NUMERICAL SOLUTION OF SADDLE POINT PROBLEMS BY PROJECTION

by

GUL KARADUMAN

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2017

To my parents Selfinaz and Kazim,

And my sisters Yesim, Gonca, Yeliz

Without whom none of my success would be possible.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Prof. Ren-Cang Li, my advisor, for his support, guidance and caring. Without him, it would have been impossible to write this thesis. I also would like to thank my thesis committee Prof. David Jorgensen, Prof. Tuncay Aktosun, and Prof. Guojun Liao for their time to review my dissertation and attend my defense.

I also wish to thank everyone in the Mathematics department at UTA, for being supportive, encouraging and kind since day one. I could not wish for a better environment than UTA for achieving a doctorate degree. Mei Yang, my academic sister, deserves special thanks for being a great friend and coworker. We have broadened our knowledge with insightful research discussions and have helped each other in every way we can during our studies. I am also grateful to my friends who have been there to help me in any way possible until the last moment.

Last but not least, I would like to express my indebtedness to my parents and sisters for their love, support, and encouragement throughout the years. Without their presence, none of my success would have been possible. I cannot wait to reunite with them.

November 17, 2017

ABSTRACT

NUMERICAL SOLUTION OF SADDLE POINT PROBLEMS BY PROJECTION

Gul Karaduman, Ph.D.

The University of Texas at Arlington, 2017

Supervising Professor: Ren-Cang Li

In this thesis, we work on iterative solutions of large linear systems of saddle point problems of the form

$$\begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix},$$

where $A \in \mathbb{R}^{n\times n}$, $B_1, B_2 \in \mathbb{R}^{m\times n}$, $f \in \mathbb{R}^n$, and $n \geq m$. Many applications in computational sciences and engineering give rise to saddle point problems such as finite element approximations to Stokes problems, image reconstruction, tomography, genetics, statistics and model order reduction for dynamical systems. Such problems are typically large and sparse.

We develop new techniques to solve the saddle point problems depending on the rank of B_2. First, we deal with the case when B_2 has full row rank, i.e., $\mathrm{rank}(B_2) = m$. The key idea is to construct a projection matrix and transform the original problem to a least squares problem then solve the least squares problem by using one of the iterative methods such as LSMR. In most applications B_2 has full rank, but not always. Next, we turn to the saddle point systems with the rank-deficient matrix

B_2. Similarly we construct a new projection matrix by using only maximal linearly independent rows of B_2. By using this projection matrix, the original problem can still be transformed into a least squares problem. Again, the new system can be solved by using one of the iterative techniques for least squares problems. Numerical experiments show that the new iterative solution techniques work very well for large sparse saddle point systems with both full rank and rank-deficient matrix B_2.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ALGORITHMS

Algorithm Page

LIST OF TABLES

LIST OF ABBREVIATIONS

KKT	Karush-Kuhn-Tucker
CG	Conjugate Gradient Method
MINRES	Minimal Residual Method
GMRES	Generalized Minimum Residual Method
GMRES(k)	Restarted Generalized Minimum Residual Method
LSMR	Least Squares Minimal Residual Method
TOL	Tolerance
LU	LU Factorization
LS	Least-Squares

LIST OF SYMBOLS

$\mathcal{A}$	Saddle Point Matrix
A, B_1, B_2	Sparse matrices
$x, y, f, g, u, v, ...$	Vectors
$\text{rank}(B_2)$	rank of matrix B_2
A^T	Transpose of matrix A
A^{-1}	Inverse of matrix A
M/K_1	Schur complement of K_1 in the block matrix M
$\nabla(f)$	Gradient of f
$\mathcal{K}_i\{M, r_0\}$	i-th Krylov subspace of M on r_0
$\perp$	is perpendicular to
$\lVert\cdot\rVert_2$	Vector 2-norm or the induced matrix 2-norm
$x_*,\ y_*$	The unique solution to the saddle point system $\begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}$
I	Identity matrix
e_j	j-th column of I

CHAPTER 1

INTRODUCTION

1.1 Introduction

A saddle point system is a linear system with the following 2-by-2 block structure:

$$\mathcal{A}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} A & {B_1}^T \\ B_2 & -C \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a square matrix of order n, B_1 and $B_2 \in \mathbb{R}^{m\times n}$ are rectangular matrices with $n \geq m$, and $C \in \mathbb{R}^{m\times m}$ is a square matrix of order m. Vectors $f \in \mathbb{R}^n$ and $g \in \mathbb{R}^m$ are right-hand side vectors and $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ are the solution vectors. The 2×2 block coefficient matrix $\mathcal{A}$ is called a *saddle point matrix*. The coefficient matrix of the saddle point system is usually large and sparse. 'KKT problem' which stand for Karush-Kuhn-Tucker problem is also used as an alternate name for the saddle point problem in some sources. Benzi, Golub, and Liesen [1] gave a definition of a saddle point problem as the constituent blocks A, B_1, B_2 and C satisfy one or more of the following conditions:

(i) A is symmetric;

(ii) The symmetric part of A, $H = \frac{1}{2}(A + A^T)$ is positive semidefinite;

(iii) $B_1 = B_2 = B$;

(iv) C is symmetric and positive definite;

(v) $C = 0$.

In this work, our focus is on the solution of the system (1.1) when condition (v) is satisfied and g is a zero vector. Therefore the subject of this thesis is to solve the saddle point system of the form

$$\begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix} \quad or \quad \mathcal{A}z = b, \tag{1.2}$$

where $A \in \mathbb{R}^{n\times n}$, $B_1 \in \mathbb{R}^{m\times n}$, $B_2 \in \mathbb{R}^{m\times n}$, and $f \in \mathbb{R}^n$. Some applications give rise to such saddle point problems have very large saddle point matrices, e.g., $n + m \approx 10^5$ or larger. Also, most of the entries of these matrices are zero. Such matrices are called sparse matrices.

