UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Galerkin Projection Methods for Solving Multiple Linear Systems

Tony F. Chan
Michael K. Ng

September 1996

CAM Report 96-31

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

Galerkin Projection Methods for Solving Multiple Linear Systems

Tony F. Chan* Michael K. Ng†

25 September 1996

Abstract

In this paper, we consider using Galerkin projection methods for solving multiple linear systems $A^{(i)}x^{(i)} = b^{(i)}$, for $1 \leq i \leq s$, where the coefficient matrices $A^{(i)}$ and the right-hand sides $b^{(i)}$ are different in general. In particular, we focus on the seed projection method which generates a Krylov subspace from a set of direction vectors obtained by solving one of the systems, called the seed system, by the conjugate gradient (CG) method and then projects the residuals of other systems onto the generated Krylov subspace to get the approximate solutions. The whole process is repeated until all the systems are solved. Most papers in the literature [6, 19, 21, 23, 24] considered only the case where the coefficient matrices $A^{(i)}$ are the same but the right-hand sides are different. We extend the method to solve multiple linear systems with varying coefficient matrices and right-hand sides. We also analyze the method and extend the theoretical result of the projection method for solving linear systems with multiple right-hand sides given in Chan and Wan [6]. A theoretical error bound is given for the approximation obtained from a projection process onto a Krylov subspace generated from solving a previous linear system. Applications of the method to multiple linear systems arising from image restorations and recursive least squares computations are considered. In particular, we show that the the theoretical error bound of the method can be applied to these applications. Finally, numerical results are reported to illustrate the effectiveness of the Galerkin projection method.

Key words. multiple linear systems, Krylov space, conjugate gradient method, Galerkin projection

AMS(MOS) subject classification. 65F10, 65Y20

*Department of Mathematics, University of Calif. at Los Angeles, Los Angeles, CA 90024. E-mail: chan@math.ucla.edu. Partially supported by the Office of Naval Research grant N00014-92-J-1890, N00014-96-1-0277, the Army Research Office under contract DAAL-03-91-C-0047 (Univ. Tenn. subcontract ORA4466.04 Amendment 1 and 2), the National Science Foundation grant DMS-9626755.

†Computer Sciences Laboratory, Research School of Information Sciences and Engineering, The Australian National University, Canberra ACT 0200, Australia. E-mail: mng@cslab.anu.edu.au. Research supported by the Cooperative Research Centre for Advanced Computational Systems.

1 Introduction

We want to solve, iteratively using Krylov subspace methods, the following linear systems:

$$A^{(i)}x^{(i)} = b^{(i)}, \quad 1 \leq i \leq s, \tag{1.1}$$

where $A^{(i)}$ are real symmetric positive definite matrices of order n, and in general $A^{(i)} \neq A^{(j)}$ and $b^{(i)} \neq b^{(j)}$ for $i \neq j$. Unlike for direct methods, if the coefficient matrices and the right-hand sides are arbitrary, there is nearly no hope to solve them more efficiently than as s completely un-related systems. Fortunately, in many practical applications, the coefficient matrices and the right-hand sides are not arbitrary, and often there is information sharable among the coefficient matrices and the right-hand sides. Such a situation occurs, for instance, in recursive least squares computations [20], wave scattering problem [14, 4, 9], numerical methods for integral equations [14] and image restorations [13]. In this paper, our aim is to propose a methodology to solve these "*related*" multiple linear systems efficiently.

In [24], Smith et al. proposed and considered using a *seed* method for solving linear systems of the same coefficient matrix but different right-hand sides, i.e.,

$$AX = [b^{(1)}\ b^{(2)}\ \cdots\ b^{(s)}].$$

In the seed method, we select one seed system and solve it by the conjugate gradient method. Then we perform a Galerkin projection of the residuals onto the Krylov subspace generated by the seed system to obtain approximate solutions for the unsolved ones. The approximate solutions are then refined by the conjugate gradient method again. In [24], a very effective implementation of the Galerkin projection method was developed which uses direction vectors generated in the conjugate gradient process to perform the projection. In [6], Chan and Wan observed that the seed method has several nice properties. For instance, the conjugate gradient method when applied to the successive seed system converges faster than the usual CG process. Another observation is that if the right-hand sides are closely related, the method automatically exploits this fact and usually only takes a few restarts to solve all the systems. In [6], a theory was developed to explain these phenomena. We remark that the seed method can be viewed as a special implementation of the Galerkin projection method which had been considered and analyzed earlier for solving linear systems with multiple right-hand sides, see for instance, Parlett [19], Saad [21], van der Vorst [26], Padrakakis et al. [18], Simoncini and Gallopoulos [22, 23]. A very different approach based on the Lanczos method with multiple starting vectors have been recently proposed by Freund and Malhotra [9].

In this paper, we extend the seed method to solve the multiple linear systems (1.1), with different coefficient matrices ($A^{(j)} \neq A^{(k)}$) and different right-hand sides ($b^{(j)} \neq b^{(k)}$). We analyze the seed method and extend the theoretical results given in [6]. We will see that the theoretical error bounds for the approximation obtained from a projection process depends on the projection of the eigenvector components of the error onto a Krylov subspace generated from the previous seed system and how different the system is from the previous one.

Unlike in [6], in the general case here where the coefficient matrices $A^{(i)}$ can be different, it is not possible to derive very precise error bounds since the $A^{(i)}$'s have different eigenvectors in

general. Fortunately, in many applications, even though the $A^{(i)}$'s are indeed different, they may be related to each other in a structured way which allows a more precise error analysis. Such is the case in the two applications that we study in this paper, namely, image restorations and recursive least squares (RLS) computations. More precisely, for the image restoration application, the eigenvectors of the coefficient matrices are the same, while for the RLS computations, the coefficient matrices differ by rank-1 or rank-2 matrices. Numerical examples on these applications are given to illustrate the effectiveness of the projection method. We will see from the numerical results that the eigenvector components of the right-hand sides are effectively reduced after the projection process and the number of iterations required for convergence decreases when we employ the projected solution as initial guess. Moreover, other examples involving more general coefficient matrices (for instance, that do not have the same eigenvectors or differ by a low rank matrix), are also given to test the performance of the projection method. We observe similar behaviour in the numerical results as in image restoration and RLS computations. These numerical results demonstrate that the projection method is effective.

The paper is organized as follows. In §2, we first describe and analyze the seed projection algorithm for general multiple linear systems. In §3, we study multiple linear systems arising from image restoration and RLS applications. Numerical examples are given in §4 and concluding remarks are given in §5.

2 Derivation of the Algorithm

Conjugate gradient methods can be seen as iterative solution methods to solve a linear system of equations by minimizing an associated quadratic functional. For simplicity, we let

$$f_i(x) = \frac{1}{2}x^T A^{(i)} x - (b^{(i)})^T x$$

be the associated quadratic functional of the linear system $A^{(i)}x^{(i)} = b^{(i)}$. The minimizer of f_j is the solution of the linear system $A^{(i)}x^{(i)} = b^{(i)}$. The idea of the projection method is that for each restart, a seed system $A^{(k)}x^{(k)} = b^{(k)}$ is selected from the unsolved ones which are then solved by the conjugate gradient method. An approximate solution $\hat{x}^{(j)}$ of the non-seed system $A^{(j)}x^{(j)} = b^{(j)}$ can be obtained by using search direction p_i^k generated from the ith iteration of the seed system. More precisely, given the ith iterate x_i^j of the non-seed system and the direction vector p_i^k, the approximate solution $\hat{x}^{(j)}$ is found by solving the following minimization problem:

$$\min_{\eta} f_j(x_i^j + \eta p_i^k). \tag{2.2}$$

It is easy to check that the minimizer of (2.2) is attained at $\hat{x}^{(j)} = x_i^j + \eta p_i^k$, where

$$\eta = \frac{(p_i^k)^T r_i^j}{(p_i^k)^T A^{(j)} p_i^k} \quad \text{and} \quad r_i^j = b^{(j)} - A^{(j)} x_i^j. \tag{2.3}$$

After the seed system $A^{(k)}x^{(k)} = b^{(k)}$ is solved to the desired accuracy, a new seed system is selected and the whole procedure is repeated. In the following discussion, we call this method

Projection Method I. We note from (2.3) that the matrix-vector multiplication $A^{(j)}p_j^k$ is required for each projection of the non-seed iteration. In general, the cost of the method will be expensive in the general case where the matrices $A^{(j)}$ and $A^{(k)}$ are different. However, in §3, we will consider two specific applications where the matrices $A^{(k)}$ and $A^{(j)}$ are structurally related. Therefore, the matrix-vector products $A^{(j)}p_j^k$ can be computed cheaply by using the matrix-vector product $A^{(k)}p_j^k$ generated from the seed iteration.

In order to reduce the extra cost in Projection Method I in the general case, we propose using the modified quadratic function $\tilde{f}_j$:

$$\tilde{f}_j(x) \equiv \frac{1}{2}x^T A^{(k)}x - (b^{(j)})^T x,$$

to compute the approximate solution of the non-seed system. Note that we have used $A^{(k)}$ instead of $A^{(j)}$ in the above definition. In this case, we determine the next iterate of the non-seed system by solving the following minimization problem:

$$\min_{\alpha} \tilde{f}_j(x_i^j + \alpha p_i^k).$$

The approximate solution $\hat{x}^{(j)}$ of the non-seed system $A^{(j)}x^{(j)} = b^{(j)}$ is given by

$$\hat{x}^{(j)} = x_i^j + \alpha p_i^k, \tag{2.4}$$

where

$$\eta = \frac{(p_i^k)^T \tilde{r}_i^j}{(p_i^k)^T A^{(k)} p_i^k} \quad \text{and} \quad \tilde{r}_i^j = b^{(j)} - A^{(k)}x_i^j. \tag{2.5}$$

Now the projection process does not require the matrix-vector product involving the coefficient matrix $A^{(j)}$ of the non-seed system. Therefore, the method does not increase the dominant cost (matrix-vector multiplies) of each conjugate gradient iteration. In fact, the extra cost is just one inner product, two vector additions, two scalar-vector multiplications and one division. We call this method *Projection Method II.* Of course, unless $A^{(j)}$ is close to $A^{(k)}$ in some sense, we do not expect this method to work well because $\tilde{f}_j$ is then far from the current f_j.

To summarize the above methods, Table 1 lists the algorithms of Projection Methods I and II. We remark that Krylov subspace methods (for instance conjugate gradient), especially when combined with preconditioning, are known to be powerful methods for the solution of linear systems [10]. We can incorporate the preconditioning strategy into the projection method to speed up its convergence rate. The idea of our approach is to precondition the seed system $A^{(k)}x^{(k)} = b^{(k)}$ by some suitable preconditioner $C^{(k)}$ for each restart. Meanwhile, an approximate solution of the non-seed system $A^{(j)}x^{(j)} = b^{(j)}$ is also obtained from the space of direction vectors generated by the conjugate gradient iterations of preconditioned seed system. We can formulate the preconditioned projection method directly produces vectors that approximate the desired solutions of the non-seed systems. Table 2 lists the preconditioned versions of Projection Methods I and II.

