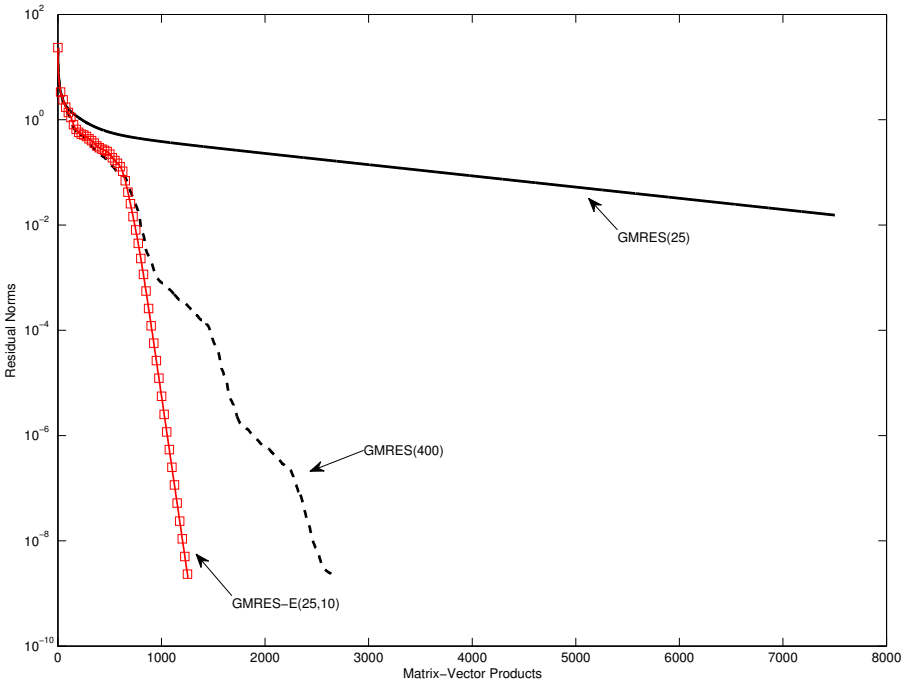


Figure 2.3: GMRES-E(25,10) vs GMRES(25) and GMRES(400).

Note that GMRES-DR does not actually form the harmonic Ritz vectors but carries out the projection as described in Algorithm 2.7. This feature makes GMRES-DR slightly less expensive than GMRES-E computationally, since GMRES-E needs to form and store the approximate eigenvectors to add them into the Krylov subspace.

GMRES-DR uses the following subspace

$$span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, r_0, Ar_0, A^2r_0, ..., A^{m-k-1}r_0\} \tag{2.17}$$

and the Arnoldi reccurence

$$AV_m = V_{m+1}\underline{H}_m \tag{2.18}$$

where $\underline{H}_m$ is upper-Hessenberg, except for a full leading $(k+1)$ by $(k+1)$ portion.

Analysis of GMRES-DR. Similar to GMRES-E, the subspace (2.17) generated by GMRES-DR is a Krylov subspace. The proof for this is identical to the proof shown in the analysis section of GMRES-E that involves implicit restarting [13]. However, for enrichment, here we will use a direct proof [14].

Lemma 2.1. *For any scalar λ, define*

$$\omega_i(\lambda) = \prod_{l=1, l\neq i}^{k} (\widetilde{\theta}_l - \lambda)$$

Let $\widetilde{\theta}_1, \widetilde{\theta}_2, ..., \widetilde{\theta}_k$ be distinct scalars. Then for $0 \leq j \leq k-2$,

$$\sum_{i=1}^{k} \frac{\widetilde{\theta}_i^j}{\omega_i(\widetilde{\theta}_i)} = 0.$$

Proof. See [14]. □

Proposition 2.5. *Suppose a subspace $\mathcal{S} = span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, v\}$ has the property that*

$$A\tilde{y}_i - \widetilde{\theta}_i\tilde{y}_i = \gamma_i v$$

for distinct $\widetilde{\theta}_1, ..., \widetilde{\theta}_k$, and for some nonzero γ_i's, then $\mathcal{S}$ is a Krylov subspace.

Algorithm 2.7 GMRES-DR

1: *Start:* Choose m, k, initial guess x_0, compute $r_0 = b - Ax_0$. Let $v_1 = r_0/||r_0||$, and $\beta = ||r_0||$.
2: *First Cycle:* Apply standard GMRES(m): Generate V_{m+1}, and $\underline{H}_m$ with the Arnoldi iteration, solve $\min||c - \underline{H}_m d||$ for d, where $c = \beta e_1$, and form the new approximate solution $\hat{x} = x_0 + V_m d$. Let $x_0 = \hat{x}$ and $r_0 = b - Ax_0$. Compute k smallest eigenpairs $(\widetilde{\theta}_i, \tilde{g}_i)$ of $H_m + h_{m+1,m}^2 H_m^{-*} e_m e_m^*$.
3: *Orthonormalization of the first k vectors:* Orthonormalize $\tilde{g}_i$'s to form an m by k matrix P_k. (It may be necessary to separate complex vectors into real and imaginary parts and adjust k).
4: *Orthonormalization of the $k+1$ vector:* First, extend the columns of P_k to length $m+1$ by appending a zero entry to each, then orthonormalize the vector $c - \underline{H}_m d$ against them to form p_{k+1}. Note that $c - \underline{H}_m d$ is the vector corresponding to the GMRES residual vector.
5: *Form portion of new H and V using the old H and V:* Let $\underline{H}_k^{new} = P_{k+1}^* \underline{H}_m P_k$ and $V_{k+1}^{new} = V_{m+1} P_{k+1}$. Then let $\underline{H}_k = \underline{H}_k^{new}$ and $V_{k+1} = V_{k+1}^{new}$.
6: *Reorthogonalization of the $k+1$ vector:* Orthogonalize v_{k+1} against the earlier columns of the new V_{k+1}.
7: *Arnoldi iteration:* Apply the Arnoldi iteration from this point to from the rest of V_{m+1} and $\underline{H}_m$. Let $\beta = h_{m+1,m}$.
8: *Form the approximate solution:* Let $c = V_{m+1}^* r_0$ and solve $\min||c - \underline{H}_m d||$ for d. Let $\hat{x} = x_0 + V_m d$. Compute the residual vector $r = b - A\hat{x} = V_{m+1}(c - \underline{H}_m d)$. Check $||r||$ for convergence, and proceed if not satisfied.
9: *Eigenvalue computations:* Compute the k smallest eigenpairs $(\widetilde{\theta}_i, \tilde{g}_i)$ of $H_m + h_{m+1,m}^2 H_m^{-*} e_m e_m^*$.
10: *Restart:* Let $x_0 = \hat{x}$, $r_0 = r$, and go to 3.

Proof. Since this is a direct approach, we will show that

$$span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, v\} = span\{s, As, A^2s, ..., A^ks\}$$

for some s.

Let the starting vector s be

$$s = \sum_{i=1}^{k} \frac{1}{\gamma_i \omega_i(\widetilde{\theta}_i)} \tilde{y}_i$$

with ω_i defined as in Lemma 2.1. Then for $j < k$, $A^j s$ is a combination of $\tilde{y}_i$'s, in particular

$$A^j s = \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^j}{\gamma_i \omega_i(\widetilde{\theta}_i)} \tilde{y}_i \tag{2.19}$$

To show this inductively, notice that for $j = 0$, (2.19) holds trivially since it is how s is defined.

Suppose the following is true:

$$A^{j-1} s = \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^{j-1}}{\gamma_i \omega_i(\widetilde{\theta}_i)} \tilde{y}_i$$

Multiply both sides by A then use the property $A\tilde{y}_i - \widetilde{\theta}_i \tilde{y}_i = \gamma_i v$, we have

$$\begin{aligned}
A^j s &= \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^{j-1}}{\gamma_i \omega_i(\widetilde{\theta}_i)} A\tilde{y}_i \\
&= \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^{j-1}}{\gamma_i \omega_i(\widetilde{\theta}_i)} (\widetilde{\theta}_i \tilde{y}_i + \gamma_i v) \\
&= \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^{j}}{\gamma_i \omega_i(\widetilde{\theta}_i)} \tilde{y}_i + \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^{j-1}}{\omega_i(\widetilde{\theta}_i)} v \\
&= \sum_{i=1}^{k} \frac{\widetilde{\theta}_i^{j}}{\gamma_i \omega_i(\widetilde{\theta}_i)} \tilde{y}_i + (0)v, \text{ (by Lemma 2.1)}
\end{aligned}$$

By induction, (2.19) holds for $j < k$.

Given the construction of the subspace using the specific choice of s, we have

$$span\{s, As, A^2s, ..., A^{k-1}s\} = span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k\}$$

Multiply $A^{k-1}s$ by A and use the property $A\tilde{y}_i - \widetilde{\theta}_i\tilde{y}_i = \gamma_i v$ again, we see that $A^k s$ is a combination of $\tilde{y}_i$'s and v.

It follows that

$$span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, v\} = span\{s, As, A^2s, ..., A^k s\}$$

.i.e. $\mathcal{S}$ is a Krylov subspace. □

Proposition 2.6. *The subspace generated by GMRES-DR*

$$span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, r_0, Ar_0, A^2r_0, ..., A^{m-k-1}r_0\}$$

is a Krylov subspace.

Proof. Define the harmonic Ritz residual vectors as

$$r_i = A\tilde{y}_i - \widetilde{\theta}_i\tilde{y}_i$$

Similar to GMRES-E, these vectors are multiples of the GMRES residual vector [13], in other word,

$$r_i = A\tilde{y}_i - \widetilde{\theta}_i\tilde{y}_i = \gamma_i r_0$$

for some γ_i's. By the previous proposition,

$$span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, r_0\}$$

is a Krylov subspace. Furthermore, since GMRES-DR adds more vectors to subspace using an Arnoldi process, the resulting subspace will be a Krylov subspace. To see what this subspace looks like, first let $v_1, ..., v_{k+1}$ be the orthonormal basis for the subspace $span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, r_0\}$ and consider the next vector v_{k+2} generated by the Arnoldi algoritm.

Since the vector v_{k+2} comes from orthonormalizing Av_{k+1} and we can write $A\tilde{y}_i = \widetilde{\theta}_i\tilde{y}_i + \gamma_i r_0$, we have

$$Av_{k+1} = A\left(c_{k+1}r_0 + \sum_{i=1}^{k} c_i\tilde{y}_i\right)$$

$$= c_{k+1}Ar_0 + \sum_{i=1}^{k} c_i A\tilde{y}_i$$
$$= c_{k+1}Ar_0 + \sum_{i=1}^{k} c_i(\widetilde{\theta}_i\tilde{y}_i + \gamma_i r_0)$$

which is a linear combination of $\tilde{y}_i$'s, r_0, and Ar_0. So,

$$v_{k+2} \in span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, r_0, Ar_0\}$$

Continue for the next basis vectors $v_{k+3}, ..., v_{m+1}$, we see that the Krylov subspace generated by GMRES-DR is, in fact, the subspace

$$span\{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k, r_0, Ar_0, A^2r_0, ..., A^{m-k-1}\}$$

□

As mentioned before, in GMRES-DR, the harmonic Ritz vectors are not actually formed, which makes it slightly less expensive in terms of storage compared to GMRES-E, where the approximate eigenvectors need to be computed and added into the Krylov subspace. In general, GMRES-DR is favored over GMRES-E when it comes to implementation, unless approximate eigenvectors need to be computed.

Numerical results. The following Example 2.3 demonstrates the effect of deflation via adding approximate eigenvectors into the subspace to improve convergence of GMRES. Unlike in GMRES-E, here, the eigenvectors are not actually formed.

Example 2.3. For comparison purposes, we will solve the same 500 by 500 linear system with $A = tridiag(-1, 2, -1)$ and b is a random vector, where GMRES(25) fails to converge, this time with GMRES-DR(25,10) , i.e. using 15 Arnoldi vectors and 10 harmonic Ritz vectors. Since GMRES-DR(25,10) is algebraically equivalent to GMRES-E(25,10), the convergence is improved at a similar rate.

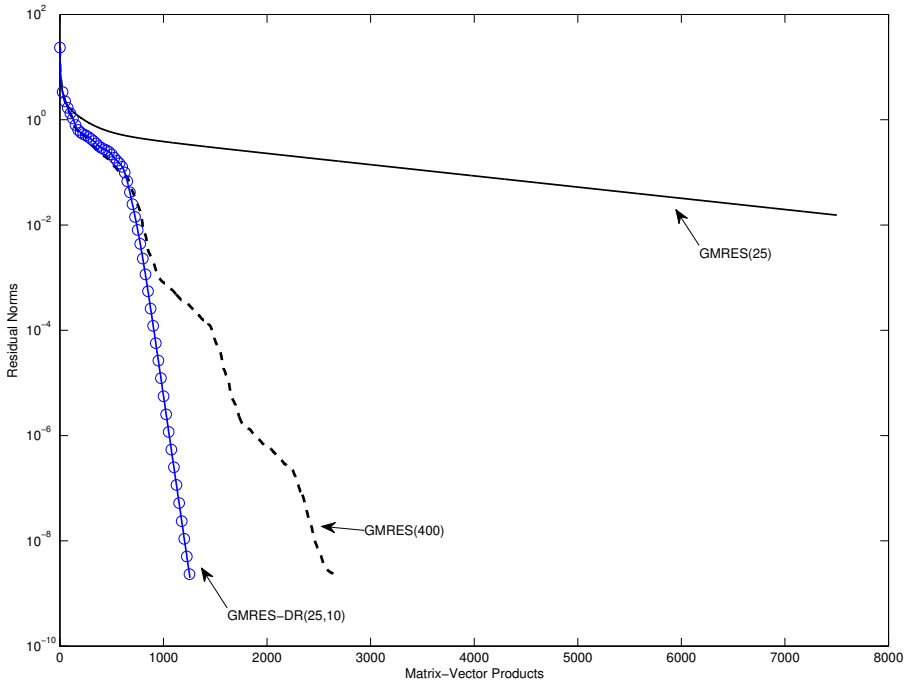


Figure 2.4: GMRES-DR(25,10) vs GMRES(25) and GMRES(400).