In this system, no assumption is required for matrices A and B_1. The (2,1)-block matrix $A \in \mathbb{R}^{n\times n}$ can be any large and sparse square matrix with size $n \times n$. Similarly, the (1,2)-block matrix ${B_1}^T \in \mathbb{R}^{n\times m}$ can be any large and sparse rectangular matrix with size $n \times m$. The right-hand side vector $f \in \mathbb{R}^n$ is also any vector. The solution technique will have different form according to the rank of the (2,1)-matrix B_2.

The system (1.2) often arises from the first order optimality conditions for the following equality constrained quadratic programming problem

$$\begin{aligned} &\text{minimize } h(x) = x^T Ax + x^T {B_1}^T y - f^T x \\ &\text{subject to } B_2 x = 0, \end{aligned}$$

where $A \in \mathbb{R}^{n\times n}$, $B_1 \in \mathbb{R}^{m\times n}$ and $B_2 \in \mathbb{R}^{m\times n}$ are sparse matrices and $f \in \mathbb{R}^n$. We define the corresponding Lagrangian function

$$\begin{aligned} \mathcal{L}(x, y) &\equiv h(x) + y^T B_2 x \\ &= x^T Ax + x^T {B_1}^T y - f^T x + y^T B_2 x \\ &= x^T Ax + x^T {B_1}^T y + y^T B_2 x - x^T f \end{aligned}$$

$$= \begin{bmatrix} x \\ y \end{bmatrix}^T \begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} x \\ y \end{bmatrix}^T \begin{bmatrix} f \\ 0 \end{bmatrix},$$

where the variable y represents the vector of Lagrangian multipliers. Any solution (x_*, y_*) of (1.2) is a saddle point (optimal solution) for the Lagrangian $\mathcal{L}(x, y)$.

To find the saddle points of $\mathcal{L}(x, y)$, we need to solve the following system

$$\bigtriangledown \mathcal{L}(x, y) = 0.$$

This explains why the name saddle point problem is given to the system (1.2). For more information on the quadratic programming problems, we suggest Nocedal and Wright [2].

An efficient and stable numerical solution for such a large class of problem is one of the fundamental duties in the numerical linear algebra. The goal of this thesis is to develop effective and efficient methods for the solution of (1.2). We only consider the problems that have large and sparse real coefficient matrices.

In this thesis, we developed new solution techniques to solve the saddle point matrix equation $\mathcal{A}z = b$ in (1.2), depending on the rank of matrix B_2. The idea can be straight forwardly extended to complex coefficient matrices. First, we deal with the case when B_2 has full row rank, i.e., $\text{rank}(B_2) = m$. The key idea is to construct a projection matrix and transform the original problem to a least squares problem then solve the least squares problem by using one of the iterative methods such as LSMR. In most applications, the (2,1)-block matrix in the saddle point problem has full rank, but not always. Next, we turn to the saddle point systems with the rank-deficient matrix B_2, i.e., $\text{rank}(B_2) < m$. Similarly we construct a new projection matrix by using only maximal linearly independent rows of B_2. By using this projection matrix, the original problem can still be transformed into a least squares problem. Again, the new system can be solved by using one of the iterative techniques for least squares problems. For both cases, the number of rows in matrix B_2 is much smaller than the number of columns in B_2, i.e., $m \leq n$.

1.2 Applications

In past years, large linear systems of saddle point problems arise frequently in a number of areas including computational science and engineering. For this reason, quite an amount of work has been on solving saddle point problems. We list some of the fields where saddle point systems are used when the block matrices A, B_1, B_2, and C satisfy some or all the conditions:

- Optimal control [21, 14, 24, 16],
- Computational fluid dynamics [30, 18, 19],
- Constrained optimization [20, 22],
- Least squares estimation [5],
- Electromagnetism [23],
- Mixed formulations of elliptic PDEs [25],
- Model order reduction for dynamical systems [36],
- Finite element discretization [27],
- Metal deformation [28],
- Image reconstruction, tomography [29],

- Finance [33, 13],
- Mesh analysis in computer graphics [17],
- Economics [35, 26],
- Linear elasticity [42],
- Domain decomposition [38].

We refer a survey of numerical solution techniques for saddle point problems by Benzi, Golub, and Liesen [1] for an extensive list of the fields where saddle point problems arise, together with some of the references.

The remainder of this thesis is organized as follows. In Chapter 2, we review some important properties of the saddle point matrix $\mathcal{A}$, give an overview of existing solution algorithms for saddle point problems, and emphasize the importance of the Krylov subspace approximation techniques for large-scale systems. Then we give a brief overview of the two Krylov subspace iterative methods: Generalized Minimal Residual which is known as GMRES [9] and Least Square Minimal Residual which is known as LSMR [12]. We also list some of the major preconditioning methods in the literature for saddle point problems.

In Chapter 3, we begin describing the general theory of the solution method for saddle point problem by using a projection matrix when B_2 is a full rank matrix i.e., $\mathrm{rank}(B_2) = m$. In this solution technique, we assume that m is small. For this solution technique, we construct a projection matrix and transform the original problem into a least squares problem then solve the least squares system by using one of the iterative methods such as LSMR. Then we present the algorithmic framework of the method.

Chapter 4 focuses on the solution method for the saddle point problem when B_2 is rank-deficient. Since $B_2 \in \mathbb{R}^{m \times n}$ does not have full rank, we need to form a different projection matrix by using the linearly independent rows of B_2. Our discussion will focus on how to construct the projection matrix and solve the transformed problem by using an iterative method. At the end of the chapter, we present an algorithmic framework of the method.

In Chapter 5, we give our numerical results to show the performances of our projected method for saddle point problems. All the testing matrices are taken from SuiteSparse matrix collection, formerly the University of Florida sparse matrix collection [47].