for k=1, ..., s until all the systems are solved
Select the kth system as seed
for i=0, 1, 2, ... m_{k+1} % CG iteration
for j=k, $k+1$, ..., s % unsolved systems
if j=k then perform usual CG steps

$$\delta_i^{k,k} = (r_i^{k,k})^T r_i^{k,k} / (r_{i-1}^{k,k})^T r_{i-1}^{k,k}$$
$$p_i^{k,k} = r_i^{k,k} + \delta_i^{k,k} p_{i-1}^{k,k}$$
$$\sigma_i^{k,k} = (r_i^{k,k})^T r_i^{k,k} / (p_i^{k,k})^T A^{(k)} p_i^{k,k}$$
$$x_{i+1}^{k,k} = x_i^{k,k} + \sigma_i^{k,k} p_i^{k,k}$$
$$r_{i+1}^{k,k} = r_i^{k,k} - \sigma_i^{k,k} A^{(k)} p_i^{k,k}$$

else perform Galerkin projection

$$\eta_i^{k,j} = (p_i^{k,k})^T r_i^{k,j} / (p_i^{k,k})^T A^{(j)} p_i^{k,k}$$
$$x_{i+1}^{k,j} = x_i^{k,j} + \eta_i^{k,j} p_i^{k,k}$$
$$r_{i+1}^{k,j} = r_i^{k,j} - \eta_i^{k,j} A^{(j)} p_i^{k,k}$$

end if
end for
end for
end for

for k=1, ..., s until all the systems are solved
Select the kth system as seed
for i=0, 1, 2, ... m_{k+1} % CG iteration
for j=k, $k+1$, ..., s % unsolved systems
if j=k then perform usual CG steps

$$\delta_i^{k,k} = (r_i^{k,k})^T r_i^{k,k} / (r_{i-1}^{k,k})^T r_{i-1}^{k,k}$$
$$p_i^{k,k} = r_i^{k,k} + \delta_i^{k,k} p_{i-1}^{k,k}$$
$$\sigma_i^{k,k} = (r_i^{k,k})^T r_i^{k,k} / (p_i^{k,k})^T A^{(k)} p_i^{k,k}$$
$$x_{i+1}^{k,k} = x_i^{k,k} + \sigma_i^{k,k} p_i^{k,k}$$
$$r_{i+1}^{k,k} = r_i^{k,k} - \sigma_i^{k,k} A^{(k)} p_i^{k,k}$$

else perform Galerkin projection

$$\eta_i^{k,j} = (p_i^{k,k})^T r_i^{k,j} / (p_i^{k,k})^T A^{(k)} p_i^{k,k}$$
$$x_{i+1}^{k,j} = x_i^{k,j} + \eta_i^{k,j} p_i^{k,k}$$
$$r_{i+1}^{k,j} = r_i^{k,j} - \eta_i^{k,j} A^{(k)} p_i^{k,k}$$

end if
end for
end for
end for

Table 1: Projection Methods I (left) and II (right). The kth system is the seed for the $(k-1)$th restart. The first and the second superscripts is used to denote the kth restart and the jth system. The subscripts is used to denote the ith step of the CG method.

for k=1, ..., s until all the systems are solved
Select the kth system as seed
for i=0, 1, 2, ... m_{k+1} % CG iteration
for j=k, $k+1$, ..., s % unsolved systems
if j=k then perform usual CG steps

$$\delta_i^{k,k} = (r_i^{k,k})^T z_i^{k,k} / (r_{i-1}^{k,k})^T z_{i-1}^{k,k}$$
$$p_i^{k,k} = z_i^{k,k} + \delta_i^{k,k} p_{i-1}^{k,k}$$
$$\sigma_i^{k,k} = (r_i^{k,k})^T z_i^{k,k} / (p_i^{k,k})^T A^{(k)} p_i^{k,k}$$
$$x_{i+1}^{k,k} = x_i^{k,k} + \sigma_i^{k,k} p_i^{k,k}$$
$$r_{i+1}^{k,k} = r_i^{k,k} - \sigma_i^{k,k} A^{(k)} p_i^{k,k}$$
$$z_{i+1}^{k,k} = (C^{(k)})^{-1} r_{i+1}^{k,k}$$ % preconditioning

else perform Galerkin projection

$$\eta_i^{k,j} = (z_i^{k,k})^T r_i^{k,j} / (p_i^{k,k})^T A^{(j)} p_i^{k,k}$$
$$x_{i+1}^{k,j} = x_i^{k,j} + \eta_i^{k,j} p_i^{k,k}$$
$$r_{i+1}^{k,j} = r_i^{k,j} - \eta_i^{k,j} A^{(j)} p_i^{k,k}$$

end if
end for
end for
end for

for k=1, ..., s until all the systems are solved
Select the kth system as seed
for i=0, 1, 2, ... m_{k+1} % CG iteration
for j=k, $k+1$, ..., s % unsolved systems
if j=k then perform usual CG steps

$$\delta_i^{k,k} = (r_i^{k,k})^T z_i^{k,k} / (r_{i-1}^{k,k})^T z_{i-1}^{k,k}$$
$$p_i^{k,k} = z_i^{k,k} + \delta_i^{k,k} p_{i-1}^{k,k}$$
$$\sigma_i^{k,k} = (r_i^{k,k})^T z_i^{k,k} / (p_i^{k,k})^T A^{(k)} p_i^{k,k}$$
$$x_{i+1}^{k,k} = x_i^{k,k} + \sigma_i^{k,k} p_i^{k,k}$$
$$r_{i+1}^{k,k} = r_i^{k,k} - \sigma_i^{k,k} A^{(k)} p_i^{k,k}$$
$$z_{i+1}^{k,k} = (C^{(k)})^{-1} r_{i+1}^{k,k}$$ % preconditioning

else perform Galerkin projection

$$\eta_i^{k,j} = (z_i^{k,k})^T r_i^{k,j} / (p_i^{k,k})^T A^{(k)} p_i^{k,k}$$
$$x_{i+1}^{k,j} = x_i^{k,j} + \eta_i^{k,j} p_i^{k,k}$$
$$r_{i+1}^{k,j} = r_i^{k,j} - \eta_i^{k,j} A^{(k)} p_i^{k,k}$$

end if
end for
end for
end for

Table 2: Preconditioned Projection Methods I (left) and II (right)

We emphasize that in [6, 19, 21, 23, 24], the authors only considered using the projection method for solving linear systems with the same coefficient matrix but different right-hand sides. In this paper, we use Projection Methods I and II to solve linear systems with different coefficient matrices and right-hand sides . An important question regarding the approximation obtained from the above process is its accuracy. For Projection Method I, it is not easy to derive error bounds since the direction vectors generated for the seed system $A^{(k)}x^{(k)} = b^{(k)}$ are only $A^{(k)}$-orthogonal but are not $A^{(j)}$-orthogonal in general. In the following discussion, we only analyze Projection Method II. However, the numerical results in §4 shows that Projection method I is very efficient for some applications and is generally faster convergent than Projection Method II.

2.1 Analysis of Projection Method II

For Projection Method II, we have the following Lemma in exact arithmetic.

Lemma 1 *Assume that a seed system $A^{(k)}x^{(k)} = b^{(k)}$ has been selected. Using Projection Method II, the approximate solution of the non-seed system $A^{(j)}x^{(j)} = b^{(j)}$ at the ith iteration is given by*

$$x_i^{k,j} = x_0^{k,j} + V_i^k (T_i^k)^{-1} (V_i^k)^T r_0^{k,j}, \tag{2.6}$$

where $x_\ell^{k,j}$ is ℓth iterate of the non-seed system, V_i^k is the Lanczos vectors generated by i steps of the Lanczos algorithm if the seed system $A^{(k)}x^{(k)} = b^{(k)}$ is solved by the Lanczos algorithm, $T_i^k = (V_i^k)^T A^{(k)} V_i^k$ and $r_0^{k,j} = b^{(j)} - A^{(k)} x_0^{k,j}$.

Proof: Let the columns of $V_i^k = [v_1^k, v_2^k, \ldots, v_i^k]$ be the orthonormal vectors of the i-dimensional Krylov subspace generated by i steps of the Lanczos method. Then we have the following well-known three-term recurrence

$$A^{(k)} V_i^k = V_i^k T_i^k + \beta_{i+1}^k v_{i+1}^k e_i^T,$$

where e_i is the ith column of the identity matrix and β_{i+1}^k is a scalar. From (2.4) (or see [24]), the approximate solution $x_i^{k,j}$ of the non-seed system is computed in the subspace generated by the direction vectors $\{p_\ell^{k,k}\}$ generated from the seed iteration. However, this subspace generated by the direction vectors is exactly the subspace spanned by the columns of V_i^k, see [10]. Therefore, we have

$$x_i^{k,j} = x_0^{k,j} + V_i^k z, \quad \text{for some } z.$$

Moreover, it is easy to check from (2.4) and (2.5) that

$$(p_\ell^{k,k})^T (b^{(j)} - A^{(k)} x_i^{k,j}) = 0, \quad \ell = 1, 2, \ldots, i.$$

It follows that the solution $x_i^{k,j}$ can be obtained by the Galerkin projection onto the Krylov subspace $\mathcal{K}^{(k)}$ generated by the seed system. Equivalently, $x_i^{k,j}$ can be determined by solving the following problem:

$$(V_i^k)^T (b^{(j)} - A^{(k)} z), \quad \text{where } z = x_0^{k,j} + y \text{ and } y \in V_i^{(k)}.$$

Noting that the solution is $z = x_0^{k,j} + V_i^k (T_i^k)^{-1} (V_i^k)^T (b^{(j)} - A^{(k)} x_0^{k,j})$. the result follows. $\square$

To analyze the error bound of Projection Method II, without loss of generality, consider only two symmetric positive definite n-by-n linear systems:

$$A^{(1)} x^{(1)} = b^{(1)} \quad \text{and} \quad A^{(2)} x^{(2)} = b^{(2)}.$$

The eigenvalues and normalized eigenvectors of $A^{(i)}$ are denoted by $\lambda_k^{(i)}$ and $q_k^{(i)}$ respectively and $0 < \lambda_1^{(i)} \leq \lambda_2^{(i)} \leq \cdots \leq \lambda_n^{(i)}$ for $i = 1, 2$. The theorem below gives error bounds for Projection Method II for solving multiple linear systems with different coefficient matrices and right-hand sides.

Theorem 1 *Suppose the first linear system $A^{(1)} x^{(1)} = b^{(1)}$ is solved to the desired accuracy in m CG steps. Let $x_0^{1,2}$ be the solution of the second system $A^{(2)} x^{(2)} = b^{(2)}$ obtained from the projection onto $\mathcal{K}_m$ generated by the first system, with zero vector as the initial guess of the second system ($x_0^{0,2} = 0$). Let the eigen-decomposition of $x^{(2)} - x_0^{1,2}$ be expressed as*

$$x^{(2)} - x_0^{1,2} = \sum_{k=1}^{n} c_k q_k^{(2)}.$$

Then the eigenvector components c_k can be bounded by:

$$|c_k| \leq E_k + F, \quad 1 \leq k \leq n,$$

where

$$E_k = \|P_m^{\perp} x^{(2)}\|_2 |\sin \angle (q_k^{(2)}, \mathcal{K}_m)| \quad \text{and} \quad F = \|(A^{(1)})^{-1}\|_2 \|(A^{(2)} - A^{(1)}) x^{(2)}\|_2.$$

Here V_m is the orthonormal vectors of $\mathcal{K}_m$, $P_m^{\perp} = I - V_m T_m^{-1} V_m^T A^{(1)}$ is the $A^{(1)}$-orthogonal projection onto $\mathcal{K}_m$ and $T_m = V_m^T A^{(1)} V_m$ is the matrix representation of the projection of $A^{(1)}$ onto $\mathcal{K}_m$.

Proof: By (2.6), we get $x_0^{1,2} \equiv x_m^{0,2} = V_m T_m^{-1} V_m^T b^{(2)}$. Then

$$x^{(2)} - x_0^{1,2} = \left(I - V_m T_m^{-1} V_m^T A^{(2)}\right) x^{(2)} = P_m^{\perp} x^{(2)} - V_m T_m^{-1} V_m^T \left(A^{(2)} - A^{(1)}\right) x^{(2)}.$$

Since V_m is the orthogonal vectors of $\mathcal{K}_m$ and

$$\lambda_{\min}(T_m) = \min_{\|x\|_2 = 1} x^T V_m^T A^{(1)} V_m x = \min_{y = Vz,\ \|y\|_2 = 1} y^T A^{(1)} y \geq \lambda_{min}(A^{(1)}),$$

we have $\|V_m\|_2 \leq 1$ and $\|T_m^{-1}\|_2 \leq \|(A^{(1)})^{-1}\|_2$. It follows that

$$\begin{aligned} |c_k| &= \left| \left[P_m^{\perp} x^{(2)} - V_m T_m^{-1} V_m^T (A^{(2)} - A^{(1)}) x^{(2)} \right]^T \cdot q_k^{(2)} \right| \\ &\leq \|P_m^{\perp} x^{(2)}\|_2 \|q_k^{(2)}\|_2 |\cos \angle (q_k^{(2)}, P_m^{\perp} x^{(2)})| + \|(A^{(1)})^{-1}\|_2 \|(A^{(2)} - A^{(1)}) x^{(2)}\|_2 \\ &\leq \|P_m^{\perp} x^{(2)}\|_2 |\sin \angle (q_k^{(2)}, \mathcal{K}_m)| + \|(A^{(1)})^{-1}\|_2 \|(A^{(2)} - A^{(1)}) x^{(2)}\|_2. \quad \square \end{aligned}$$

Theorem 1 basically states that the size of the eigenvector component c_k is bounded by E_k and F. If the Krylov subspace $\mathcal{K}_m$ generated by the seed system contains the eigenvectors $q_k^{(2)}$ well, then the projection process will kill off the eigenvector components of the initial error of the non-seed system, i.e., E_k is very small. On the other hand, F depends essentially on how different the system $A^{(2)}x^{(2)} = b^{(2)}$ is from the previous one $A^{(1)}x^{(1)} = b^{(1)}$. In particular, when $\|A^{(1)} - A^{(2)}\|_2$ is small, then F is also small.