CHAPTER THREE

Solving a Sequence of Systems of Linear Equations

3.1 Same Matrix, Changing Right-hand Sides

Consider a sequence of linear systems that come from a numerical partial differential equation where the coefficient matrix stays the same while the right-hand sides are changing (due to changes in the boundary conditions, perhaps). Supposed we have solved the first system $Ax^{(1)} = b^{(1)}$ using GMRES-DR. Since we now have good approximate eigenvectors of A, we should be able to take advantage of them immediately for solving the second system $Ax^{(2)} = b^{(2)}$. The deflation should be effective to speed up the convergence of GMRES right from the start, instead of having to wait for the eigenvectors to become accurate, as seen in GMRES-DR. This approach, called GMRES-Proj [14], has cycles of GMRES alternating with a projection phase over a the fixed set of approximate eigenvectors. Suppose at the end of GMRES-DR, we save a fixed portion of the matrices V and H, the following recurrence holds

$$AV_k = V_{k+1}\underline{H}_k \tag{3.1}$$

where $\underline{H}_k$ is a $k+1$ by k full matrix and V_k has columns that span the set of approximate eigenvectors. This can be used to project the residual vector over the approximate eigenvectors in Algorithm 3.1.

Example 3.1. Consider solving a sequence of five 500 by 500 linear systems where $A = tridiag(-1, 2, -1)$ and the right-hand sides $b^{(1)}, ..., b^{(5)}$ are random vectors. For comparison, in the first plot, we solve each system using GMRES-DR(25,10), which costs a total of 6205 matrix-vector products. In the second plot, we solve the first system using GMRES-DR(25,10), then run GMRES-Proj for the remaining 4, using the approximate eigenvectors obtained from the first run. Since the eigenvectors

Algorithm 3.1 GMRES-DR/GMRES-Proj with MINRES Projection

1: Solve the first system $Ax^{(1)} = b^{(1)}$ using GMRES-DR(m,k) and keep the matrices V_{k+1}^{pr}, and $\underline{H}_k^{pr}$ at the end.
2: **for** $i = 2$: number of right-hand sides. **do**
3: Let the current approximate solution be x_0
4: Let the residual vector be $r_0 = b^{(i)} - Ax_0$
5: Solve $min||c - \underline{H}_k^{pr} d||$, where $c = (V_{k+1}^{pr})^* r_0$.
6: Update the new approximate solution $x_k = x_0 + V_{k+1}^{pr} d$.
7: Update the new residual vector $r_k = r_0 - AV_{k+1}^{pr} d$.
8: Check for convergence, if satisfied, then Stop, else let $r_0 = r_k$ and $x_0 = x_k$.
9: Run one cycle of GMRES(m-k) with the new residual vector.
10: Solve the least-square problem. Compute the approximate solution $\hat{x}$.
11: Check for convergence, if satisfied, then Stop, else let $x_0 = \hat{x}$ and go to 3.
12: **end for**

are accurate enough, GMRES-Proj can start the deflation process effectively, and immediately, without having to wait for them to become accurate – a common delay seen in GMRES-DR and GMRES-E. This requires only about 3885 matrix-vector products, which is a significant cost reduction, especially if there are many more right-hand sides. There are many methods developed to solve linear equations with multiple right-hand sides such as block methods and seed methods [15, 16, 17].

3.2 Changing Matrices, Changing Right-hand Sides

Now consider solving a sequence of linear systems

$$A_{(i)}x^{(i)} = b^{(i)}, i = 1, 2, ...$$

where $A_{(i)} \in \mathbb{C}^{n \times n}$ and $b^{(i)} \in \mathbb{C}^n$ change from one system to the next. Each subsequent system can be viewed as a perturbation of the previous system

$$A_{(i+1)} = A_{(i)} + E_{(i)}$$

where $E_{(i)}$ is the perturbation matrix.

3.2.1 Reusing Eigenvectors with GMRES-Proj

When $||E_{(i)}||$ is small enough so that $E_{(i)}$ does not significantly alter the spectrum of $A_{(i)}$, the approximate eigenvectors of $A_{(i)}$, namely $y_1^{(i)}, y_2^{(i)}, ...y_k^{(i)}$ should also

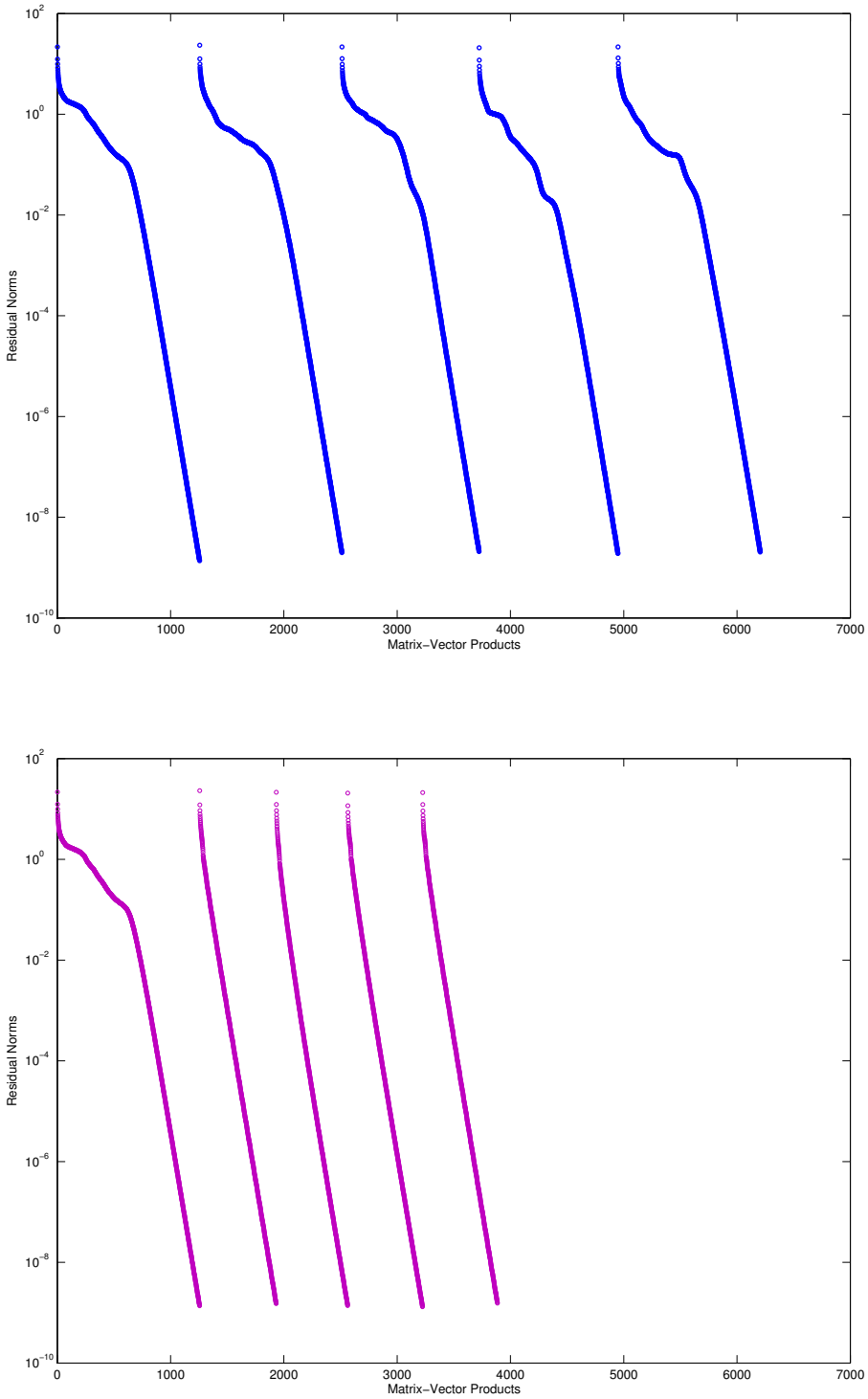


Figure 3.1: GMRES-DR (top) vs GMRES-DR/GMRES-Proj (bottom).

be approximate eigenvectors of $A_{(i+1)}$, then GMRES-Proj can re-use them to deflate small eigenvalues of $A_{(i+1)}$, thus speeding up the convergence of GMRES immediately. Example 3.2 illustrates this idea.

Example 3.2. Consider solving a sequence of twenty 500 by 500 linear systems where $A_{(1)} = tridiag(-1, 2, -1)$, $A_{(i)} = A_{(i-1)} + E$ for $i = 2, 3, ..., 20$, and E is a tridiagonal, random, nonsymmetric matrix with $||E|| = 1.0e - 6$. The right-hand sides are random vectors. For comparison, we first solved the sequence using GMRES-DR(25,10) for every system. We then solve the same sequence, this time, using GMRES-DR(25,10) on the first system, and generate 10 approximate eigenvectors; for the remaining system, we use GMRES-Proj with the approximate eigenvectors obtained from the first system. Even though the systems are changing, the change is small enough that GMRES-Proj can take full advantage of the approximate eigenvectors for deflation almost as effectively as the case where the matrix is not changing in Example 3.1. The residual norms in Figure 3.2 and the matrix-vector products per system in Table 3.1 demonstrate the advantage of reusing the eigenvectors from one system to speed up the convergence of GMRES for solving the subsequent systems. We use MATLAB's default 'Run and Time' feature to calculate the total CPU time to show how fast GMRES-Proj can be compared to GMRES-DR.

The results from Example 3.2 indicate that GMRES-Proj is very efficient at re-using approximate eigenvectors when the systems do not change much. However, for large enough change, or when the accumulated change is significant enough that the approximate eigenvectors of $A_{(i)}$ are no longer good enough to deflate the small eigenvalues of $A_{(i+j)}$ for some $j \in \mathbb{N}$, we suspect that GMRES-Proj will lose its effectiveness.

Example 3.3. We repeat Example 3.2 using a slightly larger perturbation, $||E|| = 1.0e - 5$, and the resulting matrix-vector products to solve each system shown in

Table 3.1: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-6$

System	All G-DR	G-DR/G-Proj
1	1195	1195
2	1255	660
3	1210	615
4	1225	645
5	1255	660
6	1255	645
7	1210	660
8	1255	660
9	1210	630
10	1255	660
11	1225	645
12	1210	660
13	1225	645
14	1240	630
15	1165	570
16	1135	630
17	1225	630
18	1255	645
19	1270	660
20	1240	645
Total MVP's	24515	13390
Total CPU Time	24.606s	7.827s

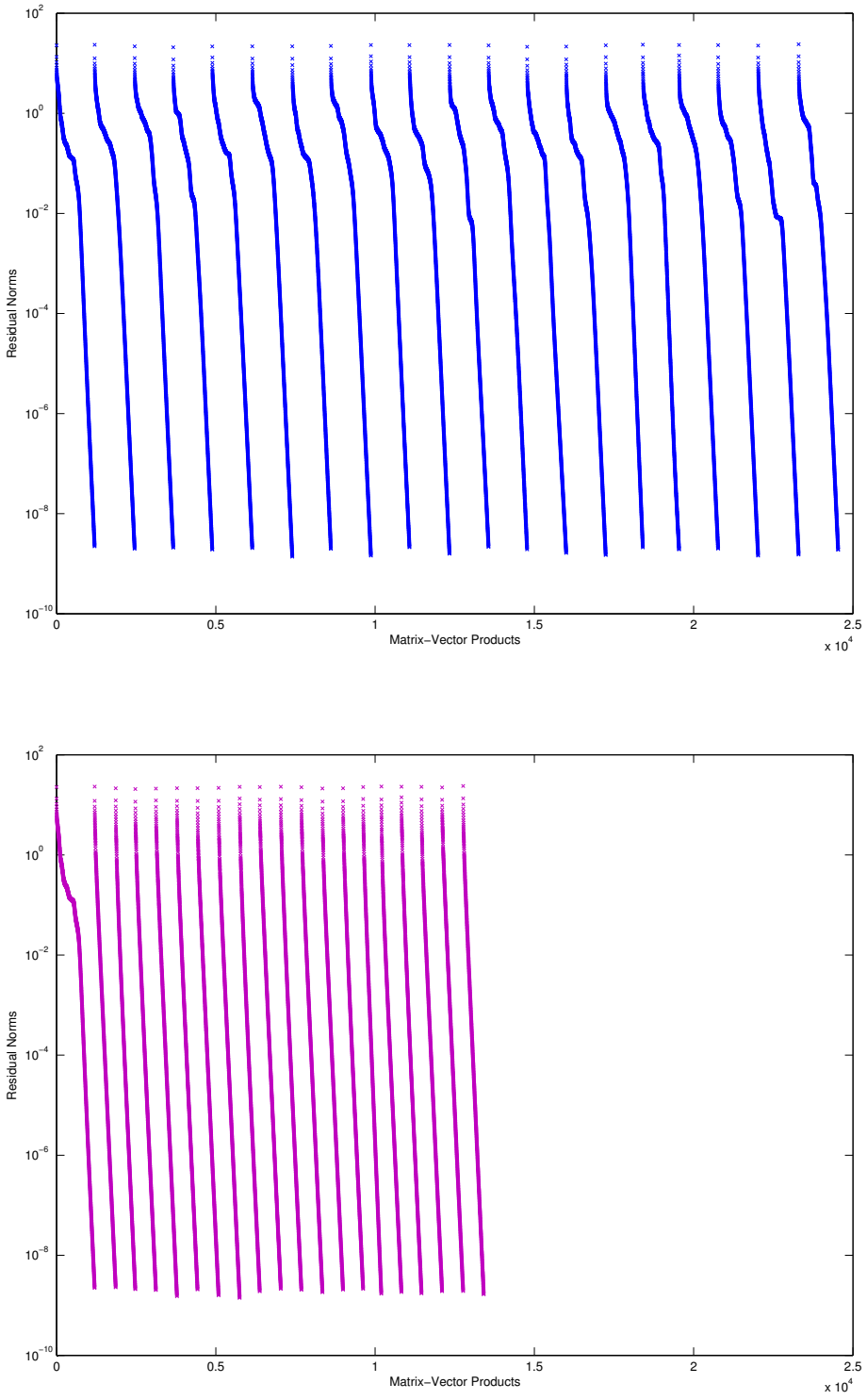


Figure 3.2: GMRES-DR (top) vs GMRES-DR/G-Proj (bottom).