Finally, we make conclusions and an outlook to our future work for the solution of saddle point systems in Chapter 6.

CHAPTER 2

BACKGROUND

This chapter contains reviews of the fundamental properties of the saddle point matrix $\mathcal{A}$, the existing solution approaches, Krylov subspace methods including two common used iterative methods, GMRES and LSMR, and surveys some preconditioning methods in the literature for generalized saddle point problems.

2.1 Properties of Saddle Point Matrices

In this section, we introduce the fundamental properties of the saddle point matrices, such as factorizations and invertibility of the saddle point matrix $\mathcal{A}$. We start with the definition of the Schur complement of a matrix block.

Definition 2.1.1 (Schur Complement)**.** Let the block matrices K_1, K_2, K_3 and K_4 be respectively $p \times p$, $p \times q$, $q \times p$ and $q \times q$ matrices, and suppose K_1 is invertible. Let

$$\mathcal{M} = \begin{bmatrix} K_1 & K_2 \\ K_3 & K_4 \end{bmatrix},$$

so that $\mathcal{M}$ is a $(p+q) \times (p+q)$ matrix. Then the Schur complement of the block K_1 in the block matrix $\mathcal{M}$ is $\mathcal{M}/K_1 := K_4 - K_3 {K_1}^{-1} K_2$.

2.1.1 Block Factorizations of a Saddle Point Matrix

In this section, we will show a few block factorizations of the saddle point matrix $\mathcal{A}$. Assume that A is a nonsingular matrix. Then $\mathcal{A}$ admits the following block triangular factorization

$$\begin{aligned}
\mathcal{A} = \begin{bmatrix} A & {B_1}^T \\ B_2 & -C \end{bmatrix} &= \begin{bmatrix} A & 0 \\ B_2 & S \end{bmatrix} \begin{bmatrix} I & A^{-1}{B_1}^T \\ 0 & I \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ B_2A^{-1} & I \end{bmatrix} \begin{bmatrix} A & {B_1}^T \\ 0 & S \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ B_2A^{-1} & I \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & A^{-1}{B_1}^T \\ 0 & I \end{bmatrix},
\end{aligned} \tag{2.1}$$

where $S = -(C + B_2A^{-1}{B_1}^T)$ is the Schur complement of A in the block matrix $\mathcal{A}$.

2.1.2 Inverse of a Saddle Point Matrix

Assume that the (1,1) block $A \in \mathbb{R}^{n\times n}$ of the saddle point coefficient matrix $\mathcal{A}$ is nonsingular. Then $\mathcal{A}$ is nonsingular if and only if the Schur complement matrix $S = -(C + B_2A^{-1}{B_1}^T)$ is nonsingular. Based on the last factorization (2.1) we find the following expression for the inverse

$$\begin{aligned}
\begin{bmatrix} A & {B_1}^T \\ B_2 & -C \end{bmatrix}^{-1} &= \begin{bmatrix} I & A^{-1}{B_1}^T \\ 0 & I \end{bmatrix}^{-1} \begin{bmatrix} A^{-1} & 0 \\ 0 & S^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ B_2A^{-1} & I \end{bmatrix}^{-1} \\
&= \begin{bmatrix} A^{-1} + A^{-1}{B_1}^TS^{-1}B_2A^{-1} & -A^{-1}{B_1}^TS^{-1} \\ -S^{-1}B_2A^{-1} & S^{-1} \end{bmatrix}.
\end{aligned}$$

2.2 Solution Approaches

Algorithms for the saddle point problems can be categorized into two types of segregated and coupled methods. Segregated methods include direct methods, iterative methods or the combination of these two. On the other hand, coupled methods can be either direct or iterative method. More details will be given in the later sections for segregated and coupled methods. Now, we will briefly explain what we mean by the direct and iterative methods.

Direct methods calculate solutions of the systems in a prescribed, finite number of steps. These methods produce the exact result in the absence of rounding error. If the system is of reasonable size, these methods are good for use. However, if the system is a large-scale linear system, a direct method requires a huge memory storage and large computational times which make the method unfavorable.

On the other hand, iterative methods provide an approximated solution of the systems. These systems can be quite large. Iterative methods are very useful for very large systems, where direct methods would be prohibitively expensive even impossible in some cases.

2.2.1 Krylov Subspace Methods

Krylov subspace methods are among the most popular methods in numerical linear algebra. These methods are iterative techniques for the solution of large and sparse linear systems. Krylov subspace methods are matrix-free iterative methods which means they only require matrix-vector multiplications. These methods solve linear systems of the form $Ms = t$, where $M \in \mathbb{R}^{m\times n}$ is a large and sparse matrix, $t \in \mathbb{R}^m$ is a real vector and $s \in \mathbb{R}^n$ is a real vector. Krylov subspace methods converge to the exact solution in a finite number of iterations by solving an iterative sequence of minimization problems.

To briefly explain the methods, we consider for example $m = n$. Krylov subspace methods produce an approximate solution s_i of $Ms = t$, at every iteration i such that

$$s_i \in s_0 + \mathcal{K}_i \{M, r_0\}, \tag{2.2}$$

where s_0 is an arbitrary initial guess to the solution, $r_0 = t - Ms_0$ and $\mathcal{K}_i \{M, r_0\}$ is the i-th Krylov subspace of M on r_0

$$\mathcal{K}_i(M, r_0) = \text{span}\left\{r_0, Mr_0, M^2 r_0, ..., M^{i-1} r_0\right\}. \tag{2.3}$$

These subspaces form a nested sequence, i.e.,

$$\mathcal{K}_1(M, r_0) \subset \mathcal{K}_2(M, r_0) \subset ... \subset \mathcal{K}_d(M, r_0) = ... = \mathcal{K}_m(M, r_0),$$

where $d \equiv \dim \mathcal{K}_m(M, r_0) \leq m$.