We remark that when $A^{(1)} = A^{(2)}$ and $b^{(1)} \neq b^{(2)}$, the term F becomes zero, and as $q_k^{(1)} = q_k^{(2)}$, the term E_k becomes $\|P_m^{\perp}x^{(2)}\|_2|\sin\angle(q_k^{(1)}, \mathcal{K}_m)\|$. It is well-known that the Krylov subspace $\mathcal{K}_m$ generated by the seed system contains the eigenvectors $q_k^{(1)}$ well. In particular, Chan and Wan [6] have the following result about the estimate of the bound $\sin\angle(q_k^{(1)}, \mathcal{K}_m)$.

Lemma 2 *Let* $\theta_k = \angle(A^{(1)}b^{(1)}, q_k^{(1)})$, $\tau_k = \frac{(\lambda_k^{(1)} - \lambda_{k+1}^{(1)})}{(\lambda_{k+1}^{(1)} - \lambda_n^{(1)})}$, *and* $\omega_k = \prod_{\nu=1}^{k-1}\left(\frac{\lambda_\nu^{(1)} - \lambda_n^{(1)}}{\lambda_\nu^{(1)} - \lambda_k^{(1)}}\right)/T_{m-k}(1 + 2\tau_k)$ *where* $T_j(x)$ *is the Chebyshev polynomial of degree* j. *Then*

$$\sin\angle(q_k^{(1)}, \mathcal{K}_m) \leq \omega_k \tan\theta_k. \tag{2.7}$$

If we assume that the eigenvalues of $A^{(1)}$ are distinct, then $T_{m-k}(1+2\tau_k)$ grows exponentially as m increases and therefore the magnitude $\sin\angle(q_k^{(1)}, \mathcal{K}_m)$ is very small for sufficiently large m. It implies that the magnitude E_k is very small when m is sufficiently large. Unfortunately, we cannot have this result in the general case since $q_k^{(1)} \neq q_k^{(2)}$, except in some special cases that will be discussed in the next section.

3 Applications of Galerkin Projection Methods

In this section, we consider using the Galerkin projection method for solving multiple linear systems arising in two particular applications from image restorations and recursive least squares computations. In these applications, the coefficient matrices differ by a parameterized identity matrix or a low rank matrix. We note from Theorem 1 that the theoretical error bound of the projection method depends on E_k and F. In general, it is not easy to refine the error bound E_k and F. However, in these cases, the error bound E_k and F can be further investigated.

3.1 Tikhonov Regularization in Image Restorations

Image restoration refers to the removal or reduction of degradations (or blur) in an image using a priori knowledge about the degradation phenomena; see for instance [13]. When the quality of the images is degraded by blurring and noise, important information remains hidden and cannot be directly interpreted without numerical processing. In matrix-vector notation, the

linear algebraic form of the image restoration problem for an n-by-n pixel image is given as follows:

$$b = Ax + \eta, \tag{3.8}$$

where b, x, and η are n^2-vectors and A is an n^2-by-n^2 matrix. Given the observed image b, the matrix A which represents the degradation, and possibly, the statistics of the noise vector η, the problem is to compute an approximation to the original signal x.

Because of the ill-conditioning of A, naively solving $Ax = b$ will lead to extreme instability with respect to perturbations in b, see [13]. The method of *regularization* can be used to achieve stability for these problems [1, 3]. In the classical *Tikhonov regularization* [12], stability is attained by introducing a stabilizing operator D (called a regularization operator), which restricts the set of admissible solutions. Since this causes the regularized solution to be biased, a scalar μ, called a regularization parameter, is introduced to control the degree of bias. More specifically, the regularized solution is computed as the solution to

$$\min_{x(\mu)} \left\| \begin{bmatrix} b \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} A \\ \mu D \end{bmatrix} x(\mu) \right\|_2^2 \quad \text{or} \quad \min_{x(\mu)} \mu^2 \|Dx(\mu)\|_2^2 + \|b - Ax(\mu)\|_2^2. \tag{3.9}$$

The term $\|Dx(\mu)\|_2^2$ is added in order to regularize the solution. Choosing D as a kth order difference operator matrix forces the solution to have a small kth order derivative. When the rectangular matrix has full column rank, one can find the solution by solving the normal equations

$$(\mu^2 D^T D + A^T A)x = A^T b. \tag{3.10}$$

The regularization parameter μ controls the degree of smoothness (i.e., degree of bias) of the solution, and is usually small. Choosing μ is not a trivial problem. In some cases *a priori* information about the signal and the degree of perturbations in b can be used to choose μ [1], or generalized cross-validation techniques may also be used, e.g., [3]. If no *a priori* information is known, then it may be necessary to solve (3.10) for several values of μ. For example, in the L-curve method discussed in [7], choosing the parameter μ requires solving the linear systems with different values of μ. This gives rise to multiple linear systems which can be solved by our proposed projection methods.

In some applications [13, 5], the regularization operator D can be chosen to be the identity matrix. Consider for simplicity two linear systems:

$$(\mu_i I + A^T A)x_i \equiv A^{(i)} x_i = A^T b, \quad i = 1, 2.$$

In this case, we can employ Projection Method I to solve these multiple linear systems as the matrix-vector product $(\mu_2 I + A^T A)p$ in the non-seed iteration can be computed cheaply by adding $(\mu_1 I + A^T A)p$ generated from the seed iteration and $(\mu_2 - \mu_1)p$ together. Moreover, we can further refine the error bound of Projection Method II in Theorem 1. Now assume that m steps of the conjugate gradient algorithm have been performed to solve the first system. We note in this case that the eigenvectors of the first and the second linear systems are the same,

i.e., $q_k^{(1)} = q_k^{(2)}$. Therefore, we can bound $\sin \angle(q_k^{(1)}, \mathcal{K}_m)$ using Lemma 2. We shall prove that if the Krylov subspace of the first linear system contains the extreme eigenvectors well, the bound for the convergence rate is effectively the classical conjugate gradient bound but with a reduced condition number.

Theorem 2 *Let $x_0^{1,2}$ be the solution of the second system obtained from the projection onto $\mathcal{K}_m$ generated by the first system. The bound for the $A^{(2)}$-norm of the error vector after i steps of the conjugate gradient process is given by*

$$\|x^{(2)} - x_i^{1,2}\|_{A^{(2)}}^2 \leq 4\|x^{(2)} - \bar{x}_i^{1,2}\|_{A^{(2)}}^2 \left(\frac{\sqrt{\kappa_r} - 1}{\sqrt{\kappa_r} + 1}\right)^{2i} + \delta,$$

where $\bar{x}_i^{1,2}$ is ith iterate of the CG process for $A^{(2)}x^{(2)} = b^{(2)}$ with the projection of $x^{(2)} - x_0^{1,2}$ onto span $\{q_k^{(1)} : k = 1, 2, \cdots, \ell\}^{\perp}$ as initial guess, $\kappa_r = \lambda_n^{(2)}/\lambda_{\ell+1}^{(2)}$ is the reduced condition number of $A^{(2)}$ and

$$\delta = \sum_{k=1}^{\ell} \lambda_k^{(2)} \left[\left|1 - \frac{\mu_2}{\mu_1}\right| \|x^{(2)}\|_2 + \|P_m^{\perp} x^{(2)}\|_2^2 \omega_k \tan\theta_k\right]^2. \tag{3.11}$$

Proof: We first expand the eigen-components of $x^{(2)} - x_0^{1,2}$,

$$x^{(2)} - x_0^{1,2} = \sum_{k=1}^{n} c_k q_k^{(1)} = \sum_{k=1}^{n} c_k q_k^{(2)}.$$

It is well-known [10] that there exists a polynomial $\bar{p}_i(t)$ of degree at most i and constant term 1 such that

$$x^{(2)} - \bar{x}_i^{1,2} = \bar{p}_i(A^{(2)})(x^{(2)} - \bar{x}_0^{1,2}) = \sum_{k=\ell+1}^{n} c_k q_k^{(2)}.$$

By using properties of the conjugate gradient iteration given in [10], we have

$$\begin{aligned}
\|x^{(2)} - x_i^{1,2}\|_{A^{(2)}}^2 &= \min_{p_i} \|p_i(A^{(2)})(x^{(2)} - x_0^{1,2})\|_{A^{(2)}}^2 \\
&\leq \|\bar{p}_i(A^{(2)})(x^{(2)} - x_0^{1,2})\|_{A^{(2)}}^2 \\
&= \|\sum_{k=1}^{n} \bar{p}_i(\lambda_k^{(2)}) c_k q_k^{(1)}\|_{A^{(2)}}^2 \\
&= \sum_{k=1}^{n} \bar{p}_i^2(\lambda_k^{(2)}) c_k^2 \lambda_k^{(2)} \\
&= \sum_{k=\ell+1}^{n} \bar{p}_i^2(\lambda_k^{(2)}) c_k^2 \lambda_k^{(2)} + \sum_{k=1}^{\ell} \bar{p}_i^2(\lambda_k^{(2)}) c_k^2 \lambda_k^{(2)} \\
&= \|x^{(2)} - \bar{x}_i^{1,2}\|_{A^{(2)}}^2 + \sum_{k=1}^{\ell} \bar{p}_i^2(\lambda_k^{(2)}) c_k^2 \lambda_k^{(2)}.
\end{aligned} \tag{3.12}$$

Now the term $\|x^{(2)} - \bar{x}_i^{1,2}\|_{A^{(2)}}^2$ can be bounded by the classical CG error estimate,

$$\|x^{(2)} - \bar{x}_i^{1,2}\|_{A^{(2)}}^2 \leq 4\|x^{(2)} - \bar{x}_0^{1,2}\|_{A^{(2)}}^2 \left(\frac{\sqrt{\kappa_r} - 1}{\sqrt{\kappa_r} + 1}\right)^{2i}.$$

Noting that $\|(A^{(1)})^{-1}\|_2 \leq 1/\mu_1$, $A^{(2)} - A^{(1)} = (\mu_2 - \mu_1)I$, $\max_{1 \leq k \leq \ell} \bar{p}_i^2(\lambda_k^{(2)}) \leq 1$, and using Theorem 1 and Lemma 2, the result follows by substitution (2.7) into (3.12). □

We see that the perturbation term δ contains two parts. One depends on the ratio μ_2/μ_1 of the regularization parameters between two linear systems and the other depends on how well the Krylov subspace of the seed system contains the extreme eigenvectors. We remark that the regularization parameter μ in practice is always greater than 0 in image restoration applications because of the ill-conditioning of A. In particular, $\mu_1 \neq 0$. If the ratio μ_2/μ_1 is near to 1, then the magnitude of this term will be near to zero. On the other hand, according to Lemma 2, the Galerkin projection will kill off the extreme eigenvector components and therefore the quantity $\omega_k \tan\theta_k$ in (3.11) will be also small for k close to 1. Hence the perturbation term δ becomes very small and the CG method, when applied to solve the non-seed system, converges faster than the usual CG process.

3.2 Recursive Least Squares Computations in Signal Processing

Recursive least squares (RLS) computations are used extensively in many signal processing and control applications; see Alexander [2]. The standard linear least squares problem can be posed as follows: Given a real p-by-n matrix X with full column rank n (so that $X^T X$ is symmetric positive definite) and a p-vector b, find the n-vector w that solves

$$\min_w \|b - Xw\|_2. \tag{3.13}$$

In RLS computations, it is required to recalculate w when observations (i.e., equations) are successively added to, or deleted from, the problem (3.13). For instance, in many applications information arrives continuously and must be incorporated into the solution w. This is called *updating*. It is sometimes important to delete old observations and have their effect removed from w. This is called *downdating* and is associated with a sliding data window. Alternatively, an exponential forgetting factor β, with $0 < \beta \leq 1$ (see for instance [2]), may be incorporated into the updating computations to exponentially decay the effect of the old data over time. The use of β is associated with an exponentially-weighted data window.