Table 3.2 are as expected. Notice from system 14 and beyond, since the approximate eigenvectors of $A_{(1)}$ are no longer good enough to help with deflating the small eigenvalues of $A_{(i)}$ for $i = 14, 15, ..., 20$ at the desired rate, GMRES-Proj begins to lose its effectiveness in terms of re-using approximate eigenvectors to speed up the convergence of GMRES. We also include the total CPU time in seconds. These numerical results suggest that we need a new method that has the capability not only to re-use approximate eigenvectors from a previous system, but also to improve those vectors to make them more suitable for the current system.

Table 3.2: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-5$

System	All G-DR	G-DR/G-Proj
1	1195	1195
2	1255	660
3	1210	615
4	1225	645
5	1255	660
6	1255	645
7	1210	660
8	1255	660
9	1210	645
10	1255	690
11	1225	660
12	1210	675
13	1225	690
14	1240	720
15	1180	810
16	1180	900
17	1225	1050
18	1255	1065
19	1270	1245
20	1240	1440
Total MVP's	24575	16330
Total CPU Time	24.591s	9.468s

3.2.2 Recycling Eigenvectors with GMRES-E

When the change is significant enough that projection doesn't work as well, we will attempt to recycle and improve the eigenvectors by adding them into the new Krylov subspace using an updated version of GMRES-E, namely GMRES-E(recycled), which is almost the same as Algorithm 2.6 starting from Step 2 with $\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k$ are approximate eigenvectors obtained from a previous system instead of being generated from scratch by running one cycle of GMRES(m). This approach shortens the amount of time it takes for the eigenvectors to be accurate enough to help with deflation, thus speeding up the convergence for solving the linear system.

Algorithm 3.2 GMRES-E(recycled)

1: Compute $r_0 = b - Ax_0$. If the approximate eigenvectors $\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k$ are available, then go to the next step. If not, Run one cycle of GMRES(m) and compute k harmonic Ritz vectors: $\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k$.
2: Generate the Arnoldi basis V_{m-k+1}, and the matrix $\underline{H}_{m-k}$.
3: Let $W(:, 1 : m-k) = V_{m-k}$, and $W(:, m-k+1 : m) = [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k]$
4: **for** $j = m-k+1 : m$ **do**
5: Compute $w = Aw_j$
6: **for** $i = 1, 2, ..., j$ **do**
7: $h_{ij} = (w, v_i)$
8: $w = w - h_{ij}v_i$
9: **end for**
10: $h_{j+1,j} = ||w||$.
11: $v_{j+1} = w/h_{j+1,j}$
12: **end for**
13: Let $\beta = ||r_0||$. Compute d that minimizes $||\beta e_1 - \underline{H}_m d||$.
14: Update the approximate solution $\hat{x} = x_0 + W_m d$
15: Check for convergence, if satisfied, then Stop, else set $x_0 = \hat{x}$, and $r_0 = b - A\hat{x}$.
16: Solve $(H_m + h_{m+1,m}^2 H_m^{-*} e_m e_m^*) g_i = \widetilde{\theta}_i g_i$ for g_i's.
17: Form the new approximate eigenvectors $\tilde{y}_i = W_m g_i$, then go to Step 2.

Example 3.4. We repeat example 3.3 by solving the first system using GMRES-DR(25,10) to generate 10 approximate eigenvectors. For the remaining systems, instead of using GMRES-Proj, we use GMRES-E(25,10) (recycled), which improves the approximate eigenvectors from the previous system to make them more suitable for deflating the small eigenvalues in the current system.

Table 3.3: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-5$

System	All G-DR	G-DR/G-Proj	G-DR/G-E(rec)
1	1195	1195	1195
2	1255	660	660
3	1210	615	630
4	1225	645	660
5	1255	660	660
6	1255	645	660
7	1210	660	660
8	1255	660	660
9	1210	645	630
10	1255	690	660
11	1225	660	645
12	1210	675	660
13	1225	690	660
14	1240	720	645
15	1180	810	570
16	1180	900	630
17	1225	1050	630
18	1255	1065	645
19	1270	1245	660
20	1240	1440	645
Total MVP's	24575	16330	13465
Total CPU Time	24.591s	9.468s	34.599s

This new approach of recycling approximate eigenvectors and improving them as the systems change eliminates the issue that arose in the previous approach of using only GMRES-Proj for the subsequent systems.

Analysis of GMRES-E(recycled). As explained by Morgan in [14], the main reason GMRES-E and GMRES-DR work so well is because the subspaces generated by each method possess certain *Krylov properties*: the residual vectors are multiples of each other; the whole subspace is a Krylov subspace that contains other Krylov subspaces with each harmonic Ritz vector as the starting vector. Supposed we are solving the l^{th} system $A_{(l)}x^{(l)} = b^{(l)}$ with GMRES-E(recycled) using approximate eigenvectors $y_1^{(j)}, ..., y_k^{(j)}$ from a previous j^{th} system. At the end of the first cycle, the subspace generated is

$$\mathcal{K}_1 = span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, ..., A_{(l)}^{m-k+1}r_0^{(l)}, y_1^{(j)}, ..., y_k^{(j)}\} \tag{3.2}$$

which is not a Krylov subspace, in general. However, the subspaces generated after the first cycles *have a chance* to be Krylov subspaces and retain all of the Krylov properties that GMRES-E has. To see this, we will explore the algorithm to see what happens at the end of the first cycle of GMRES-E(recycled). Let W_m be the $m \times m$ matrix whose first $m-k$ columns are Arnoldi vectors and the last k columns are approximate eigenvectors $y_1^{(j)}, ..., y_k^{(j)}$. The Arnoldi algorithm produces the usual orthonormal matrix V_{m+1} and the upper-Hessenberg $\underline{H}_m$ with the relation

$$A_{(l)}W_m = V_{m+1}\underline{H}_m \tag{3.3}$$

The algorithm then find d that minimizes $||\beta e_1 - \underline{H}_m||$, where $\beta = ||r_0^{(l)}||$ and computes the approximate solution to the linear system $\hat{x} = x_0 + W_m d$. Next, the algorithm computes the new approximate eigenvectors of $A_{(l)}$ by solving

$$W^* A_{(l)}^* W g_i = \frac{1}{\theta_i} W^* A_{(l)}^* A_{(l)} W g_i \tag{3.4}$$

using the harmonic Rayleigh-Ritz procedures for g_i's, then form the new approximate eigenvectors

$$y_i^{(l)} = W_m g_i$$

Finally, the residual vector of the linear system is computed, checked for convergence (which we assume is not reached since this is only after the first cycle)

$$r_0^{(l)} = b^{(l)} - A_{(l)}\hat{x} \tag{3.5}$$

This new residual vector will be used to build the subspace in the next cycle of GMRES-E(rec).

Define the residual of the newly formed approximate eigenvectors

$$r_i^{(l)} = A_{(l)}y_i^{(l)} - \theta_i y_i^{(l)}, \text{ for } i = 1, ..., k \tag{3.6}$$

Lemma 3.1. *Denote the dimension m subspace generated at the end of the first cycle of GMRES-E(recycled)*

$$\mathcal{K}_1 = span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, ..., A_{(l)}^{m-k+1}r_0^{(l)}, y_1^{(j)}, ..., y_k^{(j)}\}$$

If the subspace $A_{(l)}\mathcal{K}_1$ has dimension $m+1$, then the residual vectors $r^{(l)}$ and $r_i^{(l)}$'s are multiples of each other.

Proof. Since $r^{(l)}$ and $r_i^{(l)}$'s reside in the same subspace, we will show that they are orthogonal to the same subspace spanned by the columns of $A_{(l)}W_m$.

First, using the Arnoldi relation (3.3),

$$\begin{aligned}
(A_{(l)}W_m)^* r^{(l)} &= (A_{(l)}W_m)^*(r_0^{(l)} - A_{(l)}W_m d) \\
&= (V_{m+1}\underline{H}_m)^* r_0^{(l)} - (V_{m+1}\underline{H}_m)^* V_{m+1}\underline{H}_m d \\
&= \underline{H}_m^* V_{m+1}^* r_0^{(l)} - \underline{H}_m^* V_{m+1}^* V_{m+1}\underline{H}_m d \\
&= \underline{H}_m^* \beta e_1 - \underline{H}_m^* \underline{H}_m d \\
&= 0 \text{ (Since } d \text{ is the solution of } \min ||\beta e_1 - \underline{H}_m d||)
\end{aligned}$$

Hence, $r^{(l)} \perp A_{(l)}W_m$, i.e. the residual vector of the linear equation is orthogonal to the subspace spanned by the columns of $A_{(l)}W_m$.

Now, consider the residuals of the eigenvectors

$$\begin{aligned}(A_{(l)}W_m)^* r_i^{(l)} &= (A_{(l)}W_m)^*(A_{(l)}y_i^{(l)} - \theta_i y_i^{(l)}) \\ &= W_m^* A_{(l)}^* A_{(l)} W_m g_i - \theta_i W_m^* A_{(l)}^* W_m g_i \\ &= 0 \quad \text{(Since } g_i \text{ and } \theta_i \text{ come from solving (3.4))}\end{aligned}$$

Hence, $r_i^{(l)} \perp A_{(l)}W_m$ for each $i = 1, ..., k$, i.e. the residuals of the eigenvectors are orthogonal to the subspace spanned by the columns of $A_{(l)}W_m$.

Since the residual vectors reside in the same subspace of dimension m and they are all orthogonal to the same subspace of dimension $m+1$, they must be parallel to each other, or multiples of each other. □

As GMRES-E(recycled) continues, it uses the new residual (3.5) as the starting vector to form the new subspace by letting $r_0^{(l)} = r^{(l)}$.

$$span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, ..., A_{(l)}^{m-k+1}r_0^{(l)}, y_1^{(l)}, ..., y_k^{(l)}\} \tag{3.7}$$

Proposition 3.1. *Denote the dimension m subspace generated at the end of the first cycle of GMRES-E(recycled)*

$$\mathcal{K}_1 = span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, ..., A_{(l)}^{m-k+1}r_0^{(l)}, y_1^{(j)}, ..., y_k^{(j)}\}$$

If the subspace $A_{(l)}\mathcal{K}_1$ has dimension $m+1$, then the subspace generated at the end of the second cycle of GMRES-E(recycled)

$$\mathcal{K}_2 = span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, ..., A_{(l)}^{m-k+1}r_0^{(l)}, y_1^{(l)}, ..., y_k^{(l)}\}$$

is a Krylov subspace.