The corresponding residual for the approximate solution s_i is $r_i = t - Ms_i$. To determine s_i we force that r_i satisfies the Petrov-Galerkin condition, that is the i-th residual $r_i = t - Ms_i$ is orthogonal to an i-dimensional space $\mathcal{D}_i$,

$$r_i \perp \mathcal{D}_i, \tag{2.4}$$

where $\mathcal{D}_i$ is a constraint space and belongs to another set of nested subspaces.

Different Krylov subspace methods can be constructed by choosing different nested subspaces $\mathcal{D}_i$.

Krylov subspace methods can be categorized according to the size of the coefficient matrix of the linear system of the form $Ms = t$. There are two main class of Krylov subspace methods: methods which solve $Ms = t$ when M is square ($m = n$) and the methods that solve $Ms = t$ when M is either rectangular ($m < n$ or $m > n$) or square matrix.

In the next subsections, we introduce the Arnoldi [6] procedure which is used for the solution of square systems and the Golub-Kahan [46] procedure which is used for rectangular or square systems.

2.2.2 Arnoldi Process

In this section, we focus on nonsymmetric linear systems. The Arnoldi process [6] transforms a nonsymmetric square matrix M into an upper Hessenberg matrix H_k. The process was proposed by Arnoldi in 1951 [6]. We summarize the process as in Algorithm 2.1.

The process can be expressed as

$$\begin{aligned} MV_k &= V_k H_k + v_{k+1} h_{k+1,k} e_k^T \\ &= V_{k+1} \check{H}_k, \\ {V_k}^T M V_k &= H_k, \end{aligned} \tag{2.5}$$

where the columns of $V_{k+1} = \begin{bmatrix} V_k & v_{k+1} \end{bmatrix}$ with $V_k = \begin{bmatrix} v_1 & v_2 & \dots & v_k \end{bmatrix}$ having orthonormal columns and both $\check{H}_k = \begin{bmatrix} H_k \\ \beta_{k+1} {e_k}^T \end{bmatrix}$ and

$$H_k = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1k} \\ \beta_2 & h_{22} & \cdots & \cdots & h_{2k} \\ 0 & \beta_3 & \cdots & \cdots & h_{3k} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & \beta_k & h_{kk} \end{bmatrix}$$

are upper Hessenberg matrices. Arnoldi vectors $\{v_1, v_2, ..., v_{k+1}\}$ span the Krylov subspace $\mathcal{K}_{k+1}(M, t)$. In the Arnoldi process, all Arnoldi vectors must be kept in order to generate the next vector.

Algorithm 2.1 Arnoldi Process

Given any square matrix M and a vector t, this algorithm construct an upper Hessenberg matrix H_k and an orthogonal transformation V_k such that ${V_k}^T M V_k = H_k$.

$\beta = \|t\|_2$, $v_1 = t/\beta$
for $k = 1, 2, ..., n$ **do**
  $w = Mv_k$
  for $i = 1, 2, ..., k$ **do**
    $h_{ik} = w^T v_i$
    $w = w - h_{ik} v_i$
  end for
  $\beta_{k+1} = \|w\|_2$
  $v_{k+1} = w/\beta_{k+1}$
end for

2.2.3 GMRES

Generalized Minimal Residual (GMRES) [9] method was discovered by Saad and Schultz in 1986 for solving $Ms = t$ when M is a square matrix. Algorithm 2.2 is the summary of the Generalized Minimal Residual method.

GMRES uses the Arnoldi process to find an upper Hessenberg matrix H_k and an orthonormal matrix $V_k = [v_1, v_2, ..., v_k]$ in particular $v_1 = r_0/\beta$ where $\beta = \|r_0\|_2$.

Let

$$s = s_0 + \hat{s}, \tag{2.6}$$

where s_0 is an initial guess and $\hat{s} \in \mathcal{K}_k(M, r_0)$. Any $\hat{s} \in \mathcal{K}_k(M, r_0)$ can be written as

$$\hat{s} = V_k \hat{y} \tag{2.7}$$

for some $\hat{y} \in \mathbb{R}^k$. We also have the equality in (2.5). Then the residual vector can be expressed as

$$\begin{aligned} r = t - Ms = t - M(s_0 + \hat{s}) &= r_0 - M\hat{s} \\ &= r_0 - MV_k\hat{y} \\ &= r_0 - V_{k+1}\check{H}_k\hat{y} \\ &= V_{k+1}(\beta e_1 - \check{H}_k\hat{y}). \end{aligned}$$

Since the column vectors of V_{k+1} are orthonormal,

$$\min_{\hat{s} \in \mathcal{K}_k(M, r_0)} \|t - M(s_0 + \hat{s})\|_2 = \min_{\hat{y} \in \mathbb{R}^k} \|\beta e_1 - \check{H}\hat{y}\|_2,$$

where $\beta = \|r_0\|_2$. Once the optimal solution $\hat{y}$ is solved from the last minimization problem s can be obtained by $s = s_0 + V_k\hat{y}$.

Algorithm 2.2 GMRES

Given any initial solution guess $s_0 \in \mathbb{R}^n$, this algorithm computes a generalized minimal residual solution to the linear system $Ms = t$.

```
β = ‖t‖₂, v₁ = t/β
for k = 1, 2, ..., n do
    w = M v_k
    for i = 1, 2, ..., k do
        h_ik = w^T v_i
        w = w − h_ik v_i
    end for
    β_{k+1} = ‖w‖₂
    v_{k+1} = w/β_{k+1}
end for
Form the solution s_k = s₀ + V_k ŷ where ŷ minimizes ‖βe₁ − Ȟy‖₂
```

2.2.4 Golub-Kahan Process

The Golub-Kahan bidiagonalization process partially transforms $[t, M]$ to an upper bidiagonal form $[\beta_1 e_1, N_k]$ by constructing orthonormal matrices U_k and V_k. The process was proposed by Golub and Kahan in 1965 [46]. We summarize the process as in Algorithm 2.3.