3.2.1 Rank-1 Updating and Downdating Sliding Window RLS

At the time step t, the data matrix and the desired response vector are given by

$$X(t) = \begin{bmatrix} x(t-p+1)^T \\ \vdots \\ x(t)^T \end{bmatrix} \quad \text{and} \quad d(t) = \begin{bmatrix} d_t \\ \vdots \\ d_{t-p+1} \end{bmatrix} \tag{3.14}$$

respectively, where p is the length of sliding window (one always assumes that $p \geq n$). We solve the following least squares problem: $\min_{w(t)} \|d(t) - X(t)w(t)\|_2$. Now we assume that a row $x(t+1)^T$ is added and a row $x(t-p+1)^T$ is removed at the step $t+1$. The right-hand-side desired response vector $d(t+1)$ is modified in a corresponding fashion. One now seeks to solve the modified least squares problem $\min_{w(t+1)} \|d(t+1) - X(t+1)w(t+1)\|_2$ for the updated least squares estimate vector $w(t+1)$ at the time step $t+1$. We note that its normal equations are given by

$$\begin{aligned} & [X(t)^T X(t) + x(t+1)x(t+1)^T - x(t-p+1)x(t-p+1)^T]w(t+1) \\ = \; & X(t)^T b(t) + d_{t+1}x(t+1) - d_{t-p+1}x(t-p+1). \end{aligned} \tag{3.15}$$

Therefore, the coefficient matrices at the time step t and $t+1$ differ by a rank-2 matrix.

3.2.2 Exponentially-weighted RLS

For the exponentially-weighted case, the data matrix $X(t)$ and desired response vector $d(t)$ at the time step t are defined [2] recursively by

$$X(t) = \begin{bmatrix} \sqrt{\beta}X(t-1) \\ x(t)^T \end{bmatrix} \quad \text{and} \quad d(t) = \begin{bmatrix} \sqrt{\beta}d(t-1) \\ d_t \end{bmatrix},$$

where β is the forgetting factor, and $x^T(t) = (x_t, x_{t-1}, \cdots, x_{t-n+1})$, with $X(1) = x^T(1)$ and $d(1) = d_1$. The RLS algorithms recursively solve for the least squares estimator $w(t)$ at time t, with $t \geq n$. The least squares estimator at the time t and $t+1$ can be found by solving the corresponding least squares problems and their normal equations are given by

$$X(t)^T X(t)w(t) = X(t)^T d(t)$$

and

$$[\beta X(t)^T X(t) + x(t+1)x(t+1)^T]w(t+1) = \beta X(t)d(t) + d_{t+1}x(t+1), \tag{3.16}$$

respectively. We remark that these two coefficient matrices differ by a rank-1 matrix plus a scaling.

3.2.3 Multiple Linear Systems in RLS computations

We consider multiple linear systems in RLS computations, i.e., we solve the following least squares problem successively

$$\begin{cases} \min_{w(t)} \|d(t) - X(t)w(t)\|_2 \\ \min_{w(t+1)} \|d(t+1) - X(t+1)w(t+1)\|_2 \\ \vdots \\ \min_{w(t+s)} \|d(t+s) - X(t+s)w(t+s)\|_2, \end{cases} \tag{3.17}$$

where s is an arbitrary block size of RLS computations. The implementation of recursive least squares estimators have been proposed and used [8]. Their algorithms updates the filter coefficients by minimizing the average least squares error over a set of data samples. For instance, the least squares estimates can be computed by modifying the Cholesky factor of the normal equations with $O(n^2)$ operations per adaptive filter input [20]. For our approach, we employ the Galerkin projection method to solve the multiple linear systems arising from sliding window or exponentially-weighted RLS computations.

For the sliding window RLS computation with rank-1 updating and downdating, by (3.15), the multiple linear systems are given by

$$\begin{aligned}
1st\ system: &\quad X(t)^T X(t) w(t) = X(t)^T b(t) \\
2nd\ system: &\quad \left[X(t)^T X(t) + x(t+1)x(t+1)^T - x(t-p+1)x(t-p+1)^T\right] w(t+1) \\
&= X(t)^T b(t) + d_{t+1}x(t+1) - d_{t-p+1}x(t-p+1) \\
&\quad \vdots \\
(s+1)th\ system: &\quad \left[X(t)^T X(t) + \sum_{j=1}^{s} x(t+j)x(t+j)^T - \sum_{j=1}^{s} x(t-p+j)x(t-p+j)^T\right] w(t+s) \\
&= X(t)^T b(t) + \sum_{j=1}^{s} d_{t+j}x(t+j) - \sum_{j=1}^{s} d_{t-p+j}x(t-p+j). \qquad (3.18)
\end{aligned}$$

For the exponentially-weighted case, by (3.16), the multiple linear systems are given by

$$\begin{aligned}
1st\ system: &\quad X(t)^T X(t) w(t) = X(t)^T b(t) \\
2nd\ system: &\quad \left[\beta X(t)^T X(t) + x(t+1)x(t+1)^T\right] w(t+1) = \beta X(t)^T b(t) + d_{t+1}x(t+1) \\
&\quad \vdots \\
(s+1)th\ system: &\quad \left[\beta^s X(t)^T X(t) + \sum_{j=1}^{s} \beta^{s-j} x(t+j)x(t+j)^T\right] w(t+s) \\
&= \beta^s X(t)^T b(t) + \sum_{j=1}^{s} \beta^{s-j} d_{t+j}x(t+j). \qquad (3.19)
\end{aligned}$$

According to (3.18), the consecutive coefficient matrices only differ by a rank-2 matrix in the sliding data window case. From (3.19), the consecutive coefficient matrices only differ by a rank-1 matrix and the scaled coefficient matrix in the exponentially-weighted case. In these RLS computations, Projection Method I can be used to solve these multiple linear systems as the matrix-vector product in the non-seed iteration can be computed inexpensively. For instance, the matrix-vector product for the new system can be computed by

$$[\beta X(t)^T X(t) + x(t+1)x(t+1)^T]p := \beta p_1 + x(t+1)x(t+1)^T p,$$

where $p_1 := X(t)^T X(t)p$ is generated from the seed iteration. The extra cost is some inner products. We remark that for the other linear systems in (3.18) and (3.19), we need more inner

products because the coefficient matrices $X(t)^T X(t)$ and $X(t+s)^T X(t+s)$ differ by a rank-s or rank-$2s$ matrices.

We analyze below the error bound given by Projection Method II for the case that the coefficient matrices differ by a rank-1 matrix, i.e.,

$$A^{(2)} = A^{(1)} + \rho r r^T,$$

where r has unit 2-norm and each component is greater than zero. For the exponentially-weighted case, we note that

$$A^{(1)} = X(t)^T X(t), \quad r = \frac{x(t+1)}{\sqrt{\beta \|x(t+1)\|_2}}, \quad \text{and} \quad \rho = \|x(t+1)\|_2.$$

By using the eigenvalue-eigenvector decomposition of $A^{(1)}$, we obtain

$$A^{(2)} = Q(\Lambda^{(1)} + \rho z z^T) Q^T,$$

with $Q = [q_1^{(1)} q_2^{(1)} \cdots q_n^{(1)}]$, $\Lambda^{(1)}$ is a diagonal matrix containing eigenvalues $\lambda_i^{(1)}$ of $A^{(1)}$ and $z = Q^T r$. It has been shown in [11] that if $\lambda_k^{(1)} \neq \lambda_k^{(2)}$ for all k, then the eigenvalues $\lambda_k^{(2)}$ of $A^{(2)}$ can be computed by solving the secular equation

$$\phi(\lambda) = 1 + \rho \sum_{i=1}^{n} \frac{[(q_i^{(1)})^T r]^2}{(\lambda_k^{(1)} - \lambda)} = 0.$$

Moreover, the eigenvectors $q_k^{(2)}$ of $A^{(2)}$ can be calculated by the formula:

$$q_k^{(2)} = \frac{Q(\Lambda^{(1)} - \lambda_k^{(2)} I)^{-1} Q^T r}{\|(\Lambda^{(1)} - \lambda_k^{(2)} I)^{-1} Q^T r\|_2}, \quad 1 \leq k \leq n. \tag{3.20}$$

Theorem 3 *Suppose the first linear system $A^{(1)} x^{(1)} = b^{(1)}$ is solved to the desired accuracy in m CG steps. Then the eigenvector components c_k of the second system are bounded by $|c_k| \leq E_k + F$, for $1 \leq k \leq n$, where*

$$E_k = \|P_m^{\perp} x^{(2)}\|_2 \sum_{i=1}^{n} |\gamma_{i,k}| \left| \sin \angle(q_i^{(1)}, \mathcal{K}_m) \right|, \quad \text{and} \quad F = |\rho| \|(A^{(1)})^{-1}\|_2 \|r^T x^{(2)}\|_2,$$

and

$$\gamma_{i,k} = \frac{\frac{(q_i^{(1)})^T r}{(\lambda_i^{(1)} - \lambda_k^{(2)})}}{\sqrt{\sum_{j=1}^{n} \left[\frac{(q_j^{(1)})^T r}{\lambda_j^{(1)} - \lambda_k^{(2)}} \right]^2}},$$

where $\{q_i^{(1)}\}$ is the orthonormal eigenvectors of $A^{(1)}$ and $\mathcal{K}_m$ is the Krylov subspace generated for the first system.

Proof: We just note from Theorem 1 that $|c_k| \leq |(P_m^{\perp}x^{(2)})^T \cdot q_k^{(2)}| + |\rho|\|(A^{(1)})^{-1}\|_2\|r^T x^{(2)}\|_2$. By using (3.20), Theorem 1 and Lemma 2, we can analyze the term $|(P_m^{\perp}x^{(2)})^T \cdot q_k^{(2)}|$ and obtain

$$\left|(P_m^{\perp}x^{(2)})^T \cdot q_k^{(2)}\right| = \sum_{i=1}^{n} |\gamma_{i,k}| \left|\cos(q_i^{(1)}, P_m^{\perp}x^{(2)})\right| \leq \|P_m^{\perp}x^{(2)}\|_2 \sum_{i=1}^{n} |\gamma_{i,k}| \left|\sin \angle(q_i^{(1)}, \mathcal{K}_m)\right|. \quad \square$$

Since $|\gamma_{i,k}|$ and $|\sin \angle(q_i^{(1)}, \mathcal{K}_m)|$ are less than 1, we have

$$E_k \leq \|P_m^{\perp}x^{(2)}\|_2 \left[\sum_{\text{small and large } i} \left|\sin \angle(q_i^{(1)}, \mathcal{K}_m)\right| + \sum_{\text{remaining } i} |\gamma_{i,k}| \right]. \tag{3.21}$$

From Lemma 2, for i close to 1 or n, $|\sin \angle(q_i^{(1)}, \mathcal{K}_m)|$ is sufficiently small when m is large. Moreover, we note that if $\rho > 0$, then

$$\lambda_k^{(1)} \leq \lambda_k^{(2)} \leq \lambda_{k+1}^{(1)}, \quad k = 1, 2, \cdots, n-1, \quad \text{and} \quad \lambda_n^{(1)} \leq \lambda_n^{(2)} \leq \lambda_n^{(1)} + \rho.$$

if $\rho < 0$, then

$$\lambda_1^{(1)} - \rho \leq \lambda_1^{(2)} \leq \lambda_1^{(1)}, \quad \text{and} \quad \lambda_{k-1}^{(1)} \leq \lambda_k^{(2)} \leq \lambda_k^{(1)}, \quad k = 2, 3, \cdots, n,$$

see [10]. Therefore, if the values $(q_i^{(1)})^T r$ are about the same magnitude for each eigenvector $q_i^{(1)}$, then the maximum value of $|\gamma_{i,k}|$ is attained at either $i = k$ or $i = k+1$. We may expect that the second term of the inequality (3.21) is small when k is close to 1 or n. By combining these facts, we can deduce that E_k is also small when k is close to 1 or n. On the other hand, if the scalar ρ is small (i.e., the 2-norm of rank-1 matrix is small), then F is also small. To illustrate the result, we apply Projection Method II to solve $A^{(1)}x^{(1)} = b^{(1)}$ and $(A^{(1)} + \rho rr^T)x^{(2)} = b^{(2)}$, where $A^{(1)}$ = diag$(1, \cdots, 100)$ and r, $b^{(1)}$ and $b^{(2)}$ are random vectors with unit 2-norm. Figures 1 and 2 show that some of the extreme eigenvector components of $b^{(2)}$ are killed off by the projection especially when $|\rho|$ is small. This property suggests that the projection method is useful to solve multiple linear systems arising from recursive lease squares computations. Numerical examples will be given in the next section to illustrate the efficiency of the method.