Proof. We will show $span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, y_i^{(l)}\} = span\{y_i^{(l)}, A_{(l)}y_i^{(l)}, A_{(l)}^2 y_i^{(l)}\}$. Since $A_{(l)}\mathcal{K}_1$ has dimension $m+1$, by Lemma 3.1, the residual vectors are multiples of each other.

$$r_i^{(l)} = A_{(l)}y_i^{(l)} - \theta_i y_i^{(l)} = \gamma_i r_0^{(l)}, \text{ for some } \gamma_i \in \mathbb{C}$$

we can write the residual of the linear system as

$$r_0^{(l)} = \frac{1}{\gamma_i} A_{(l)} y_i^{(l)} - \frac{\theta_i}{\gamma_i} y_i^{(l)}$$

which is a linear combination of $y_i^{(l)}$ and $A_{(l)} y_i^{(l)}$, so

$$r_0^{(l)} \in span\{y_i^{(l)}, A_{(l)} y_i^{(l)}, A_{(l)}^2 y_i^{(l)}\}$$

Next,

$$A_{(l)} r_0^{(l)} = \frac{1}{\gamma_i} A_{(l)}^2 y_i^{(l)} - \frac{\theta_i}{\gamma_i} A_{(l)} y_i^{(l)}$$

which is a linear combination of $A_{(l)} y_i^{(l)}$ and $A_{(l)}^2 y_i^{(l)}$, so

$$A_{(l)} r_0^{(l)} \in span\{y_i^{(l)}, A_{(l)} y_i^{(l)}, A_{(l)}^2 y_i^{(l)}\}$$

Hence,

$$span\{r_0^{(l)}, A_{(l)} r_0^{(l)}, y_i^{(l)}\} \subset span\{y_i^{(l)}, A_{(l)} y_i^{(l)}, A_{(l)}^2 y_i^{(l)}\}$$

Now consider

$$A_{(l)} y_i^{(l)} = \theta_i y_i^{(l)} + \gamma_i r_0^{(l)}$$

which is a linear combination of $y_i^{(l)}$ and $r_0^{(l)}$, so

$$A_{(l)} y_i^{(l)} \in span\{r_0^{(l)}, A_{(l)} r_0^{(l)}, y_i^{(l)}\}$$

Next,

$$\begin{aligned} A_{(l)}^2 y_i^{(l)} &= \theta_i A_{(l)} y_i^{(l)} + \gamma_i A_{(l)} r_0^{(l)} \\ &= \theta_i (\theta_i y_i^{(l)} + \gamma_i r_0^{(l)}) + \gamma_i A_{(l)} r_0^{(l)} \\ &= \theta_i^2 y_i^{(l)} + \theta_i \gamma_i r_0^{(l)} + \gamma_i A_{(l)} r_0^{(l)} \end{aligned}$$

which is a linear combination of $r_0^{(l)}$, $A_{(l)} r_0^{(l)}$, and $y_i^{(l)}$, so

$$A_{(l)}^2 y_i^{(l)} \in span\{r_0^{(l)}, A_{(l)} r_0^{(l)}, y_i^{(l)}\}$$

Hence,

$$span\{y_i^{(l)}, A_{(l)} y_i^{(l)}, A_{(l)}^2 y_i^{(l)}\} \subset span\{r_0^{(l)}, A_{(l)} r_0^{(l)}, y_i^{(l)}\}$$

It follows from the double inclusion that

$$span\{r_0^{(l)}, A_{(l)}r_0^{(l)}, y_i^{(l)}\} = span\{y_i^{(l)}, A_{(l)}y_i^{(l)}, A_{(l)}^2 y_i^{(l)}\}$$

which is a Krylov subspace with $y_i^{(l)}$ as the starting vector. Furthermore, this is true for each $i = 1, 2, ..., k$, and the proof can be extended for larger values of m and for the subsequent cycles of GMRES-E(recycled). □

Unfortunately, in general, if the subspace $\mathcal{K}_1$ is not a Krylov subspace, the subspace $A_{(l)}\mathcal{K}_1$ is not guaranteed to have dimension $m + 1$, which means it will not be Krylov. However, when the change in the system is small enough that the subspaces generated are nearly Krylov, the method can still solve the linear system as efficiently. Furthermore, by adding $y_1^{(j)}, ..., y_k^{(j)}$ to the subspace in the first cycle, the approximate eigenvectors generated at the end of the first cycle of GMRES-E(recycled) are much more accurate than those generated from scratch at the end of the first cycle of GMRES-E. This allow deflation in GMRES-E(recycled) to take place sooner since GMRES-E (and GMRES-DR) must wait longer until the approximate eigenvectors are accurate enough for deflation to be effective. The following Table 3.4 shows the residual norms of the eigenvectors generated at the end of the first cycle of GMRES-E compared to those generated at the end of the first cycle of GMRES-E(recycled) for solving the second system from Example 3.4.

3.2.3 Regenerating New Eigenvectors with GMRES-DR

Due to the loss of the Krylov property, GMRES-E(recycled) can only improve the accuracy of the eigenvectors to a certain degree. For the most part, this does not have a negative impact on the convergence of GMRES-E(recycled) when solving the linear system, since the eigenvectors – although not very accurate – are accurate *enough* to help with the deflation of small eigenvalues. However, there are extreme cases where the the eigenvectors need to be just a little more accurate before they can help with deflation, but since GMRES-E(recycled) is unable to improve them

Table 3.4: Residual norms of the 10 approximate eigenvectors

Residual Norm	End of First Cycle of G-E	End of First Cycle of G-E(rec)
$\|\|r_1\|\|$	6.50e-03	3.90e-06
$\|\|r_2\|\|$	2.79e-02	3.70e-06
$\|\|r_3\|\|$	6.52e-02	4.00e-06
$\|\|r_4\|\|$	1.14e-01	3.80e-06
$\|\|r_5\|\|$	1.67e-01	3.80e-06
$\|\|r_6\|\|$	2.62e-01	4.10e-06
$\|\|r_7\|\|$	3.34e-01	4.00e-06
$\|\|r_8\|\|$	4.50e-01	3.70e-06
$\|\|r_9\|\|$	5.47e-01	3.80e-06
$\|\|r_{10}\|\|$	6.72e-01	1.00e-05

any further, the method may fail to converge. In these instances, the eigenvectors must be regenerated from a Krylov subspace, and our method of choice is GMRES-DR. To illustrate the need for regenerating new eigenvectors instead of using recycled eigenvectors, we use GMRES-E(recycled) to solve a sequence of linear systems where the change from one matrix to the next is large, then compare the results to GMRES-DR in Example 3.5.

Example 3.5. We repeat Example 3.4 solving $A_{(i)}x^{(i)} = b^{(i)}$ for $i = 1, 2, ..., 20$ with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-2$ using GMRES-E(25,10)(recycled) and compare the results to using GMRES-DR(25,10). Due to such a large, random, and non-symmetric perturbation, the resulting systems are sometimes very tough to solve and require very accurate eigenvectors in order to effectively deflate the small eigenvalues. The plot of the residual norms in Figure 3.3 (top) shows that GMRES-E(recycled) converges very slowly, and sometimes even fails to converge. One of the main reasons some of these systems are so tough to solve is due the fact that the newly perturbed matrices might have very small eigenvalues, sometimes negative eigenvalues near 0. Consider system 3 where GMRES-E(recycled) simply fails to converge. We take a look at the spectrum of $A_{(3)}$ in in Table 3.6 and see that some

of the smallest eigenvalues are negative. This is a very tough problem because those negative eigenvalues must be sufficiently deflated. Since GMRES-E(recycled) cannot make the approximate eigenvectors accurate enough to effectively deflate the corresponding eigenvalues, it may not converge. On the other hand, since GMRES-DR is able to keep improving the accuracy of the eigenvectors until they can effectively deflate the small (and negative) eigenvalues, it actually reaches convergence for solving the linear equations. This example justifies why sometimes new eigenvectors must be regenerated. Figure 3.4 shows the residual norms of the eigenvectors computed by GMRES-E(recycled) (top), and the residual norms computed by GMRES-DR (bottom), indicating that, GMRES-DR is capable of improving the eigenvectors to a much higher accuracy.

Example 3.6. The results from Example 3.5 suggest that GMRES-E(recycled) can struggle when the change from one system to the next is big enough. Another strategy besides regenerating new eigenvectors with GMRES-DR is to use a larger subspace and/or to deflate more eigenvalues. We repeat Example 3.5 using GMRES-E(recycled)(35,15) and compare it to GMRES-DR(35,15). Figure 3.5 shows that even though the eigenvectors computed by GMRES-E(rec)(35,15) are not as accurate as those computed by GMRES-DR(35,15), the subspace is sufficiently large that GMRES-E(rec)(35,15) is still able to solve the linear system without struggling like GMRES-E(rec)(25,10) did, and even performs slightly better than GMRES-DR(35,15) in terms of matrix-vector products. Table 3.7

3.2.4 Combined Algorithm: GMRES-RRR

Although GMRES-E(recycled) alone is sufficient to solve a sequence of changing systems while minimizing the costs in terms of matrix-vector products, notice from Table 3.3, for systems 2 to 12, GMRES-Proj and GMRES-E(recycled) both incur similar expenses in terms of matrix-vector products, but since GMRES-Proj

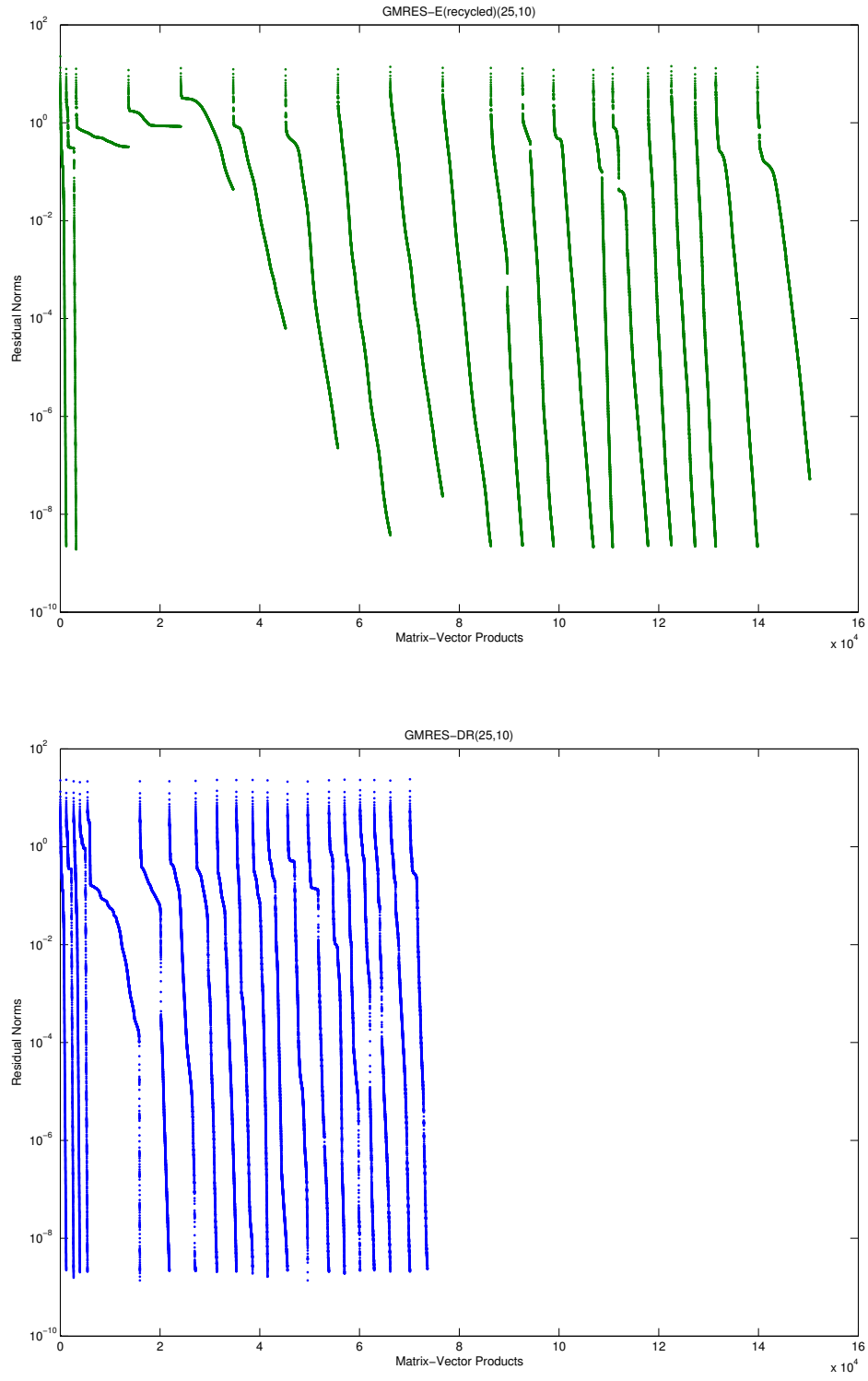


Figure 3.3: GMRES-E(recycled)(25,10) (top) vs GMRES-DR(25,10)(bottom).