1. Set $\beta_1 = \|t\|_2,\ u_1 = t/\beta_1,\ \hat{v}_1 = M^T u_1,\ \alpha_1 = \|\hat{v}_1\|_2,\ v_1 = \hat{v}_1/\alpha_1$.

2. For $k = 1, 2, \cdots,$

$$\hat{u}_{k+1} = M v_k - \alpha_k u_k,\ \beta_{k+1} = \|\hat{u}_{k+1}\|_2,\ u_{k+1} = \hat{u}_{k+1}/\beta_{k+1}$$

$$\hat{v}_{k+1} = M^T u_{k+1} - \beta_{k+1} v_k,\ \ \alpha_{k+1} = \|\hat{v}_{k+1}\|_2,\ \ v_{k+1} = \hat{v}_{k+1}/\alpha_{k+1}.$$

After k steps, we have

$$MV_k = U_{k+1}N_k,\ \ M^T U_{k+1} = V_k N_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T,$$

where $V_k = \begin{bmatrix} v_1 & v_2 & \cdots & v_k \end{bmatrix}, \quad U_k = \begin{bmatrix} u_1 & u_2 & \cdots & u_k \end{bmatrix},$

$$N_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix}, \quad U_k^T U_k = I, \quad V_k^T V_k = I.$$

Algorithm 2.3 Golub-Kahan Process

Given any matrix M and a vector t, this algorithm constructs orthonormal matrices U_k and V_k to partially transform $[t, M]$ to an upper bidiagonal matrix $[\beta_1 e_1, N_k]$

```
β_1 = ||t||_2, u_1 = t/β_1, v̂_1 = M^T u_1, α_1 = ||v̂_1||_2, v_1 = v̂_1/α_1.
for k = 1, 2, ...n do
    û_{k+1} = M v_k − α_k u_k,
    β_{k+1} = ||û_{k+1}||_2
    u_{k+1} = û_{k+1}/β_{k+1}
    v̂_{k+1} = M^T u_{k+1} − β_{k+1} v_k
    α_{k+1} = ||v̂_{k+1}||_2
    v_{k+1} = v̂_{k+1}/α_{k+1}
end for
```

2.2.5 LSMR

Least Squares Minimal Residual (LSMR) [12] is an iterative method for computing a solution s to either of the following problems.

Linear system

$$Ms = t. \tag{2.8}$$

Least squares problem

$$\min \left\| Ms - t \right\|_2. \tag{2.9}$$

Here $M \in \mathbb{R}^{m\times n}$ is a square or rectangular matrix and $t \in \mathbb{R}^m$ is a vector. LSMR was presented by Fong and Saunders in 2011 [12]. The method LSMR is based on the Golub-Kahan bidiagonalization process [46].

Suppose that k steps of Bidiagonalization have been taken. The k-th approximate solution s_k, such that $s_k = V_k y_k$ where $V_k = \begin{bmatrix} v_1 & v_2 & \cdots & v_k \end{bmatrix}$ and for some y_k, is sought in the Krylov subspace

$$\begin{aligned} \operatorname{span}(V_k) &= \mathcal{K}_k\left(M^T M, M^T t\right) \\ &= \operatorname{span}\left(M^T t, M^T M(M^T t), \cdots, (M^T M)^{k-1}(M^T t)\right). \end{aligned}$$

For LSMR we wish to minimize $\left\|M^T r_k\right\|_2$, where $r_k = t - Ms_k$ is the residual for the approximate solution s_k.

Since

$$M^T r_k = M^T t - M^T M s_k = \beta_1 \alpha_1 v_1 - M^T M V_k y_k$$

and

$$MV_k = U_{k+1} N_k, \;\; M^T U_{k+1} = V_k N_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T,$$

we have

$$\begin{aligned}
M^T r_k &= \beta_1 \alpha_1 v_1 - M^T U_{k+1} N_k y_k \\
&= \beta_1 \alpha_1 v_1 - (V_k N_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T) N_k y_k \\
&= \beta_1 \alpha_1 v_1 - V_{k+1} \begin{bmatrix} N_k^T \\ \alpha_{k+1} e_{k+1}^T \end{bmatrix} N_k y_k \\
&= V_{k+1} \left(\beta_1 \alpha_1 e_1 - \begin{bmatrix} N_k^T N_k \\ \alpha_{k+1} \beta_{k+1} e_k^T \end{bmatrix} y_k \right).
\end{aligned}$$

Since ${V_{k+1}}^T V_{k+1} = I_{k+1}$,

$$\min_{s_k} \left\| M^T r_k \right\|_2 = \min_{y_k} \left\| \bar{\beta}_1 e_1 - \begin{bmatrix} N_k^T N_k \\ \bar{\beta}_{k+1} e_k^T \end{bmatrix} y_k \right\|_2,$$

where $\bar{\beta}_k = \alpha_k \beta_k$ and $\bar{\beta}_1 = \alpha_1 \beta_1$. LSMR uses the double QR decomposition on $N_k^T N_k$ to iteratively minimize $\left\| M^T r_k \right\|_2$.

We may use

$$\left\| M^T r_k \right\|_2 = \left\| M^T (t - s_k) \right\|_2 = |\zeta_{k+1}| < \text{tol} \tag{2.10}$$

as a stopping criterian for both (2.8) and (2.9), where tol is 10^{-8}.

The LSMR algorithm is summarized in Algorithm 2.4. More details can be found in [12].

Algorithm 2.4 LSMR

Given any initial guess $s_0 \in \mathbb{R}^n$, this algorithm computes a minimal residual solution to the linear system $Ms = t$ a least squares problem $\min\left\|Ms - t\right\|_2$.