4 Numerical Results

In this section, we provide experimental results of using Projection Methods I and II to solve multiple linear systems (1.1). All the experiments are performed in MATLAB with machine precision 10^{-16}. The stopping criterion is: $\|r_i^{k,j}\|_2 < tol \times \|b^{(j)}\|_2$, where tol is the tolerance we used. The first and the second examples are Tikhonov regularization in image restoration and the recursive least squares estimation, exactly as discussed in §3. The coefficient matrices $A^{(i)}$'s have the same eigenvectors in the Example 1. In Example 2, the coefficient matrices $A^{(i)}$'s differ

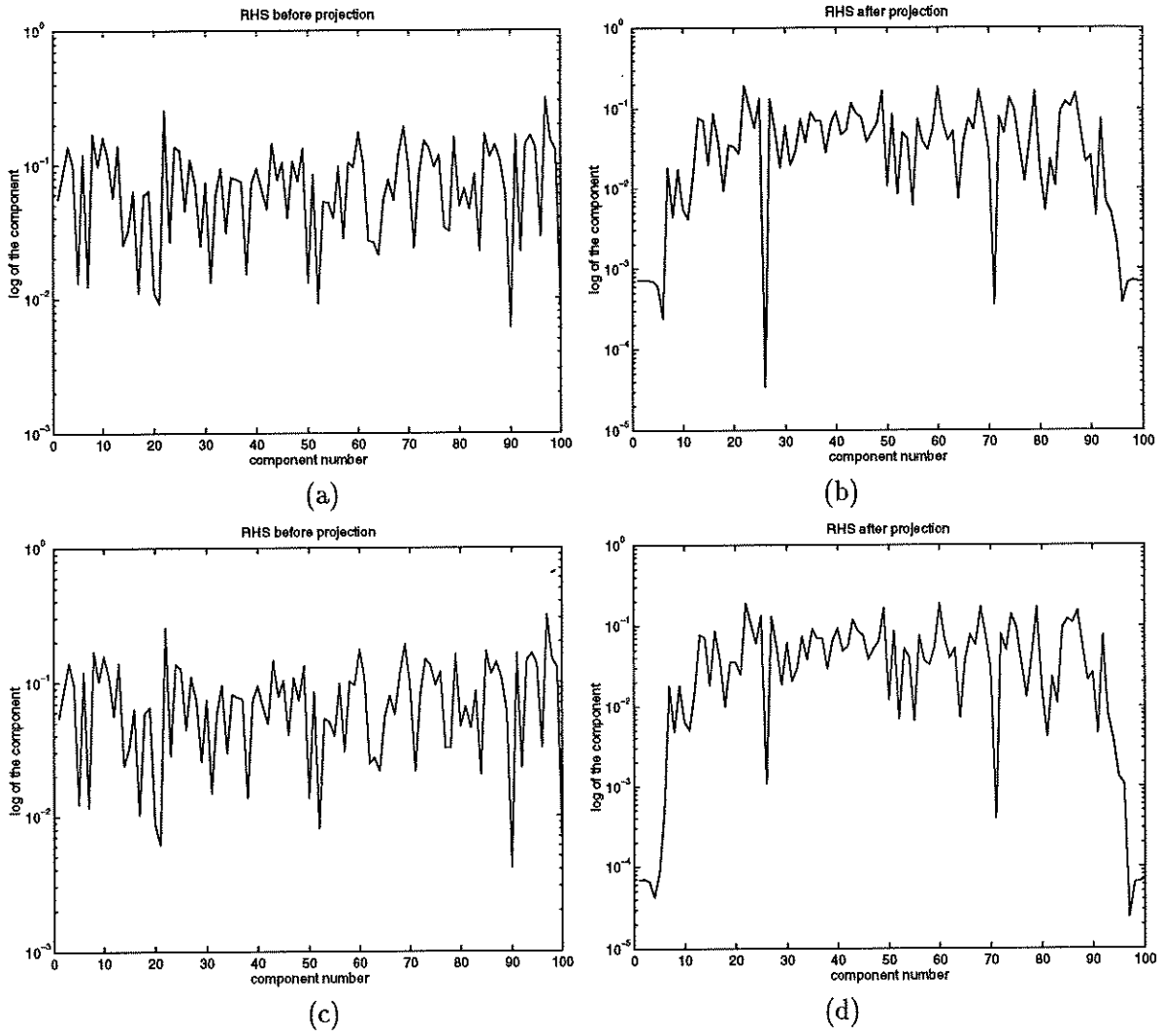

Figure 1: Size distribution of the components of (a) the original right hand side $b^{(2)}$, (b) $b^{(2)}$ after Galerkin projection when $\rho = 1$. Size distribution of the components of (c) the original right hand side $b^{(2)}$, (d) $b^{(2)}$ after Galerkin projection when $\rho = 0.1$.

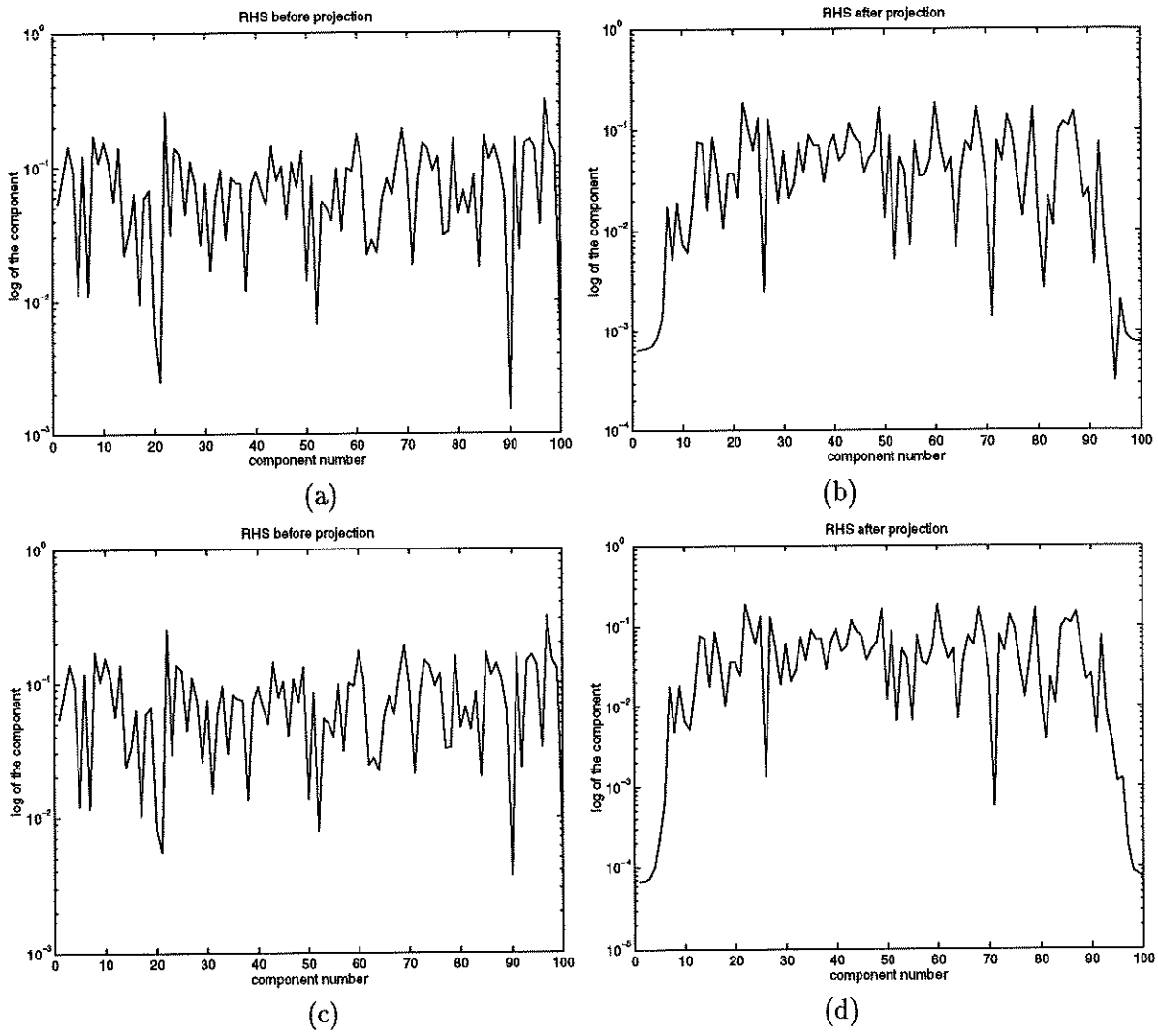

Figure 2: Size distribution of the components of (a) the original right hand side $b^{(2)}$, (b) $b^{(2)}$ after Galerkin projection when $\rho = -1$. Size distribution of the components of (c) the original right hand side $b^{(2)}$, (d) $b^{(2)}$ after Galerkin projection when $\rho = -0.1$.

Linear Systems	(1)	(2)	(3)	(4)	Total
Starting with Projection Method I	36	37	43	49	165
Starting with Projection Method II	36	48	55	76	205
Starting with previous solution	36	54	66	87	243
Starting with random initial guess	38	60	74	98	270
Starting with Projection Method I using preconditioner	9	9	9	11	38
Starting with Projection Method II using preconditioner	9	9	11	16	45
Starting with previous solution using preconditioner	9	13	16	23	61

Table 3: (Example 1) Number of matrix-vector multiplies required for convergence of all the systems. Regularization parameter μ = (1) 0.072, (2) 0.036, (3) 0.018 and (4) 0.009.

by a rank-1 or rank-2 matrices. We will see that the extremal eigenvector components of the right-hand sides are effectively reduced after the projection process. Moreover, the number of iterations required for convergence when we employ the projected solution as initial guess is less than that required in the usual CG process.

Example 1 ($tol = 10^{-4}$): We consider a 2-dimensional deconvolution problem arising in ground-based atmospheric imaging and try to remove the blurring in an image (see Figure 3(a)) resulting from the effects of atmospheric turbulence. The problem consists of a 256-by-256 image of an ocean reconnaissance satellite observed by a simulated ground-based imaging system together with a 256-by-256 image of a guide star (Figure 3(b)) observed under similar circumstances. The data are provided by the Phillips Air Force Laboratory at Kirkland AFB, NM through Prof. Bob Plemmons at Wake Forest University. We restore the image using the identity matrix as the regularization operator suggested in [5] and therefore solve the linear systems (3.10) with different regularization parameters μ. We also test the effectiveness of the preconditioned projection method. The preconditioner we employed here is the block-circulant-circulant-block matrix proposed in [5].

Table 3 shows the number of matrix-vector multiplies required for the convergence of all the systems. Using the projection method, we save on number of matrix-vector multiplies in the iterative process with or without preconditioning. From Table 3, we also see that the performance of Projection Method I is better than that of Projection Method II. For comparison, we present the restorations of the images when the regularization parameters are 0.072, 0.036, 0.018 and 0.009 in Figure 3. We see that when the value of μ is large, the restored image is very smooth, while the value of μ is small, the noise is amplified in the restored image. By solving these multiple linear systems successively by projection method, we can select Figure 3(e) that presents the restored image better than the others.