Table 3.5: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-2$

System	All G-DR	G-DR/GE(rec)
1	1195	1195
2	1480	1980
3	1255	10500
4	1525	10500
5	10510	10500
6	5905	10500
7	5260	10500
8	4270	10500
9	3895	10500
10	3265	9660
11	2995	6345
12	4000	6210
13	4045	7995
14	4255	3870
15	3130	7095
16	3115	4680
17	2845	4755
18	3220	4140
19	3925	8370
20	3505	10500
Total MVP's	73595	150295
Total CPU Time	74.452s	330.198s

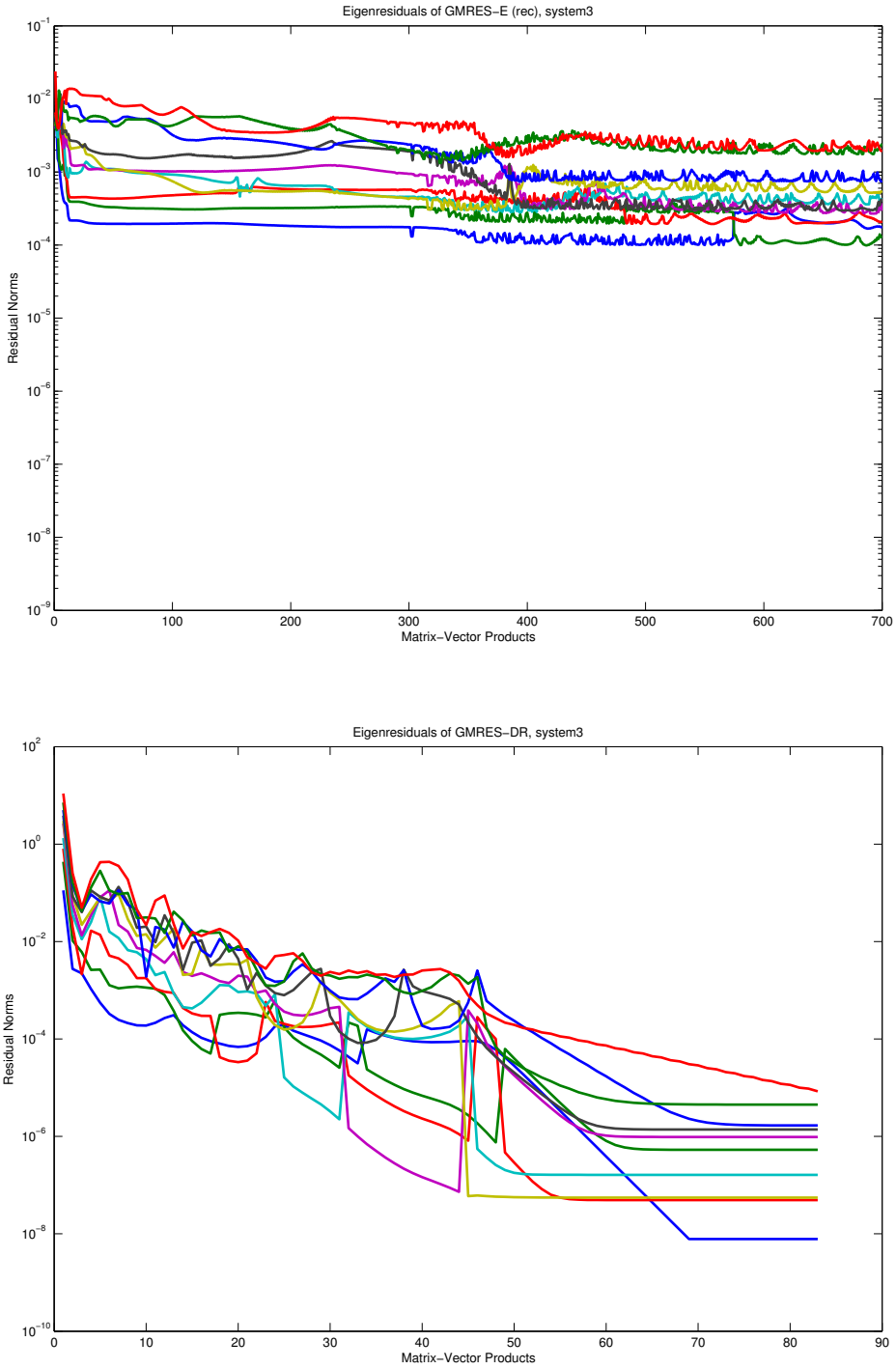


Figure 3.4: Eigenresiduals of GMRES-E(recycled)(top) and GMRES-DR(bottom).

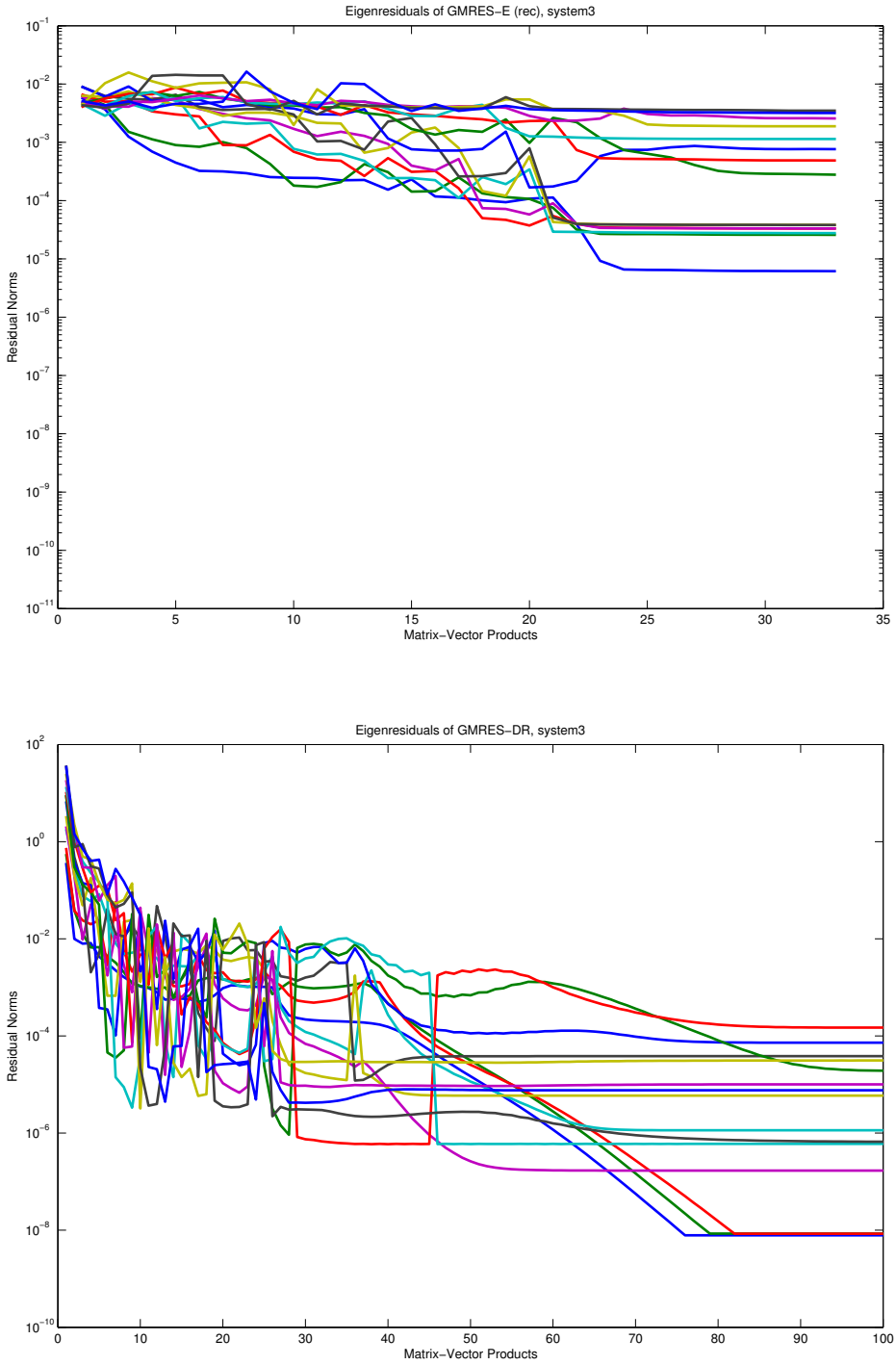


Figure 3.5: Eigenresiduals of GMRES-E(recycled)(top) and GMRES-DR(bottom).

Table 3.6: Some eigenvalue of $A_{(3)}$

Sorted Eigenvalues of A in Order of Magnitude	
λ_1	0.000202036743847
λ_2	0.000376409685278
λ_3	−0.000389932183701
λ_4	−0.000519247933623
λ_5	0.000664420018127
λ_6	−0.000797329272257
λ_7	0.001104429859130
λ_8	0.001967643563859
λ_9	0.002469946787251
λ_{10}	0.003530443729578
⋮	⋮
λ_{499}	4.000751339457972
λ_{500}	4.001384339324436

is much faster and cheaper, we want to be able to take advantage of GMRES-Proj's speed when we can, and only use GMRES-E(recycled) when necessary. Therefore, we propose a new algorithm of GMRES that REUSES the approximate eigenvectors from the previous system to deflate the small eigenvalues from the current system using GMRES-PRoj when it is appropriate. When the projection loses its effectiveness, the algorithm will RECYCLE the approximate eigenvectors from the previous system by adding them to the current Krylov subspace, thus improving them so that they can be useful for deflation once again using GMRES-E(recycled). If the system has changed too much, or the new system is completely unrelated to the previous system, or the new systems are simply too tough to solve as seen in Example 3.5, the algorithm will REGENERATE a new set of approximate eigenvectors to help with deflation. We name the algorithm GMRES-RRR, where the R's stand for REUSE, RECYCLE, and REGENERATE.

We give the user the option to choose which method to use manually, or to let the algorithm decide automatically. In order to help the algorithm select which

Table 3.7: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-2$

System	All G-DR	G-DR/GE(rec)
1	815	815
2	815	680
3	795	660
4	835	800
5	1475	620
6	735	880
7	715	520
8	715	460
9	695	440
10	1055	460
11	1715	1220
12	1095	1500
13	1275	1620
14	1415	1800
15	1495	1380
16	1235	1340
17	1555	1180
18	1235	1260
19	1635	1320
20	2055	1940
Total MVP's	23360	20895
Total CPU Time	36.222	73.834

method to use, we introduce a SWITCHING CRITERION. Define the *accumulated change* from the j^{th} system to the l^{th} system by

$$\mathcal{E} = A_{(l)} - A_{(j)}$$

where $A_{(l)}$ is the coefficient matrix of the current system and $A_{(j)}$ is the matrix from which the approximate eigenvectors were generated. The algorithm will use the norm of $\mathcal{E}$ as the switching criterion. For example: Use GMRES-Proj if $||\mathcal{E}|| < 1.0e-4$, use GMRES-DR if $||\mathcal{E}|| > 1.0e-2$, and use GMRES-E(recycled) otherwise. These bounds are user-defined to accomodate a wider range of applications and suit each user's need for their designs.

Algorithm 3.3 GMRES-RRR

Initialization: Manually assign a value to *method* (1 to 3) or give an *upperBound* and a *lowerBound* for the *switching criterion* so the algorithm can use to assign values to *method*. Choose m and k.
if *method* is not assigned **then**
    Compute *criterion*:= $||A_{(l)} - A_{(j)}||$ if necessary.
    if $y_1, ..., y_k$ are not available OR *criterion* > *upperBound* **then**
        method:=1
    else if *criterion* < *lowerBound* **then**
        method:=2
    else
        method:=3
    end if
end if
for i=1: number of linear systems **do**
    if *method*==1, solve $A_{(i)}x^{(i)} = b^{(i)}$ using GMRES-DR(m,k).
        Save k approximate eigenvectors at the end.
    if *method*==2, solve $A_{(i)}x^{(i)} = b^{(i)}$ using GMRES-Proj.
    if *method*==3, solve $A_{(i)}x^{(i)} = b^{(i)}$ using GMRES-E(m,k)(recycled).
        Save k approximate eigenvectors at the end.
end for

3.2.5 Numerical Results

In order to test the performance of GMRES-RRR, we apply the algorithm to different sequences of linear systems. The size of E is different for each sequence.

Example 3.7. We repeat Example 3.3 by solving the first system using GMRES-DR(25,10) to generate 10 approximate eigenvectors. For the remaining systems, instead of using only GMRES-Proj or only GMRES-E(recycled), we run GMRES-RRR and let the algorithm decide which method to use. As for the switching criterion, the algorithm will use GMRES-Proj if $||\mathcal{E}|| < 1.0e-4$, use GMRES-DR if $||\mathcal{E}|| > 1.0e-2$, and use GMRES-E(recycled) otherwise. The results in Table 3.8 show that the algorithm was able to re-use the approximate eigenvectors generated from the first system for deflation using GMRES-Proj until system 12, where as GMRES-Proj begins to lose effectiveness in previous example, GMRES-RRR recycles and improves the approximate eigenvectors via GMRES-E(recycled), so that for the subsequent systems, they become good enough that GMRES-Proj is effective once again. By comparing the CPU times and the tota matrix-vector products in Table 3.8, we see that GMRES-RRR has successfully taken full advantage of the speed of GMRES-Proj and only need to call GMRES-E(recycled) once.

Next, we test the new algorithm for solving a sequence of linear systems when the change from one system to the next is slightly larger that those seen in Example 3.7. We anticipate that since the approximate eigenvectors will be losing accuracy more quickly, GMRES-RRR will need to call GMRES-E(recycled) more often.