$\beta_1 = \|t\|_2$, $u_1 = t/\beta_1$, $\hat{v}_1 = M^T u_1$, $\alpha_1 = \|\hat{v}_1\|_2$, $v_1 = \hat{v}_1/\alpha_1$, $\bar{\alpha}_1 = \alpha_1$, $\bar{\zeta}_1 = \alpha_1\beta_1$, $\rho_0 = 1$, $\bar{\rho}_0 = 1$, $\bar{c}_0 = 1$, $\bar{s}_0 = 1$, $h_1 = v_1$, $\bar{h}_0 = 0$, $\bar{x}_0 = 0$

for $k = 1, 2, ...,$ **do**

$\quad \hat{u}_{k+1} = Mv_k - \alpha_k u_k,$

$\quad \beta_{k+1} = \|\hat{u}_{k+1}\|_2$

$\quad u_{k+1} = \hat{u}_{k+1}/\beta_{k+1}$

$\quad \hat{v}_{k+1} = M^T u_{k+1} - \beta_{k+1} v_k$

$\quad \alpha_{k+1} = \|\hat{v}_{k+1}\|_2$

$\quad v_{k+1} = \hat{v}_{k+1}/\alpha_{k+1}$

$\quad \rho_k = \sqrt{\bar{\alpha}^2 + \beta^2{}_{k+1}}$

$\quad c_k = \bar{\alpha}_k/\rho_k$

$\quad s_k = \bar{\beta}_{k+1}/\rho_k$

$\quad \theta_{k+1} = s_k \alpha_{k+1}$

$\quad \bar{\alpha}_{k+1} = c_k \alpha_{k+1}$

$\quad \bar{\theta}_k = \bar{s}_{k-1}\rho_k$

$\quad \bar{\rho} = \sqrt{(\bar{c}_{k-1}\rho_k)^2 + \theta^2{}_{k+1}}$

$\quad \bar{c}_k = \bar{c}_{k-1}\rho_k/\bar{\rho}_k$

$\quad \bar{s}_k = \theta_{k+1}/\bar{\rho}_k$

$\quad \zeta_k = \bar{c}_k \bar{\zeta}_k$

$\quad \bar{\zeta}_{k+1} = -\bar{s}_k \bar{\zeta}_k$

$\quad \bar{h}_k = h_k - (\bar{\theta}_k \rho_k/(\rho_{k-1}\bar{\rho}_{k-1}))\bar{h}_{k-1}$

$x_k = x_{k-1} + (\rho_k/(\zeta_{k-1}\bar{\zeta}_k))\bar{h}_k$

$h_{k+1} = v_{k+1} - (\theta_{k+1}/\rho_k)h_k$

If $|\zeta_{k+1}|$ small enough then stop

end for

2.2.6 Segregated Methods

Since the meaning of the word "segregated" is "separated", segregated methods starts dividing the problem into two subproblems. Namely, it solves the two unknown vectors x and y separately. Segregated methods reduce the whole problem to two smaller ones, i.e., two linear systems of sizes smaller than $n + m$ that is the size of the whole problem. This is why the smaller size system is called a reduced system. These methods solve one of the two unknown vectors first, then perform the back-substitution into the original system to obtain the remaining solution vector.

One of the main representatives of the segregated approach is the Schur complement reduction. In the next subsection, we will briefly review the solution method.

2.2.7 Schur Complement Reduction

Consider the saddle point system (1.2). Assume that A and the Schur complement matrix $S = BA^{-1}B^T$ are nonsingular. Then we know that $\mathcal{A}$ is also nonsingular. Now we can consider the block LU factorization of the saddle point matrix $\mathcal{A}$,

$$\begin{aligned}\mathcal{A} &= \begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} \begin{bmatrix} A & {B_1}^T \\ 0 & B_2A^{-1}{B_1}^T \end{bmatrix}.\end{aligned}$$

Then the system (1.2) becomes,

$$\begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} \begin{bmatrix} A & B_1{}^T \\ 0 & B_2A^{-1}B_1{}^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}. \tag{2.11}$$

Multiply both sides of (2.11) by $\begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix}$ to get

$$\begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} \begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} \begin{bmatrix} A & B_1{}^T \\ 0 & B_2A^{-1}B_1{}^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} \begin{bmatrix} f \\ 0 \end{bmatrix}. \tag{2.12}$$

Since $\begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} \begin{bmatrix} I & 0 \\ B_2A^{-1} & -I \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$, the equation (2.12) becomes

$$\begin{bmatrix} A & B_1{}^T \\ 0 & B_2A^{-1}B^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ B_2A^{-1}f \end{bmatrix}. \tag{2.13}$$

This block upper triangular system can be solved by block substitution. It will lead to the following two reduced systems,

a.

$$B_2A^{-1}B_1{}^Ty = B_2A^{-1}f, \tag{2.14}$$

a reduced system of order m for y involving the (negative) Schur complement $-S = B_2A^{-1}B_1{}^T$. Part solution vector y_* is computed by solving the system (2.14)

b.

$$Ax = f - B_1{}^Ty_*, \tag{2.15}$$

a reduced system of order n for x involving the matrix A. Once the solution has been computed from system (2.14) the unknown x_* can be obtained by solving (2.15). If both approximate solutions x_k and y_k are computed in parallel as a solution of $Ax_k = f - {B_1}^T y_k$ then the Schur complement method can be seen as a coupled method.

These two systems can be solved either directly or iteratively.

Schur complement method has some disadvantages. Some of the major disadvantages are

- A needs to be an invertible matrix to form the Schur complement $S = -B_2 A^{-1} {B_1}^T$.
- Computing the Schur complement matrix may be expensive.
- Numerical instabilities may occur when forming S when A is an ill-conditioned matrix.

2.2.8 Coupled Methods

Unlike segregated methods, coupled methods solve the whole problem together. These methods compute x and y or the approximations of them simultaneously. Coupled methods can be either direct methods based on triangular factorizations of the coefficient matrix $\mathcal{A}$ or iterative methods like Krylov subspace methods applied to the whole system (1.2).