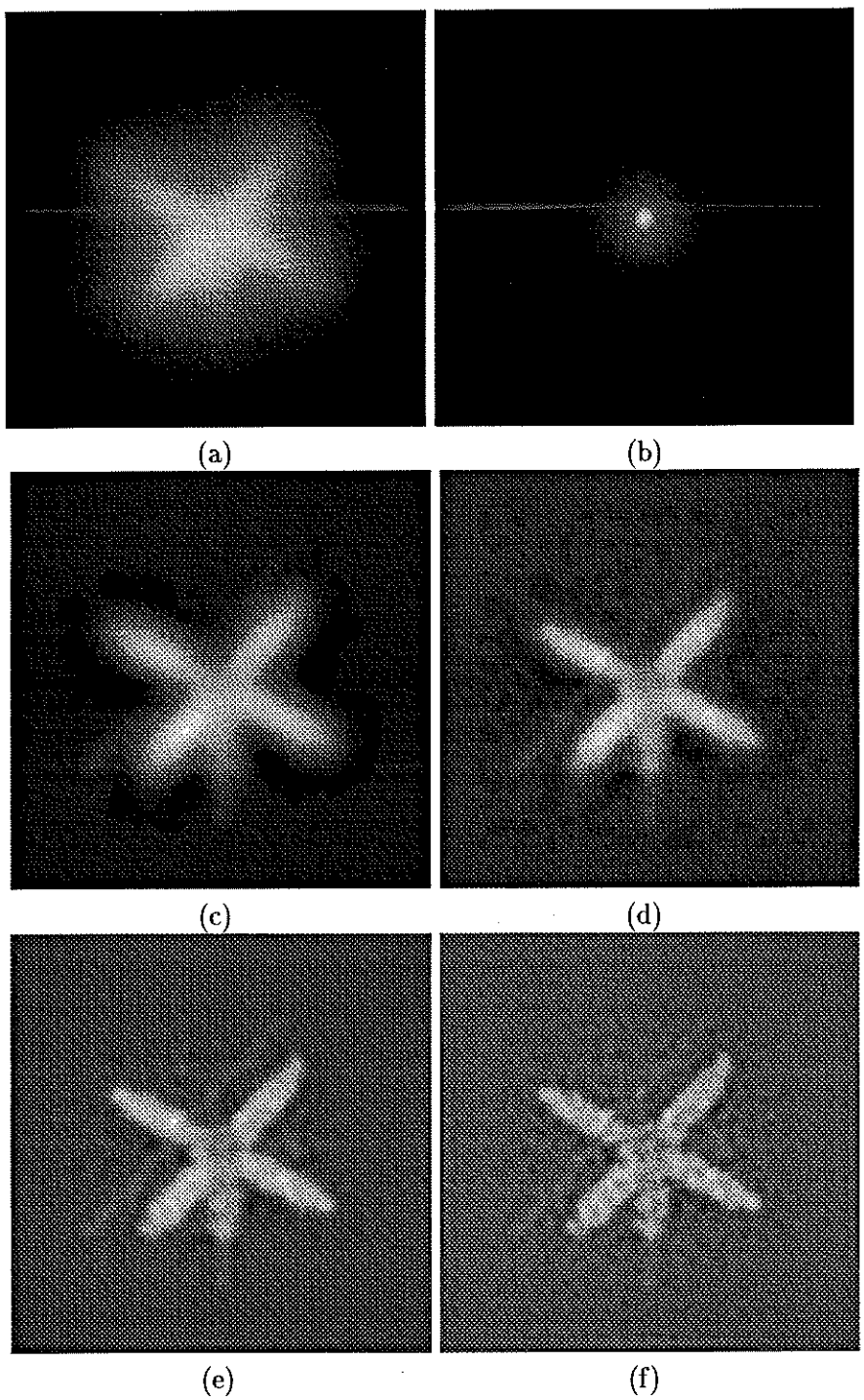

Figure 3: (Example 1) Observed Image (a), guide star image (b), restored images using regularization parameter $\mu =$ (c) 0.072, (d) 0.036, (e) 0.018 and (f) 0.009.

Linear Systems	(1)	(2)	(3)	(4)	(5)	Total
Starting with Projection Method I	45	31	28	25	24	153
Starting with Projection Method II	45	37	32	31	29	174
Starting with previous solution	45	43	44	42	40	214

(a)

Linear Systems	(1)	(2)	(3)	(4)	(5)	Total
Starting with Projection Method I	68	51	45	36	30	165
Starting with Projection Method II	68	55	49	42	35	249
Starting with previous solution	68	61	59	56	54	308

(b)

Table 4: (Example 2) Number of matrix-vector multiplies required for convergence of all the systems. (a) Exponentially-weighted RLS computations and (b) Sliding window RLS computations

Example 2 ($tol = 10^{-8}$): In this example, we test the performance of Projection Methods I and II in the block (sliding window and exponentially-weighted) RLS computations. We illustrate the convergence rate of the method by using the adaptive Finite Impulse Response (FIR) system identification model, see [15]. The second order autoregressive process $x_t + 0.8x_{t-1} + 0.1x_{t-2} = v_t$ where $\{v_t\}$ is a white noise process with variance being 1, is used to construct the data matrix $X(t)$ in §3.2. The reference (unknown) system $w(t)$ is an n-th order FIR filter. The Gaussian white noise measurement error with variance 0.025 is added into the desired response $d(t)$ in §3.2. In the tests, the forgetting factor β is 0.99 and the order n of filter is 100.

In the case of the exponentially-weighted RLS computations, the consecutive systems differ by a rank-1 positive definite matrix, whereas in the case of the sliding window computations, the consecutive systems differ by the sum of a rank-1 positive definite matrix and a rank-1 negative definite matrix. Table 4 lists the number of matrix-vector multiplies required for the convergence of all the systems arising from exponentially-weighted and sliding window RLS computations. We observe that the performance of Projection Method I is better than that of Projection Method II. The projection method requires less matrix-vector multiplies than that using the previous solution as an initial guess. We note from Figures 4 and 5 that the eigenvector components of $b^{(2)}$ are effectively reduced after projection in both cases of exponentially-weighted and sliding window RLS computations. We see that the decreases of eigenvector components when using Projection Method I are indeed greater than those when using Projection Method II.

In the next three examples, we consider more general coefficient matrices, i.e., the consecutive linear systems do not differ by the scaled identity matrix and rank-1 or rank-2 matrices. In these examples, the matrix-vector products for the non-seed iteration may not be computed cheaply,

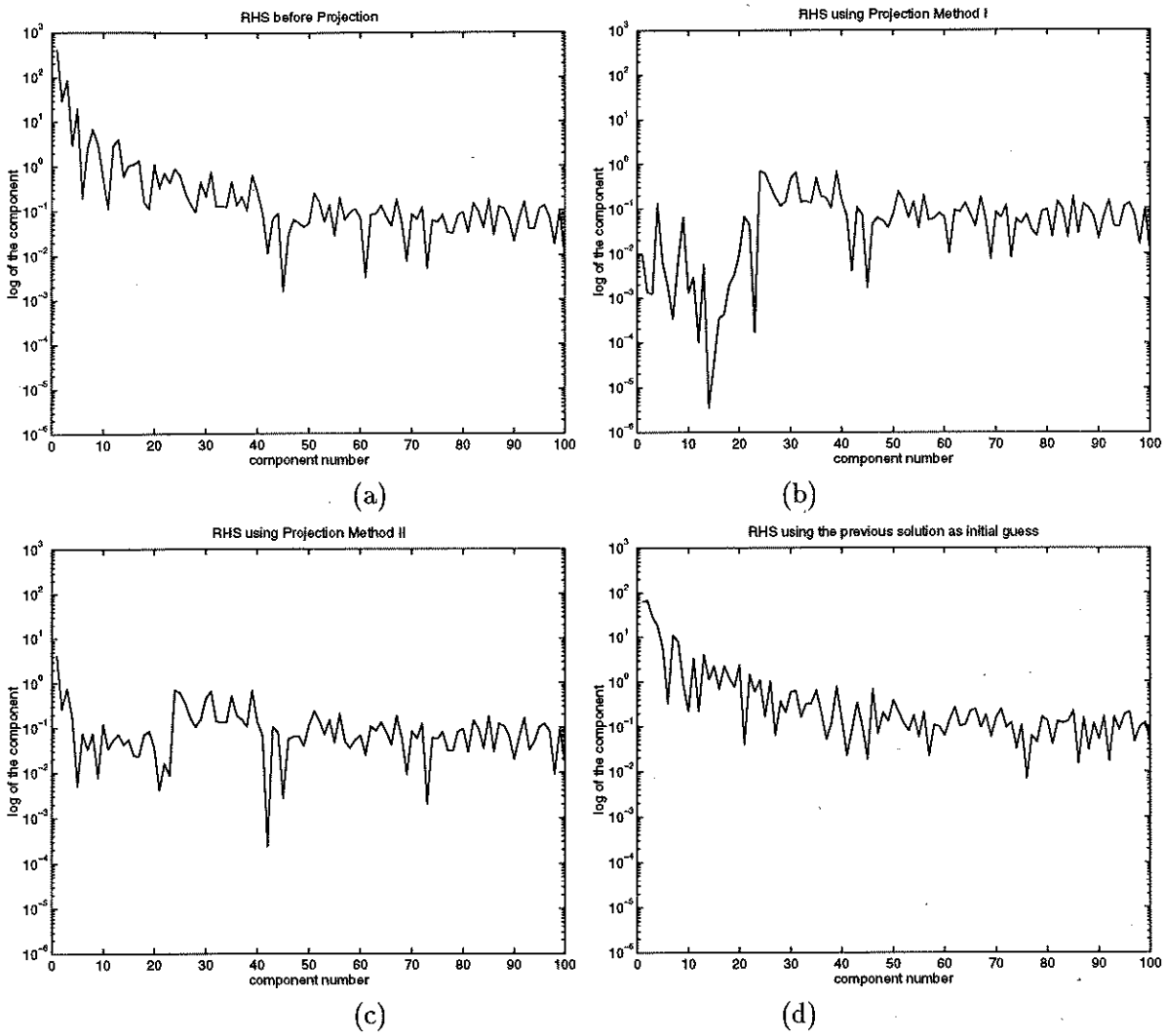

Figure 4: (Example 2) Exponentially-weighted RLS computations. Size distribution of the components of (a) the original right hand side $b^{(2)}$, (b) $b^{(2)}$ after using Projection Method I, (c) $b^{(2)}$ after using Projection Method II, (d) $b^{(2)} - A^{(2)}x^{(1)}$ (using the previous solution as an initial guess).

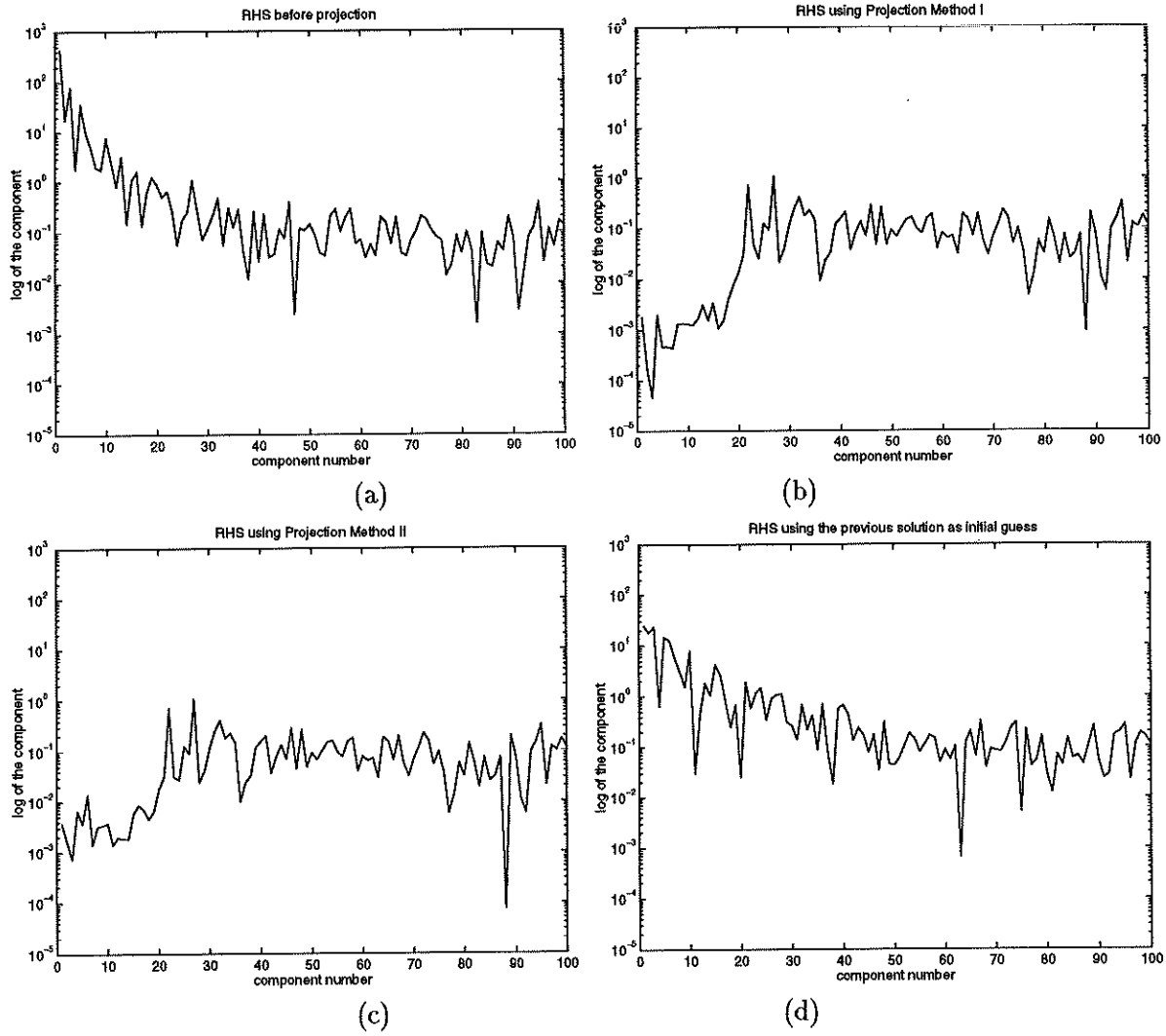

Figure 5: (Example 2) Sliding window RLS computations. Size distribution of the components of (a) the original right hand side $b^{(2)}$, (b) $b^{(2)}$ using Projection Method I, (c) $b^{(2)}$ using Projection Method II, (d) $b^{(2)} - A^{(2)}x^{(1)}$ (using the previous solution as an initial guess).

we therefore only apply Projection Method II to solve the multiple linear systems. However, the same phenomena as in Examples 1 and 2 is observed in these three examples as well.