Table 3.8: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-5$

System	G-DR Only	G-Proj	G-E(rec)	GMRES-RRR
1	1195	1195	1195	1195 (G-DR)
2	1255	660	660	660 (G-Proj)
3	1210	615	630	615 (G-Proj)
4	1225	645	660	645 (G-Proj)
5	1255	660	660	660 (G-Proj)
6	1255	645	660	645 (G-Proj)
7	1210	660	660	660 (G-Proj)
8	1255	660	660	660 (G-Proj)
9	1210	645	630	645 (G-Proj)
10	1255	690	660	690 (G-Proj)
11	1225	660	645	660 (G-Proj)
12	1210	675	660	660 (G-E(rec))
13	1225	690	660	660 (G-Proj)
14	1240	720	645	630 (G-Proj)
15	1180	810	570	570 (G-Proj)
16	1180	900	630	645 (G-Proj)
17	1225	1050	630	645 (G-Proj)
18	1255	1065	645	645 (G-Proj)
19	1270	1245	660	675 (G-Proj)
20	1240	1440	645	690 (G-Proj)
Total MVP's	24575	16330	13465	13555
Total CPU Time	24.591s	9.468s	34.599s	9.306s

Example 3.8. We repeat Example 3.7 solving $A_{(i)}x^{(i)} = b^{(i)}$ for $i = 1, 2, ..., 20$ with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 4.0e-5$ using the same switching criterion. As anticipated, since the systems are changing more rapidly, GMRES-Proj loses effectiveness more quickly, and as a result, our algorithm ends up using GMRES-E(recycled) more often. Table 3.9. For comparison purposes, we include the results obtained by running only GMRES-Proj after the first system. As the matrices change more and more from the original system, GMRES-Proj's effectiveness decreases rapidly, and eventually, GMRES-Proj fails to converge when solving systems $13, ..., 20$. This demonstrate one of the major strengths of GMRES-RRR: by recycling the approximate eigenvectors, even only once in every few system, GMRES-RRR is able to extend the usefulness of those vectors for deflation to in order to minimize the total cost in matrix-vector products while speeding up the convergence for solving linear systems. To illustrate the actions of GMRES-RRR, we look at the residual norms on the eigenvectors at the beginning of each linear system in Figure 3.6. When GMRES-RRR only calls GMRES-Proj, as the systems change, the eigenvectors become less accurate. Once the switching criterion is met, for system 4, GMRESS-RRR calls GMRES-E(recycled) which improves the accuracy of some of the eigenvectors so they can meet the condition to be re-used by GMRES-Proj once again for system 5 and 6. The drops in the plots correspond to the use of GMRES-E(recycled).

During several experiments, we noticed that sometimes, even though the accumulated change is still small enough to meet the criterion to use GMRES-Proj, the convergence rate suggests that the algorithm should not have used GMRES-Proj. For instance, in Example 3.8, GMRES-RRR uses GMRES-Proj to solve system 9, 15, and 18, which incurs higher cost in terms of matrix-vector products. This is not a dramatic spike in expenses, but it does indicate that for some particular cases, the user needs to introduce a stricter criteron for switching, especially when the major computational cost is matrix-vector products.

Table 3.9: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 4.0e-5$

System	G-DR Only	G-Proj	G-E(rec)	GMRES-RRR
1	1195	1195	1195	1195 (G-DR)
2	1255	660	660	660 (GProj)
3	1225	630	630	630 (GProj)
4	1240	660	660	660 (G-E(rec))
5	1270	1020	660	660 (GProj)
6	1255	1665	660	660 (GProj)
7	1225	1845	660	660 (G-E(rec))
8	1255	3330	660	705 (GProj)
9	1225	4140	630	930 (GProj)
10	1255	5685	675	675 (G-E(rec))
11	1240	7710	645	645 (GProj)
12	1225	9570	660	675 (GProj)
13	1240	10500	660	660 (G-E(rec))
14	1255	10500	645	645 (GProj)
15	1180	10500	600	885 (GProj)
16	1210	10500	630	630 (G-E(rec))
17	1255	10500	630	660 (GProj)
18	1270	10500	660	825 (GProj)
19	1285	10500	660	660 (G-E(rec))
20	1270	10500	660	660 (GProj)
Total MVP's	24830	122110	13540	14380
Total CPU Time	24.63s	60.841s	37.202s	16.264s

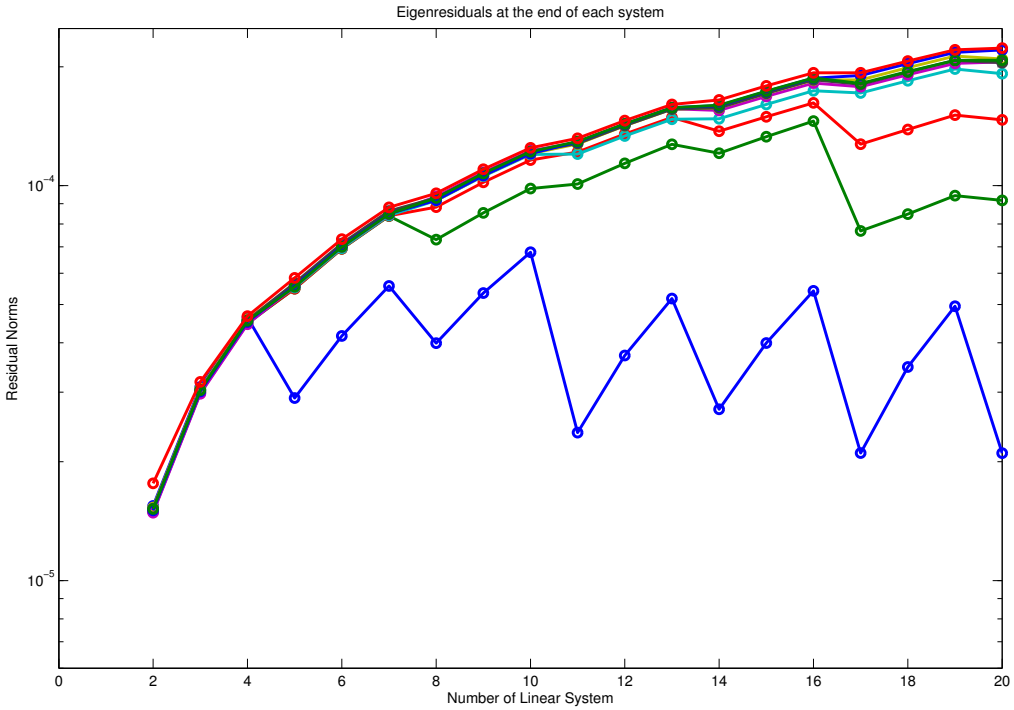


Figure 3.6: Eigenresiduals at the beginning of each linear system.

Instead of considering only the size of the accumulated change from the previous system, we now also look at how accurately the eigenvectors of the previous system, $A_{(j)}$, approximate the eigenvectors of the current system, $A_{(l)}$. Consider the eigenresidual vectors

$$r_i = A_{(l)}y_i^{(j)} - \rho_i^{(l)}y_i^{(j)}, \text{ for } i = 1, 2, ..., k$$

where $y_i^{(j)}$'s are approximate eigenvectors of a previous matrix $A_{(j)}$ and $\rho_i^{(l)}$ are the Rayleigh Quotients

$$\rho_i^{(l)} = \frac{(y_i^{(j)})^* A_{(l)} y_i^{(j)}}{(y_i^{(j)})^* y_i^{(j)}}$$

Since, the norm of r_i indicates whether $y_i^{(j)}$ is a good approximate eigenvector of $A_{(l)}$, in order to incorporate this into a switching criterion for GMRES-RRR, define

$$\text{maxri} = \max_{i=1:k} ||r_i||$$

GMRES-RRR will use GMRES-Proj when the accumulated change $||\mathcal{E}|| < 1.0e-4$ AND maxri $< 1.0e-4$. These bounds are also user-defined. With this stricter switching criterion, GMRES-RRR would be able to avoid a scenario where the accumulated change is small but the eigenvectors have been altered significantly that GMRES-Proj is no longer effective. Example 3.9 demonstrates the impact of having stricter switching criteria.

Example 3.9. We repeat Example 3.8 with this new switching criterion: the algorithm will use GMRES-PRoj if $||\mathcal{E}|| < 1.0e-4$ AND maxri $< 1.0e-4$, use GMRES-DR if $||\mathcal{E}|| > 1.0e-2$, and use GMRES-E(recycled) otherwise. As anticipated, from system 8 to system 9, even though the accumulated change is still less than $1.0e-4$ but the eigenvectors are not good enough for projection, the stricter switching criterion signals GMRES-RRR to use GMRES-E(recycled) instead. The results in Table 3.10 show that by using a stricter switching condition, GMRES-RRR is able to minimize the cost in terms of matrix-vectors product at the expense

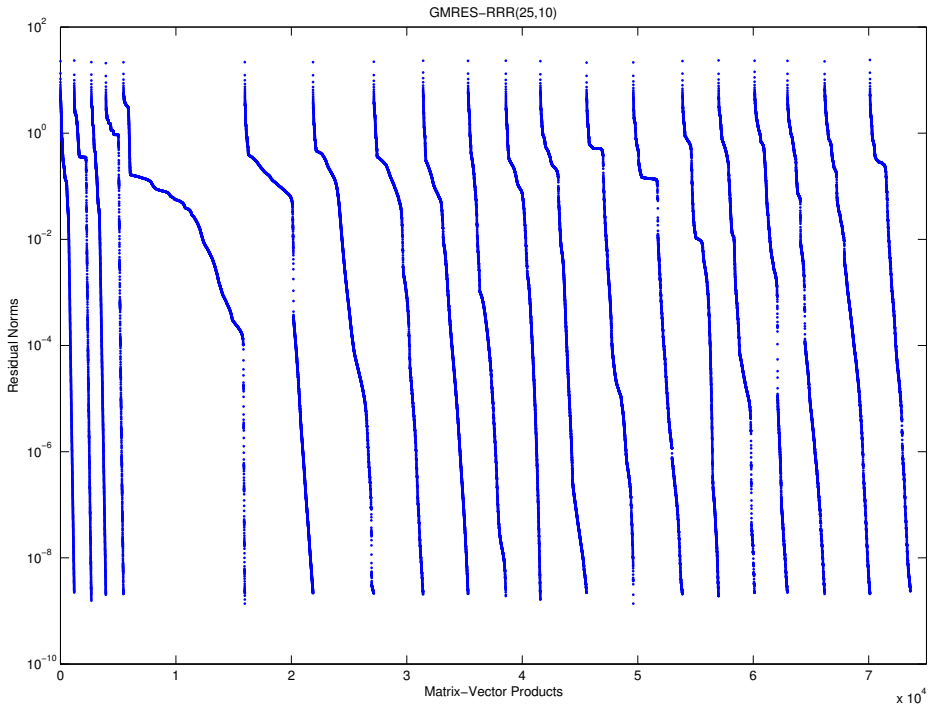
of CPU time. This is one of the reasons why we let the users choose the switching conditions depends on what's more important to their models: minimizing the total matrix-vector products or optimizing the overall speed for solving each sequence of linear systems.

Table 3.10: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 4.0e-5$

System	G-DR Only	GMRES-RRR	GMRES-RRR (strict)
1	1195	1195 (G-DR)	1195 (G-DR)
2	1255	660 (G-Proj)	660 (G-Proj)
3	1225	630 (G-Proj)	630 (G-Proj)
4	1240	660 (G-E(rec))	660 (G-E(rec))
5	1270	660 (G-Proj)	660 (G-Proj)
6	1255	660 (G-Proj)	660 (G-Proj)
7	1225	660 (G-E(rec))	660 (G-E(rec))
8	1255	705 (G-Proj)	705 (G-Proj)
9	1225	930 (G-Proj)	630 (G-E(rec))
10	1255	675 (G-E(rec))	660 (G-E(rec))
11	1240	645 (G-Proj)	645 (G-E(rec))
12	1225	675 (G-Proj)	660 (G-E(rec))
13	1240	660 (G-E(rec))	660 (G-E(rec))
14	1255	645 (G-Proj)	645 (G-E(rec))
15	1180	885 (G-Proj)	630 (G-E(rec))
16	1210	630 (G-E(rec))	630 (G-E(rec))
17	1255	660 (G-Proj)	630 (G-E(rec))
18	1270	825 (G-Proj)	660 (G-E(rec))
19	1285	660 (G-E(rec))	660 (G-E(rec))
20	1270	660 (G-Proj)	660 (G-E(rec))
Total MVP's	24830	14380	13600
Total CPU Time	24.63s	16.264s	26.084s

Next, we test GMRES-RRR on a sequence of systems where the change from one system to the next is large enough that the approximate eigenvectors formed by GMRES-E(recycled) are not effective enough that a set of new eigenvectors must be regenerated from scratch by GMRES-DR so that they can reach the necessary level of accuracy to be good enough for deflation. Note that these highly perturbed systems generated for this example are for demonstrations only. In general, if the sequence comes from a practical model, the change from one system to the next will be small enough that GMRES-RRR will usually only need to switch between GMRES-E(recycled) and GMRES-Proj.

Example 3.10. We repeat Example 3.7 solving $A_{(i)}x^{(i)} = b^{(i)}$ for $i = 1, 2, ..., 20$ with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e - 2$ using the same switching criterion. This large perturbation will signal GMRES-RRR(25,10) to use GMRES-DR for every system. Having such a large random perturbation sometimes results in linear systems that are very difficult for GMRES-DR(25,10) to solve, such as system 5. The plot of the residual norms in Figure 3.7(top) illustrates GMRES-DR(25,10)'s struggle. One simple way to address this issue is to use a larger subspace. However, suppose the maximum size of the subspace is restricted to 25, we can address the convergence issue by using more approximate eigenvectors for deflation. The plot of the residual norms in Figure 3.7(bottom) indicates that by using 10 Arnoldi vectors and 15 approximate eigenvectors, GMRES-RRR(25,15) was able to solve every linear system in the sequence while keeping the maximum size of the subspace at 25. The cost in terms of matrix-vector products is given in Table 3.11. We also include the cost for solving the sequence using GMRES-RRR(30,10) and GMRES-RRR(35,15) to show how the overal performance can sometimes be improved simply by changing the size of the subspace and/or increase the number of eigenvectors used for deflation.