2.2.9 Preconditioning Methods

Preconditioning has been the most active research area for the solution of linear systems especially for the numerical solution of saddle point problems. In the literature, many preconditioners were introduced to improve convergence of the iterative methods for the saddle point problems in the last several years. The main idea of the preconditioning technique is to find an invertible matrix $\mathcal{P}$ such that Krylov subspace method applied to the preconditioned system

$$\mathcal{P}^{-1}\mathcal{A}z = \mathcal{P}^{-1}b \tag{2.16}$$

will converge faster than otherwise. A fast convergence for the Krylov subspace methods depends on the clustered spectrum of $\mathcal{P}^{-1}\mathcal{A}$. Preconditioning usually improves the spectral properties of the system matrix. If the eigenvalues of $\mathcal{P}^{-1}\mathcal{A}$ are clustered enough then the convergence of the iterative method most likely is faster. Unfortunately, finding an effective preconditioner is not easy. A preconditioning matrix must be easy to compute and evaluating its inverse must be cheap.

For an extensive review of preconditioning techniques for the saddle point problems we refer the reader to [1, 48, 50]

We list some of the preconditioning techniques which arise from different applications of saddle point problems:

1. Constraint preconditioning [31, 32],

2. Augmented Lagrangian based approach [54],

3. Multigrid methods [7],

4. Hermitian/skew-Hermitian Splitting [43, 53, 8],

5. Schur complement-based method,
    - Block diagonal preconditioning [51, 34],

- Block triangular preconditionin [15, 52],
- Uzawa preconditioning [44].

CHAPTER 3

SADDLE POINT PROBLEMS WITH FULL RANK (2,1)-BLOCK MATRIX

This chapter is devoted to the solution of the saddle point linear system of the form

$$\mathcal{A}z = b,$$

where $\mathcal{A} \in \mathbb{R}^{(n+m)\times(n+m)}$ is a large and sparse matrix with the 2-by-2 block structure, $b \in \mathbb{R}^{n+m}$ is the right hand side vector and the vector $z \in \mathbb{R}^{n+m}$ is the solution vector. The saddle point system can be written as

$$\begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}, \tag{1.2}$$

where $A \in \mathbb{R}^{n\times n}$ is a large and sparse matrix, $B_1 \in \mathbb{R}^{m\times n}$ and $B_2 \in \mathbb{R}^{m\times n}$ are rectangular matrices with $n \geq m$. The vector $f \in \mathbb{R}^n$ is the right hand side vector and the vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ are the solution vectors of the saddle point system.

In most cases, the fast convergence of the iterative methods depends on the spectral properties of the coefficient matrix $\mathcal{A}$. For example, Krylov subspace methods generally converge rapidly if the coefficient matrix of the linear system has clustered eigenvalues. For some other cases, the inverse matrix of the block matrix A is necessary for the solution of the system (1.2). The purpose of this chapter is to solve a linear system of the form (1.2) with a low computational cost even the spectral properties of $\mathcal{A}$ is not nice or the inverse of A is not available.

In this chapter, we are interested in the solution of the saddle point systems when the (2,1)-block, $B_2 \in \mathbb{R}^{m\times n}$ is a full row rank matrix, i.e., $\text{rank}(B_2) = m$ and the number of rows in B_2 is rather small compared to the number of columns in B_2. These two conditions are the only conditions for the method we present in this chapter.

3.1 General Theory

In this section, we will give the theoretical explanation of the solution method when the (2,1)-block, $B_2 \in \mathbb{R}^{m\times n}$ is of full rank and m is relatively small to n.

Theorem 3.1.1. *(Rank-Nullity Theorem). Let B_2 be an $m \times n$ matrix. Then*

$$\text{rank}(B_2) + \text{null}(B_2) = n. \tag{3.1}$$

Proof. [55] □

Theorem 3.1.2. *An $m \times m$ square matrix $B_2{B_2}^T$ is invertible, if $B_2 \in \mathbb{R}^{m\times n}$ is a full row rank matrix (i.e.,* $\text{rank}(B_2) = m$*).*

Proof. Suppose $B_2 \in \mathbb{R}^{m\times n}$ is a full row rank matrix. We want to prove that $B_2{B_2}^T$ is invertible. It suffices to show that if $B_2{B_2}^T x = 0$ for some vector x, then $x = 0$. Since $\text{rank}(B_2) = m$, then the rank of ${B_2}^T$ is also m, i.e., $\text{rank}({B_2}^T) = m$. By Theorem 3.1.1

$$\begin{aligned} \text{null}({B_2}^T) &= m - \text{rank}({B_2}^T) \\ &= m - m \\ &= 0. \end{aligned} \tag{3.2}$$

If $B_2 {B_2}^T x = 0$, then

$$\begin{aligned} 0 = x^T B_2 {B_2}^T x &= ({B_2}^T x)^T ({B_2}^T x) \\ &= \langle {B_2}^T x, {B_2}^T x \rangle \\ &= \left\| {B_2}^T x \right\|. \end{aligned}$$

If $\left\| {B_2}^T x \right\| = 0$, then ${B_2}^T x = 0$. We also know that $\text{null}({B_2}^T) = 0$ by (3.2). Then x is a zero vector. Hence $B_2 {B_2}^T \in \mathbb{R}^{m \times m}$ is an invertible matrix. □

The previous theorem will allow us to construct the projection matrix that we need for the solution of the saddle point system.

Theorem 3.1.3. *A vector $x \in \mathbb{R}^n$ is in the null space of $B_2 \in \mathbb{R}^{m \times n}$ if and only if it can be written as*

$$x = Qz, \tag{3.3}$$

where $Q = I - {B_2}^T (B_2 {B_2}^T)^{-1} B_2 \in \mathbb{R}^{n \times n}$ and $z \in \mathbb{R}^n$.