Example 3 ($tol = 10^{-12}$): In this example, we consider a discrete ill-posed problem, which is a discretization of a Fredholm integral equation of the first kind

$$\int_a^b K(s,t)f(t)dt = g(s), \quad c \le s \le d.$$

The particular integral equation that we shall use is a one dimensional model problem in image reconstruction [7] where an image is blurred by a known point-spread function. The desired solution f is given by

$$f(t) = 2e^{(-4(t-0.5)^2)} + e^{(-4(t+0.5)^2)}, \quad -\frac{\pi}{2} \le t \le \frac{\pi}{2},$$

while the kernel K is the point spread function of an infinitely long slit given by

$$K(s,t) = (\cos s + \cos t)\left\{\frac{\sin[\pi(\sin s + \sin t)]}{[\pi(\sin s + \sin t)]}\right\}^2, \quad -\frac{\pi}{2} \le t \le \frac{\pi}{2}.$$

We use collocation with n (=64) equidistantly spaced points in $[-\pi/2, \pi/2]$ to derive the matrix A and the exact solution x. Then we compute the exact right-hand sides $b = Ax$ and then perturb it by uncorrelated errors (white noise) normally distributed with zero mean and standard derivation 10^{-4}. Here we choose a matrix D equal to the second derivative operator ($D = tridiag(-1, 2-1)$). Different regularization parameters μ are used to compute the L-curve (see Figure 6) and test the performance of the Projection Method II for solving multiple linear systems

$$(\mu_i D^T D + A^T A)x^{(i)} = A^T b, \quad 1 \le i \le s.$$

We emphasize that the consecutive systems do not differ by the scaled identity matrix.

Table 5 shows the number of iterations required for convergence of all 10 systems using Projection Method II and using the previous solution as initial guess having the same residual norm. We see that the projection method requires 288 matrix-vector multiplies to solve all the systems, but the one using the previous solution as initial guess requires 365 matrix-vector multiplies. In particular, the tenth system can be solved without restarting the conjugate gradient process after the projection.

Example 4 ($tol = 10^{-9}$): We consider the integral equation

$$f(s) + \frac{ab}{\pi}\int_0^{2\pi} \frac{f(t)dt}{c^2 + d^2 - (c^2 - d^2)\cos(s+t)} = g(s), \quad 0 \le s \le 2\pi, \tag{4.22}$$

corresponding to the Dirichlet problem for the Laplace equation in the interior of an ellipse with semiaxis $c \ge d > 0$. We solve the case where the unique solution and the right-hand side are given by

$$f(s) = e^{\cos s}\cos(\sin s), \quad \text{and} \quad g(s) = f(s) + e^{z\cos s}\cos(z\sin s), \quad 0 \le s \le 2\pi,$$

Linear Systems	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	Total
Starting with Projection Method II	79	38	33	25	27	26	23	21	15	1	288
Starting with previous solution	79	44	37	34	32	34	28	35	30	23	376

Table 5: (Example 3) Number of matrix-vector multiplies required for convergence of all the systems with $\mu_i = 0.005 \times \frac{1}{2^i}$.

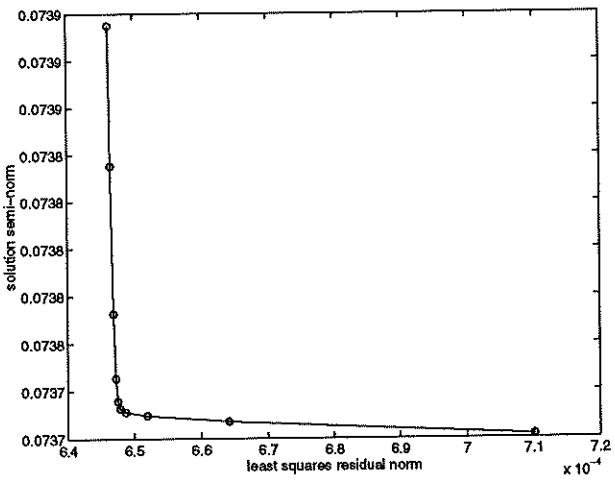

Figure 6: (Example 3) The Tikhonov L-curve with regularization parameters used in Table 4.

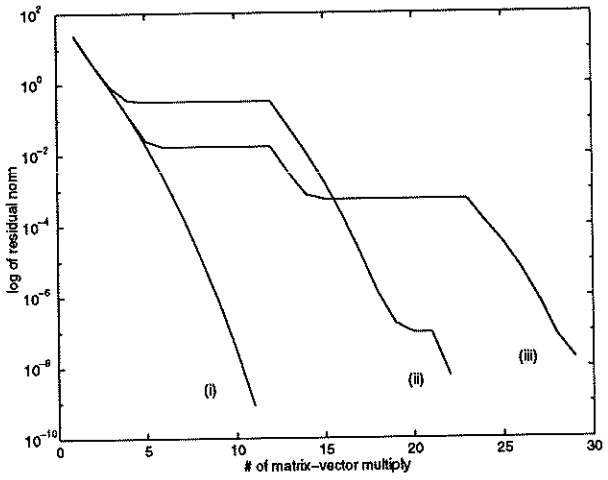

Figure 7: (Example 4) The convergence behaviour of all the systems (i) $c = 2.0516$ and $d = 0.3893$, (ii) $c = 2.3822$ and $d = 0.4265$ and (iii) $c = 4.7374$ and $d = 0.9017$.

where $z = (c-d)/(c+d)$. The coefficient matrices $A^{(k)}$ and the right hand sides $b^{(k)}$ $(k = 1, 2, 3)$ are obtained by discretization of the integral equation (4.22). The size of all systems is 100. The values of c and d are arbitrary chosen from the intervals $[2, 5]$ and $[0, 1]$ respectively. We emphasize that in this example, the consecutive discretized systems do not differ by low rank or small norm matrices.

The convergence behaviour of all the systems is shown in Figure 7. In the plot, each steepest declining line denotes the convergence of a seed and also for the non-seed in the last restart. Note that we plot the residual norm against the cost (the number of matrix-vector multiply) in place of the iteration number so that we may compare the efficiency of these methods. We remark that the shape of the plot obtained is similar to those numerical results given in [6] for the Galerkin projection method for solving linear systems with multiple right hand sides. If we use the solution of the second system as an initial guess for the third system, the number of iteration required is 13. However, the number of iteration required is just 8 for Projection Method II to have the same residual norm as that of the previous solution method; see Figure 8. Figure 9 shows the components of the corresponding right-hand side of the third system before the Galerkin projection, after the projection and using the previous solution as initial guess. The figure clearly reveals that the eigenvector components of $b^{(3)}$ are effectively reduced after the projection.

Example 5 ($tol = 10^{-7}$): The matrices for the final set of experiments corresponding to the three-point centered discretization of the operator $-\frac{d}{dx}(a(x)\frac{du}{dx})$ in $[0, 1]$ where the function $a(x)$ is given by $a(x) = c + dx$, where c and d are two parameters. The discretization is performed using a grid size of $h = 1/65$, yielding matrices of size 64 with different values of c and d. The right hand sides of these systems are generated randomly with its 2-norm being 1. We remark that the consecutive linear systems do not differ by low rank or small norm matrices in this

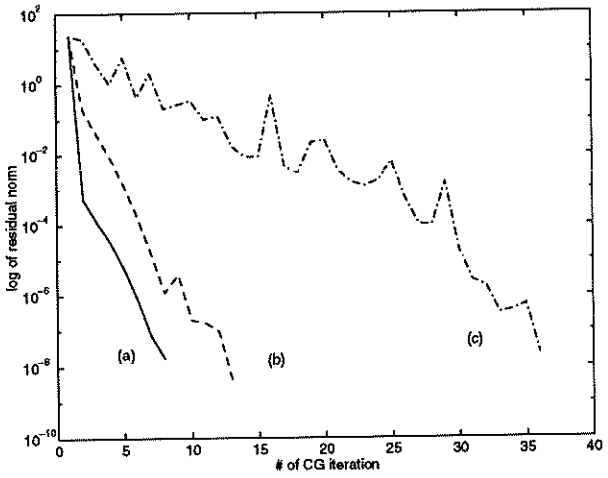

Figure 8: (Example 4) The convergence behaviour of the third system, (a) with projected solution as initial guesses, (b) with previous solution vector as initial guess and (c) with random vector as initial guess.

Linear Systems	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	Total
Starting with Projection Method II	83	82	80	75	62	53	47	21	26	24	553
Starting with previous solution	83	83	83	83	83	83	83	83	84	83	831

Table 6: (Example 5) Number of matrix-vector multiplies required for convergence of all the systems with $c_k = 0.1551 \times 0.9524^k$ and $d_k = 7.7566 \times 0.9524^k$.

example.

Table 6 shows the number of iterations required for convergence of all the systems using Projection Method II and using previous solution as initial guess having the same residual norm. We observe from the results that the one using the projected solution as the initial guess converges faster than that using the previous solution as initial guess. Figure 10 shows the components of the corresponding right-hand side of the seveth system before the Galerkin projection and after the projection. Again, it illustrates that the projection can reduce the eigenvector components effectively.