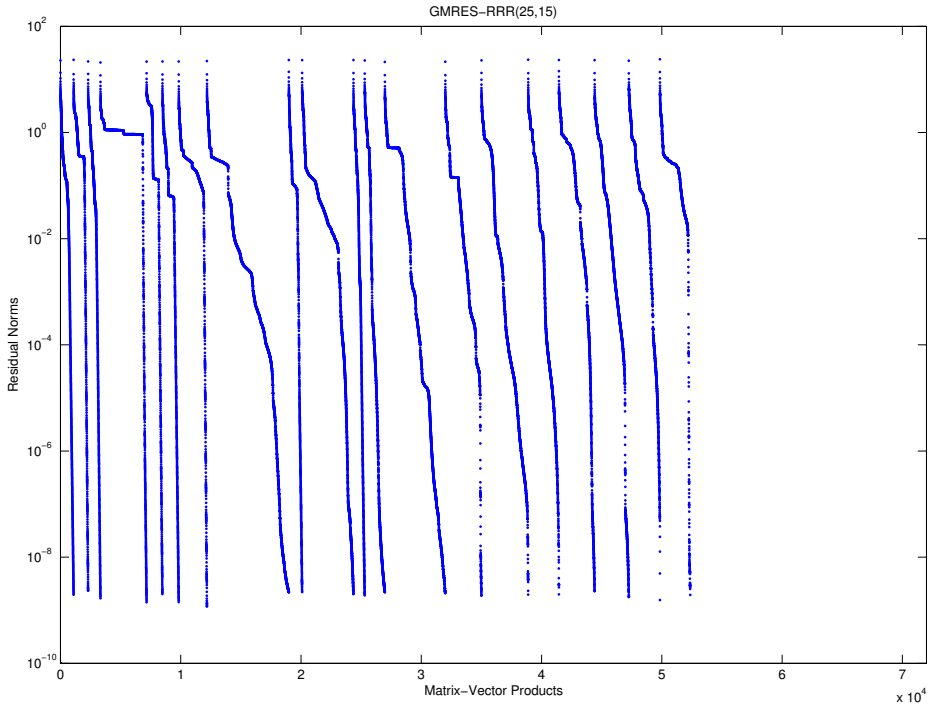


Figure 3.7: GMRES-RRR(25,10) and GMRES-RRR(25,15).

Table 3.11: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-2$

System	G-RRR(25,10)	G-RRR(25,15)	G-RRR(30,10)	G-RRR(35,15)
1	1195	1095	1030	815
2	1480	1205	2130	815
3	1255	1025	1070	795
4	1525	3835	2890	835
5	10510	1315	1110	1475
6	5905	1355	3730	735
7	5260	2345	2650	715
8	4270	6805	2410	715
9	3895	1105	2710	695
10	3265	4265	2750	1055
11	2995	935	2870	1715
12	4000	1675	2490	1095
13	4045	5025	2530	1275
14	4255	3005	2110	1415
15	3130	3885	2570	1495
16	3115	2555	2070	1235
17	2845	2965	2970	1555
18	3220	2845	3190	1235
19	3925	2585	3670	1635
20	3505	2516	3910	2055
Total MVP's	73595	52341	50860	23360
Total CPU Time	74.452s	69.027s	49.434s	29.008s

CHAPTER FOUR

Comparing GMRES-RRR to Subspace Recycling

The process of saving portions of one Krylov subspace to speed up the convergence for solving the next systems is sometimes referred to as *Krylov subspace recycling*. We will compare our new approach of recycling to a method developed by Parks and De Sturler in [22] called GCRO-DR, which is a combination of GMRES-DR and GCRO – the *outer* version of the *Generalize Conjugate Residual* method [7]. We will give a brief description of GCRO-DR and a numerical example to see how well GMRES-RRR performs compared GCRO-DR.

Similar to GMRES-RRR, when solving the first system or when the approximate eigenvectors are not available, GCRO-DR will use GMRES-DR to solve the linear equations and generate k harmonic Ritz vectors. For the next system, instead of adding the approximate eigenvectors into the subspace like GMRES-E(recycled) does, GCRO-DR uses the approximate eigenvectors to form a new linear operator then generates the Krylov subspace with this operator instead of A.

4.1 Implementation of GCRO-DR

Suppose we have solved the j^{th} system, $A_{(j)}x^{(j)} = b^{(j)}$, and obtained k approximate eigenvectors $Y_k = [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_k]$. GCRO-DR then computes the matrices U_k and $C_k \in \mathbb{C}^{n\times k}$ such that

$$A_{(j+1)}U_k = C_k \text{ and } C_k^* C_k = I_k$$

Next, the method computes the optimal solution over the subspace range(U_k) as

$$x = x_0 + U_k C_k^* r_0$$

and set

$$r = r_0 - C_k C_k^* r_0 \text{ and } v_1 = r/||r||$$

Next, GCRO-DR generates the Krylov subspace of dimension $m-k+1$ using the matrix $(I - C_k C_k^*)A$ which results in the Krylov subspace

$$\mathcal{K} = span\{r_0, (I - C_k C_k^*)Ar_0, \left((I - C_k C_k^*)A\right)^2 r_0, ..., \left((I - C_k C_k^*)A\right)^{m-k+1} r_0\} \quad (4.1)$$

and the Arnoldi relations

$$(I - C_k C_k^*)AV_{m-k} = V_{m-k+1}\underline{H}_{m-k} \quad (4.2)$$

which can be rewritten as

$$A\begin{bmatrix} U_k & V_{m-k} \end{bmatrix} = \begin{bmatrix} C_k & V_{m-k+1} \end{bmatrix} \begin{bmatrix} I_k & B_k \\ 0 & \underline{H}_{m-k} \end{bmatrix} \quad (4.3)$$

where $B_k = C_k^* A V_{m-k}$. Compute the diagonal matrix D_k such that $\widetilde{U}_k = U_k D_k$ has unit columns and define the following matrices

$$\widehat{V}_m = \begin{bmatrix} \widetilde{U}_k & V_{m-k} \end{bmatrix} \quad (4.4)$$

$$\widehat{W}_{m+1} = \begin{bmatrix} C_k & V_{m-k+1} \end{bmatrix} \quad (4.5)$$

$$\underline{G}_m = \begin{bmatrix} D_k & B_k \\ 0 & \underline{H}_{m-k} \end{bmatrix} \quad (4.6)$$

the relation (4.3) can be rewritten as

$$A\widehat{V}_m = \widehat{W}_{m+1}\underline{G}_m \quad (4.7)$$

Let $\beta = ||r_0||$, GCRO-DR then solves the $(m+1) \times m$ least-square problem

$$\min ||\beta e_{k+1} - \underline{G}_m d||$$

for $d \in \mathbb{C}^m$ and forms the approximate solution of the linear system

$$\hat{x} = x_0 + \widehat{V}_m d$$

.

Similar to GMRES-E and GMRES-DR, in order to compute approximate eigenvectors, GCRO-DR uses a version of the harmonic Rayleigh-Ritz procedure to solve the generalized eigenvalue problem

$$\underline{G}_m^* \underline{G}_m g_i = \widetilde{\theta}_i \underline{G}_m^* \widehat{W}_{m+1}^* \widehat{V}_m g_i \tag{4.8}$$

The approximate eigenvectors are obtained as harmonic Ritz vectors $\tilde{y}_i = \widehat{V}_m g_i$.

As noted by the author in [22], GCRO-DR is slightly more expensive than GMRES-DR since it needs to store the approximate eigenvectors, as does GMRES-E(recycled). In that regard, GMRES-RRR and GCRO-DR will be similarly expensive in terms of matrix-vector products. However, GMRES-RRR can shine when it is able to use GMRES-Proj instead of GMRES-E(recycled).

4.2 Numerical Results

The following examples help demonstrate how GMRES-RRR performs compared to GCRO-DR in terms of matrix-vector products, and CPU time.

Example 4.1. We now compare the performance of GMRES-RRR(25,10) to GCRO-DR(25,10) for solving a sequence of linear systems where the change from one system to the next is relatively small, $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-5$, by repeating the experiment in Example 3.7. The costs in terms of matrix-vector products are given in Table 4.1. Looking at computational costs in terms of matrix-vector products, GMRES-RRR(25,10) performs comparably to the more standard approach, GCRO-DR(25,10). Furthermore, GCRO-DR is generally more expensive and more complicated to implement while GMRES-RRR is much simpler to program and is able to take advantage of the speed of GMRES-Proj, especially when the change in the systems is small enough.

Example 4.2. This example compares the performance of GMRES-RRR(25,10) to GCRO-DR(25,10) for solving a sequence of linear systems where the change from

Table 4.1: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-5$

System	GMRES-RRR(25,10)	GCRO-DR(25,10)
1	1195 (G-DR)	1195
2	660 (G-Proj)	660
3	615 (G-Proj)	615
4	645 (G-Proj)	645
5	660 (G-Proj)	660
6	645 (G-Proj)	645
7	660 (G-Proj)	660
8	660 (G-Proj)	660
9	645 (G-Proj)	630
10	690 (G-Proj)	660
11	660 (G-Proj)	645
12	660 (G-E(rec))	660
13	660 (G-Proj)	645
14	630 (G-Proj)	630
15	570 (G-Proj)	570
16	645 (G-Proj)	630
17	645 (G-Proj)	630
18	645 (G-Proj)	645
19	675 (G-Proj)	660
20	690 (G-Proj)	645
Total MVP's	13555	13390
Total CPU Time	9.306s	47.102s

one system to the next is larger, $||E|| = ||A_{(i)} - A_{(i-1)}|| = 4.0e-5$, by repeating the experiment in Example 3.8. The costs in terms of matrix-vector products are given in Table 4.2.

Table 4.2: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 4.0e-5$

System	GMRES-RRR(25,10)	GCRO-DR(25,10)
1	1195 (G-DR)	1195
2	660 (G-Proj)	660
3	630 (G-Proj)	630
4	660 (G-E(rec))	645
5	660 (G-Proj)	660
6	660 (G-Proj)	660
7	660 (G-E(rec))	660
8	705 (G-Proj)	660
9	930 (G-Proj)	630
10	675 (G-E(rec))	660
11	645 (G-Proj)	645
12	675 (G-Proj)	660
13	660 (G-E(rec))	660
14	645 (G-Proj)	645
15	885 (G-Proj)	570
16	630 (G-E(rec))	630
17	660 (G-Proj)	630
18	825 (G-Proj)	645
19	660 (G-E(rec))	660
20	660 (G-Proj)	660
Total MVP's	14380	13465
Total CPU Time	16.264s	49.329s

Example 4.3. Finally, we compare the performance of GMRES-RRR(25,10) to GCRO-DR(25,10) for solving a sequence of linear systems where the change from one system to the next is very large, $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-2$, by repeating the experiment in Example 3.10 The costs in terms of matrix-vector products are given in Table 4.3. GCRO-DR(25,10) appears to be struggle in a way that is similar to GMRES-E(recycled)(25,10) in Example 3.5 , which indicates that sometimes recycling might not be very effective, and new eigenvectors need to be regenerated. As mentioned, in the previous chapter, instead of regenerating new eigenvectors, we could use a slightly larger subspace and/or deflate more eigenvalues.

Table 4.3: Matrix-Vector Products with $||E|| = ||A_{(i)} - A_{(i-1)}|| = 1.0e-2$

System	GMRES-RRR(25,10)	GCRO-DR(25,10)
1	1195	1195
2	1480	1680
3	1255	840
4	1525	1125
5	10510	9135
6	5905	3465
7	5260	10500
8	4270	10500
9	3895	3915
10	3265	3105
11	2995	3255
12	4000	9480
13	4045	10500
14	4255	10500
15	3130	8430
16	3115	4710
17	2845	4965
18	3220	4965
19	3925	4485
20	3505	5250
Total MVP's	73595	112000
Total CPU Time	74.452s	669.663s

4.3 Some Analysis of Subspace Recycling

Here we discuss how GMRES-E and GCRO-DR improve approximate eigenvectors that they are given even though they are focusing mainly on solving linear equations. It is assumed that the method starts with approximate eigenvectors from a different matrix so that we do not have the Krylov properties of GMRES-DR. We have seen examples in which both GMRES-E and GCRO-DR improve the approximate eigenvectors enough to be effective even though the matrix is changing. We now look at what makes them work. For GMRES-E, the situation is simpler, because the GMRES-E subspace contains a Krylov subspace with the matrix A. Therefore it is not surprising that this subspace helps improve approximate eigenvectors of A. However, this improvement is much less than in GMRES-DR. This is because unlike with GMRES-DR, GMRES-E does not have Krylov subspaces with the approximate eigenvectors as starting vectors. For GCRO-DR, the operator that is used to build a Krylov subspace is $N \equiv (I - CC^*)A$, where $C = AU$, with the columns of U spanning the subspace of approximate eigenvectors $Span\{y_1, y_2 \ldots, y_k\}$. Since the Krylov subspace is not generated with A, it seems that this is the wrong operator to be using to improve eigenvectors of A. However, there are relationships between the eigenpairs of N and those of A that show it is possible for a Krylov subspace with N to improve eigenvectors of A.