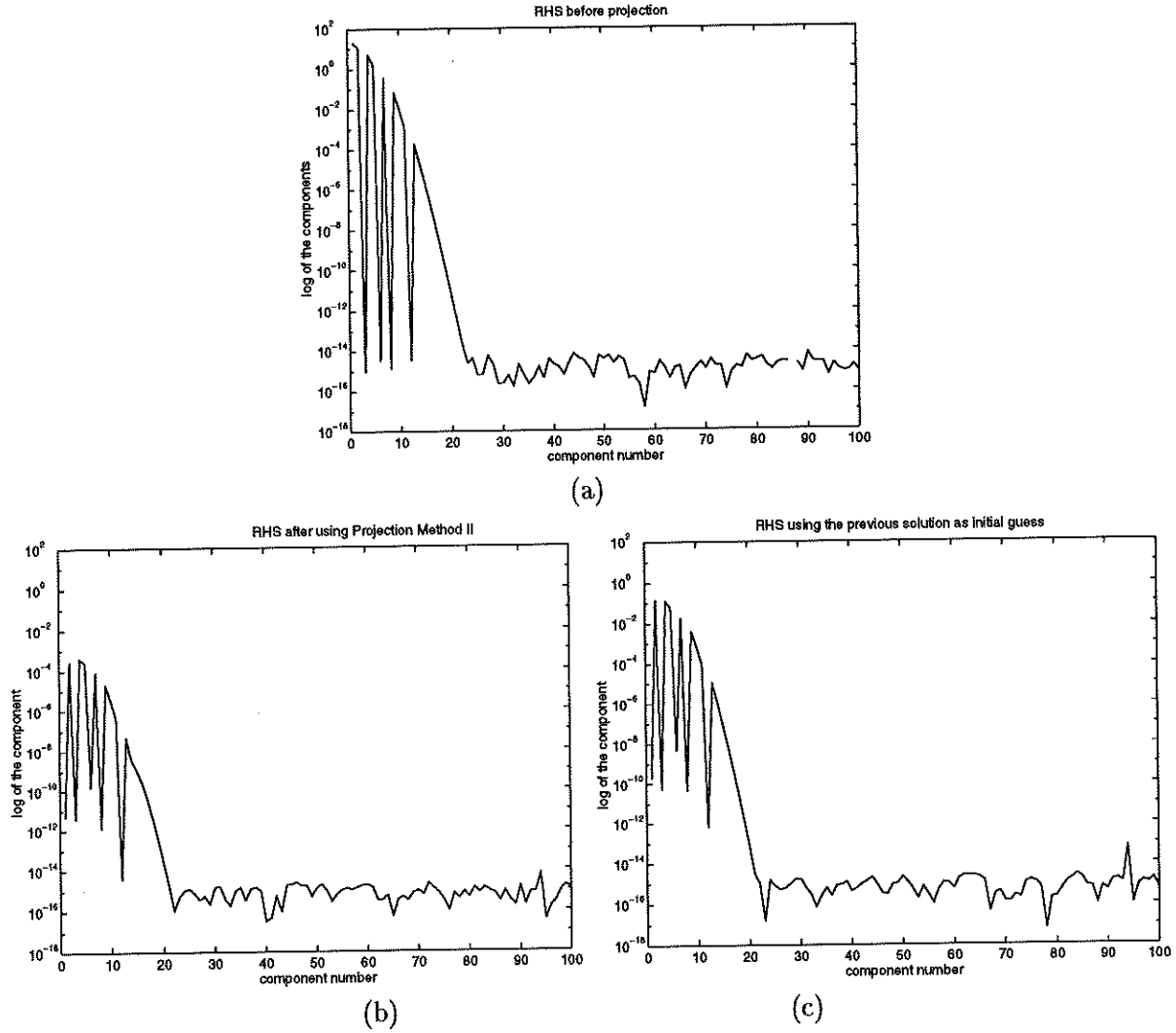

Figure 9: (Example 4) Size distribution of the components of (a) the original right hand side $b^{(3)}$, (b) $b^{(3)}$ after Galerkin projection, (c) $b^{(3)} - A^{(3)}x^{(2)}$ (using the previous solution as an initial guess).

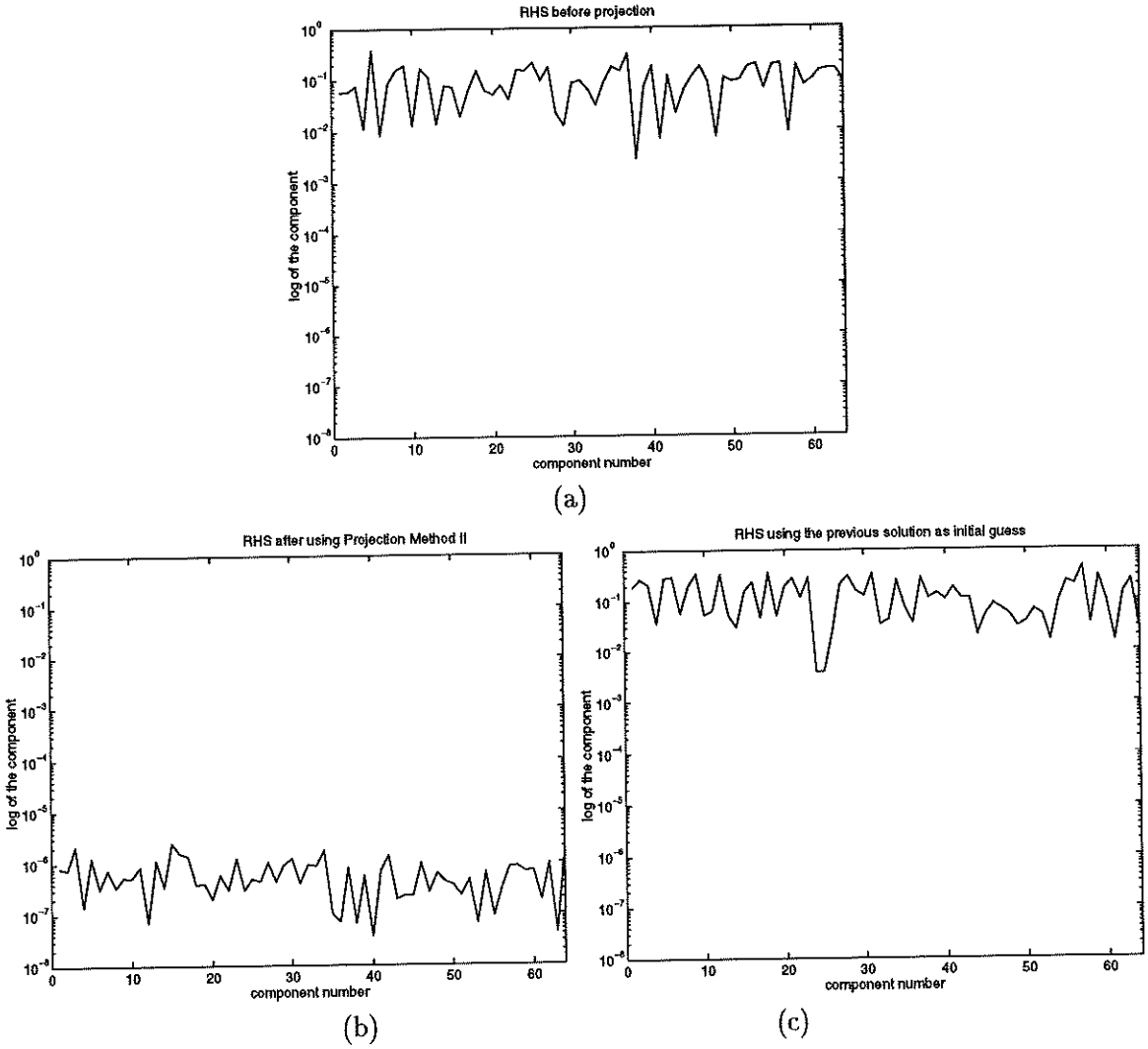

Figure 10: (Example 5) Size distribution of the components of (a) the original right hand side $b^{(7)}$, (b) $b^{(7)}$ after Galerkin projection and (c) $b^{(7)} - A^{(7)}x^{(6)}$ (using the previous solution as an initial guess).

5 Concluding Remarks

In this paper, we developed Galerkin projection methods for solving multiple linear systems. Experimental results show that the method is an efficient method. We end with concluding remarks about the extensions of the Galerkin projection method.

1. A block generalization of the Galerkin projection method can be employed in many applications. The method is to select more than one system as seed so that the Krylov subspace generated by the seed is larger and the initial guess obtained from the Galerkin projection onto this subspace is expected to be better. One drawback of the block method is that it may break down when singularity of the matrices occurs arising from the conjugate gradient process. For details about block Galerkin projection methods, we refer to Chan and Wan [6].

2. The literature for nonsymmetric systems with multiple right-hand sides is vast. Two methods that have been proposed are block generalizations of solvers for nonsymmetric systems; the block biconjugate gradient algorithm [17, 16], block GMRES [25], block QMR [4, 9]. Recently, Simoncini and Gallopoulos [23] proposed a hybrid method by combining the Galerkin projection process and Rishardson acceleration technique to speed up the convergence rate of the conjugate gradient process. In the same spirit, we can modify the above Galerkin projection algorithms to solve nonsymmetric systems with multiple coefficient matrices and right-hand sides.

References

[1] J. Abbiss and P. Earwicker, *Compact operator equations, regularization and super-resolution*, in Mathematics in Signal Processing, Clarendon Press, Oxford, 1987.

[2] S. Alexander, *Adaptive Signal Processing*, Springer Verlag, New York, 1986.

[3] A. Björck, *Least squares methods*, in Handbook of Numerical Methods, ed. P. Ciarlet and J. Lions, Elsevier, North Holland V1, 1989.

[4] W. Boyse and A. Seidl, *A Block QMR Method for Computing Multiple Simultaneous Solutions to Complex Symmetric Systems*, SIAM J. on Sci. Comput. 17 (1996), pp. 263–274.

[5] R. Chan, M. Ng and R. Plemmons, *Generalization of Strang's Preconditioner with Applications to Toeplitz Least Squares Problems*, Numerical Linear Algebra with Applications, 3 (1996), pp. 45–64.

[6] T. Chan and W. Wan, *Analysis of Projection Methods for Solving Linear Systems with Multiple Right-hand Sides*, CAM Rep. 94-26, Department of Math., UCLA, to appear in SIAM J. on Sci. Comput.

[7] P. HANSEN, *Analysis of Discrete Ill-posed Problems by Means of the L-curve*, SIAM Review, 34 (1992), pp. 561-580.

[8] G. CLARK AND S. MITRA, *Block Implementation of Adaptive Digital Filters*, IEEE Trans. Circuits and Systems, CAS-28 (1981).

[9] R. FREUND AND M. MALHOTRA, *A Block-QMR Algorithm for Non-Hermitian Linear Systems with Multiple Right-Hand Sides*, Manuscript SCCM-96-01, Scientific Computing and Computational Mathematics Program, Stanford University, 1996.

[10] G. GOLUB AND C. LOAN, *Matrix Computations*, Johns Hopkins Press, Baltimore, Second Edition, 1989.

[11] G. GOLUB, *Some Modified matrix Eigenvalue Problems*, SIAM Review, 15 (1973), pp. 318–334.

[12] C. GROETSCH, *The Theory of Tikhonov Regularization for Fredholm Equations of the First Kind*, Pittman Publishing, Boston, 1984.

[13] A. JAIN, *Fundamentals of Digital Image Processing*, Prentice-Hall, Engelwood Cliffs, NJ 1989.

[14] R. KRESS, *Linear integral equations*, Springer-Verlag, New York, 1989.

[15] M. NG AND R. PLEMMONS, *Fast RLS Adaptive Filtering by FFT–Based Conjugate Gradient Iterations*, SIAM J. on Sci. Comp., 17, No. 4 (1996), pp. 154–170.

[16] A. NIKISHIN AND A. YEREMIN, *Variable Block CG Algorithms for Solving Large Sparse Symmetric Positive Definite Linear Systems on Parallel Computers, I: General Iterative Scheme.*, SIAM J. on Matrix Ana., to appear.

[17] D. O'LEARY, *The block conjugate gradient algorithm and realted methods*, Linear Algebra Appl., 29 (1980), pp. 293-322.

[18] M. PAPADRAKAKIS AND S. SMEROU, *A New Implementation of the Lanczos Method in Linear Problems*, International Journal for Numerical Methods in Engineering, 29 (1990), pp. 141–159.

[19] B. PARLETT, *A New Look at the Lanczos Algorithm for Solving Symmetric Systems of Linear Equations*, Linear Algebra Appl., 29 (1980), pp. 323–346.

[20] R. PLEMMONS, *FFT–based RLS in Signal Processing*, Proc. IEEE Confer. on Acoustics, Speech & Signal Processing ICASSP-93, 3 (1993), pp. 571–574.

[21] Y. SAAD, *On the Lanczos Method for Solving Symmetric Linear Systems with Several Right-Hand Sides*, Math. Comp., 48 (1987), pp. 651–662.

[22] V. SIMONCINI AND E. GALLOPOULOS, *A Memory-conserving Hybrid Method for Solving Linear Systems with Multiple Right-hand Sides*, in Copper Mountain Conf. Iterative Methods, April 1992.

[23] V. SIMONCINI AND E. GALLOPOULOS, *An Iterative Method for Nonsymmetric Systems with Multiple Right-hand Sides*, SIAM J. Sci. Comp., 16 (1995), pp. 917–933.

[24] C. SMITH, A. PETERSON AND R. MITTRA, *A Conjugate Gradient Algorithm for the Treatment of Multiple Incident Electromagnetic Fields*, IEEE Transactions On Antennas and Propagation, 37 (1989), pp. 1490–1493.

[25] B. VITAL, *Etude de quelques méthodes de résolution de problémes linéaires de grande taille sur multiprocesseur*, Ph.D. thesis, Iniversité de Rennes, Rennes, France, Nov., 1990.

[26] *H. Vorst, An Iteration Solution Method for Solving $f(A) = b$, using Krylov Subspace Information Obtained for the Symmetric Positive Definite Matrix A*, Journal of Computational and Applied Mathematics, 18 (1987), pp. 249–263.