The first proposition assumes that A is symmetric. It is also assumed that we have the k exact eigenvectors of A for U, so that C has k exact eigenvectors of A as columns. Then the eigenvectors of N are the same as those of A. However, the k eigenvalues corresponding to the k eigenvectors in U are all at 0. The other $n - k$ eigenvalues do not change. So this points out that there is potential for a Krylov subspace with N to help compute eigenvectors of A, since it has the correct eigenvectors in this idealized situation.

Proposition 4.1. *If A is symmetric and $C = [z_1 z_2 \dots z_k]$, then N has k eigenvalues at 0 with associated eigenspace $Span\{z_1, z_2, \dots, z_k\}$. Also, for $j > k$, if (λ_j, z_j) is an eigenpair of A, then it is also an eigenpair of N.*

Proof. We use the fact that since A is symmetric and we have exact eigenvectors, the columns of U can be just the eigenvectors of A. For $i = 1 \dots k$, $Nz_i = (I - CC^*)Az_i = \lambda_i(I - CC^*)z_i = \lambda_i(z_i - Ce_i) = \lambda_i(z_i - z_i) = 0$, where e_i is the ith coordinate vector.

For $j > k$, $Nz_j = (I - CC^*)Az_j = \lambda_j(I - CC^*)z_j = \lambda_j(z_j - 0) = \lambda_j z_j$. □

Next we consider A nonsymmetric, and again assume that we have the exact eigenvectors of A. While the columns of U span the subspace $Span\{z_1, \dots, z_k\}$, the columns cannot equal to $z_1, \dots, z_k$ as before because U has orthonormal columns.

Proposition 4.2. *Assume the columns of U and thus C span $Span\{z_1, z_2 \dots, z_k\}$. Then N has k eigenvalues at 0 with associated eigenspace $Span\{z_1, z_2, \dots, z_k\}$. Also, for $j > k$, then the eigenvalue λ_j is an eigenvalue of N, with associated eigenvector a combination of z_j and $z_1, \dots z_k$.*

Proof. . It is easy to show that $Nz_i = 0$ for i from 1 to k, because the columns of C span the subspace $S = Span\{z_1, \dots, z_k\}$. Thus $(I - CC^*)$ projects all of S to the zero vector. Next, for $j > k$, $N(\alpha_1 z_1 + \dots + \alpha_k z_k + \alpha_j z_j) = 0 + \dots + 0 + \alpha_j \lambda_j w$, where w is a combination of $z_1, \dots, z_k$ and z_j. Then if the α_i's are chosen to match this last combination, we have that $\alpha_1 z_1 + \dots + \alpha_k z_k + \alpha_j z_j$ is an eigenvector of N. □

Next, the case of not having exact eigenvectors of A is considered. We let the approximate eigenvectors of A be y_i's.

Proposition 4.3. *Assume the columns of U span the subspace $Span\{y_1, y_2 \dots y_k\}$. Then N has k eigenvalues at 0 with associated eigenspace $Span\{y_1, y_2, \dots y_k\}$.*

Proof. . We will show that $Ny_i = 0$ for i from 1 to k. Since the columns of C span the subspace $S = Span\{Ay_1, \dots, Ay_k\}$, $(I - CC^*)$ projects all of S to the zero vector. So for $i = 1, \dots, k$, $Ny_i = (I - CC^*)(Ay_i) = 0$. □

With the y_i's as eigenvectors, the operator N has approximations to the correct eigenvectors. However, more study is needed in order to fully understand how GCRO-DR works.

CHAPTER FIVE

Further Considerations and Challenges

5.1 Considerations

GMRES-RRR is designed to work for the most general case where the change from one system to the next is a random perturbation. However, in many applications, the changes can be more specific and have properties that GMRES-RRR could take advantage to be more efficient. We include the results from previous experiments to demonstrate this idea.

Consider the one-dimensional Poisson problem with Dirichlet boundary conditions

$$-w''(x) = f(x), \text{ for } x \in (0,1) \tag{5.1}$$

$$w(0) = w(1) = 1 \tag{5.2}$$

Using a second-order, central difference scheme over a general grid $[x_0, x_1, ..., x_{N+1}]$, we have the following equations

$$l_{i-1}w_{i-1} + m_i w_i + u_i w_{i+1} = f(x_i) \tag{5.3}$$

$$l_{i-1} = -\frac{2}{h_{i-1}(h_i + h_{i-1})} \tag{5.4}$$

$$m_i = \frac{2}{h_i h_{i-1}} \tag{5.5}$$

$$w_i = -\frac{2}{h_i(h_i + h_{i-1})} \tag{5.6}$$

where $i = 1, ..., N$, $h_i = x_{i+1} - x_i$, $x_0 = 0, x_{N+1} = 1$, $w_0 = w_{N+1} = 1$, and $w(x_i) \approx w_i$. This discretization results in a system of linear equations $Tw = b$ where $w = (w_1, ..., w_N)^T$, $b = (f(x_1), ..., f(x_N))^T$, and T is a tridiagonal matrix of the form

$$T = \begin{pmatrix} m_1 & u_1 & & & & \\ l_1 & m_2 & u_2 & & & \\ & l_2 & m_3 & u_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & l_{N-2} & m_{N-1} & u_{N-1} \\ & & & & l_{N-1} & u_N \end{pmatrix}$$

When the grid is uniform, the resulting matrix is

$$T \equiv T_U = \frac{1}{h^2} \begin{pmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{pmatrix}$$

whose eigenvalues and eigenvectors can be computed explicitly,

$$\lambda_i = \frac{1}{h^2}\left(1 + 2\cos\left(\frac{i\pi}{N+1}\right)\right)$$
$$z_i = \left(\sin\left(\frac{i\pi}{N+1}\right), ..., \sin\left(\frac{iN\pi}{N+1}\right)\right)^T$$

The matrices obtained from non-uniform grids can be viewed as perturbations of the matrix from a uniform grid, that is $T = T_U + E$, where E is a real, tridiagonal perturbation matrix. As seen in previous examples, when E is small enough that the eigenvectors of T_U are good approximate eigenvectors of T, we can use them to speed up the convergence for solving linear equations using GMRES-Proj [19]. Furthermore, we can take advantage of the fact that since all of these matrices come from the same differential equation, their eigenvectors are closely related. We can use the eigenvectors from one grid to generate approximate eigenvectors for another grid via an interpolation method. This could be implemented in order to further increase the effectiveness of GMRES-RRR.

5.2 Challenges

Due to the loss of the Krylov properties, GMRES-E(recycled) can only improve the accuracy of the recycled eigenvectors to a certain degree. In general, these eigenvectors – even though not very accurate – are accurate *enough* that they can still help to speed up the convergence for solving the linear equations. However, as seen in one of the examples, there are situations where the eigenvectors needs to be just a little more accurate before they can be effective for deflation, and since GMRES-E(recycled) can improve them no further, the method may fail to converge. One potential fix, instead of having to regenerate a new set of eigenvectors, is to implement a sub-routine that improves the eigenvectors outside of the subspace generated by GMRES-E(recycled) using a version Arnoldi-E [11]. With the extra help, the eigenvectors will be improved to the accuracy needed to help with solving the linear equations; however, this requires generating additional Krylov subspaces, which incurs extra cost. Further study and experiments are required in order to determine the exact expense.

CHAPTER SIX

Conclusion and Future Work

With the updated version of GMRES-E, we now have a method that is capable of recycling approximate eigenvectors from a previous system to speed up the convergence of GMRES for solving the subsequent systems. We combine this new approach, GMRES-E(recycled), with GMRES-Proj and GMRES-DR into a new algorithm to solve a sequence of linear systems, GMRES-RRR. The default algorithm checks a user-defined criteria and switches between methods to take advantage of each method's best features: to REUSE eigenvectors with GMRES-Proj, to RECYCLE eigenvectors with GMRES-E(recycled), or to REGENERATE new eigenvectors with GMRES-DR. We also discuss some guidelines to help the user choose different switching criteria, giving them the flexibility to fine-tune GMRES-RRR to suit their models.

The numerical results indicate that GMRES-RRR can work very well, and comparable to an industry-standard approach, especially when the change from one system to the next is small enough.

For future work, we plan to apply our simple approach of subspace recycling to solving sequences of linear systems that come from practical problems in various disciplines such as computational fluid dynamics, financial derivatives, and other nonlinear models.

BIBLIOGRAPHY

[1] W. E. Arnoldi, *The principle of minimized iteration in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17-29.

[2] O. Axelsson, *A survey of preconditioned iterative methods for linear systems of equations*, BIT, 25 (1985), pp. 166-187.

[3] J. Baglama, D. Calvetti, G. H. Golub, and L. Reichel, *Adaptively preconditioned GMRES algorithms*, SIAM J. Sci. Comput., 20 (1998), pp. 243-269.

[4] M. A. Beauregard and Q. Sheng, *Solving degenerate quenching- combustion equations by an adaptive splitting method on evolving grids*, Comput. Struct., doi:10.1016/j.compstruc.2012.10.014, 2012.

[5] M. A. Beauregard and Q. Sheng, *An adaptive splitting approach for the quenching solution of reaction-diffusion equations over nonuniform grids*, J. Comput. Appl. Math., 241 (2013), pp. 30-44.

[6] K. Burrage and J. Ethel, *On the performance of various adaptive preconditioned GMRES strategies*, Num. Lin. Alg. with Appl., 5 (1998), pp. 101-121.

[7] E. De Sturler, *Nested Krylov methods based on GCR*, J. Comput. Appl. Math., 67 (1996), pp. 1541.

[8] E. de Sturler, *Truncation strategies for optimal Krylov subspace methods*, SIAM J. Numer. Anal., 36 (1999), pp. 864889.

[9] J. Erhel, K. Burrage, and B. Pohl, *Restarted GMRES preconditioned by deflation*, J. Comput. Appl. Math., 69 (1996), pp. 303-318.

[10] S. A. Kharchenko and A. Y. Yeremin, *Eigenvalue translation based preconditions for the GMRES(k) method*, Num. Lin. Alg. with Appl., 2 (1995), pp. 51-77.

[11] R. B. Morgan, *Computing interior eigenvalues of large matrices*, Linear Algebra Appl., 154-156 (1991), pp. 289-309.

[12] R. B. Morgan, *A restarted GMRES method augmented with eigenvectors*, SIAM J. Matrix Anal. Appl., 16(1995), pp. 1154-1171.

[13] R. B. Morgan, *Implicitly restarted GMRES and Arnoldi methods for nonsymmetric systems of equations*, SIAM J. Matrix Anal. Appl., 21 (2000), No. 4, pp. 11121135.

[14] R. B. Morgan, *GMRES with deflated restarting*, SIAM J. Sci. Comput., 24(2002), pp. 20-37.

[15] R. B. Morgan, *Restarted block GMRES with deflation of eigenvalues*, Applied Numerical Mathematics, vol. 54 (2005), pp. 222-236.

[16] R. B. Morgan, D. Darnell, W. Wilcox, *Deflated GMRES for systems with multiple shifts and multiple right-hand sides*, Lin. Alg. Appl., vol. 429 (2008), pp. 2415-2434.

[17] R. B. Morgan, A. Rehim, and W. Wilcox, *Improved seed methods for symmetric positive definite linear equations with multiple right-hand sides*. Numer. Linear Alg. Appl., vol. 21 (2014), pp. 453-471.

[18] R. B. Morgan and M. Zeng, *Harmonic projection methods for large non-symmetric eigenvalue problems*, Num. Lin. Alf. with Appl., 5 (1998), pp. 33-55.

[19] H. V. Nguyen, M. A. Beauregard and R. B. Morgan, *Improving the Speed of Convergence of GMRES for Certain Perturbed Tridiagonal Systems*, IEEE 45th Southeastern Symposium on System Theory(SSST), (2013), pp. 63-67.

[20] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Num. Lin. Alg. with Appl., 2 (1995), pp. 115-133.

[21] B. N. Parlett, *The Symmetric Eigenvalue Problem*, PrenticeHall, Englewood Cliffs, NJ, (1980).

[22] M. L. Parks, E. De Sturler, G. Mackey, D. D. Johnson, and S. Maiti, *Recycling Krylov subspaces for sequences of linear systems*, SIAM J. Sci. Comput., Vol. 28, No. 5 (2006), pp. 1651-1674.

[23] Y. Saad, *Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices*, Linear Algebra Appl., 34 (1980), pp. 269-295.

[24] Y. Saad, *Krylov subspace methods for solving large unsymmetric linear systems*, Math. Comp. 87 (1981), pp. 105-126.

[25] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Statist. Comput., 14 (1993), pp. 461-469.

[26] Y. Saad, *Iterative Methods for Sparse Linear System 2nd edition*, SIAM, Philadelphia, PA, 2003.

[27] Y. Saad and M. H. Schultz, *GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7(1986), pp.856-869.

[28] G. L. G. Sleijpen and H. A. van der Vorst, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 401425.

[29] A. Stathopoulos, Y. Saad, and C. Fischer, *Robust preconditioning of large, sparse, symmetric eigenvalue problems*, J. compute. Apple. Math., 64 (1995), pp. 197-215.