Proof. $(\Rightarrow)$ Assume that $x \in \mathbb{R}^n$ is in the null space of $B_2 \in \mathbb{R}^{m \times n}$. Then

$$B_2 x = 0. \tag{3.4}$$

Also every $x \in \mathbb{R}^n$ can be written as

$$\begin{aligned} x &= x - {B_2}^T (B_2 {B_2}^T)^{-1} B_2 x + {B_2}^T (B_2 {B_2}^T)^{-1} B_2 x \\ &= [I - {B_2}^T (B_2 {B_2}^T)^{-1} B_2] x + {B_2}^T (B_2 {B_2}^T)^{-1} B_2 x. \end{aligned}$$

By (3.4) we have

$$\begin{aligned} x &= [I - {B_2}^T (B_2 {B_2}^T)^{-1} B_2] x \\ &= Qx, \end{aligned}$$

where $Q = I - {B_2}^T(B_2{B_2}^T)^{-1}B_2 \in \mathbb{R}^{n\times n}$.

($\Leftarrow$) Suppose that $x \in \mathbb{R}^n$ takes the form

$$x = Qz,$$

where $Q = I - {B_2}^T(B_2{B_2}^T)^{-1}B_2 \in \mathbb{R}^{n\times n}$ and $z \in \mathbb{R}^n$. Then

$$\begin{aligned}
B_2x &= B_2Qz \\
&= B_2[I - {B_2}^T(B_2{B_2}^T)^{-1}B_2]z \\
&= [B_2 - B_2{B_2}^T(B_2{B_2}^T)^{-1}B_2]z \\
&= (B_2 - B_2)z \\
&= 0.
\end{aligned}$$

Hence x is in the null space of B_2. This completes the proof of the theorem. $\square$

Now we turn to the problem itself.

The saddle point system (1.2) can be written as

$$Ax + {B_1}^Ty = f,$$
$$B_2x = 0.$$

Since x is in the null space of B_2, by Theorem 3.1.3 x takes the form

$$x = Qz.$$

Once we substitute $x = Qz$ into the saddle point problem, we obtain

$$\begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} Qz \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}.$$

Since $B_2Qz = 0$, we only need to write the first block equation,

$$AQz + {B_1}^T y = f,$$

which can be expressed by

$$\begin{bmatrix} AQ & {B_1}^T \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} = f.$$

Therefore the saddle point problem is turned into

$$\min_{z,y} \left\| \begin{bmatrix} AQ & {B_1}^T \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} - f \right\|_2 , \tag{3.5}$$

which is a least squares problem whose coefficient matrix is in $\mathbb{R}^{n \times (n+m)}$. By solving the least squares problem the solution vectors z and y will be obtained. Once the solution z is obtained, x can be computed from the equation $x = Qz$.

Next we solve the underdetermined least squares problem (3.5) by using one of the Krylov subspace methods, i.e., LSMR. LSMR is an iterative solution technique for sparse least squares problems. It is based on the Golub-Kahan bidiagonalization [46]. The Golub-Kahan process is a recursive procedure which transforms $\begin{bmatrix} f & [AQ \quad {B_1}^T] \end{bmatrix}$ to upper-bidiagonal form $\begin{bmatrix} \beta_1 e_1 & F_k \end{bmatrix}$ by constructing orthogonal matrices U and V as follows:

$$U^T \begin{bmatrix} f & [AQ \quad {B_1}^T] \end{bmatrix} \begin{bmatrix} 1 & \\ & V \end{bmatrix} = \begin{bmatrix} * & * & & \\ & * & \ddots & \\ & & \ddots & * \\ & & & * \end{bmatrix}.$$

It is equivalent to the following

$$\begin{bmatrix} f & [AQ \quad {B_1}^T]V \end{bmatrix} = U \begin{bmatrix} \beta_1 e_1 & F \end{bmatrix},$$

where F is a lower bidiagonal matrix. In the Golub-Kahan procedure we need to calculate the multiplication of the coefficient matrix in the problem and a vector and the multiplication of the transpose of the coefficient matrix and a vector. Namely, $\begin{bmatrix} AQ & {B_1}^T \end{bmatrix} v$ and $\begin{bmatrix} QA^T \\ B_1 \end{bmatrix} u$ are calculated for some vectors u and v.

In order to solve the problem efficiently, we need to perform the following two actions efficiently:

1. $w \leftarrow \begin{bmatrix} QA^T \\ B_1 \end{bmatrix} u$,

$$w_0 = A^T u$$

$$w = w_0 - B_2^T[(B_2 {B_2}^T)^{-1}(B_2 w_0)]$$

$$w = \begin{bmatrix} w \\ B_1 u \end{bmatrix}.$$

2. $w \leftarrow \begin{bmatrix} AQ & {B_1}^T \end{bmatrix} v$. Let

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix},$$

where $v_1 \in \mathbb{R}^n$ and $v_2 \in \mathbb{R}^m$. Then

$$\begin{aligned}\begin{bmatrix} AQ & {B_1}^T \end{bmatrix} v &= \begin{bmatrix} AQ & {B_1}^T \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\ &= AQv_1 + B_1^T v_2\end{aligned}$$

$$w = Qv_1$$

$$w = w - {B_2}^T[(B_2{B_2}^T)^{-1}(B_2 v_1)]$$

$$w = Aw + {B_1}^T v_2,$$

Algorithm 3.1 Full Rank Saddle Point Problem (FRSPP)

Given any initial guess $\begin{bmatrix} z_0 \\ y_0 \end{bmatrix} \in \mathbb{R}^{n+m}$, this algorithm computes a minimal residual solution to the least squares system $\min \left\| \begin{bmatrix} AQ & {B_1}^T \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} - f \right\|_2$ and computes the solution vector $x \in \mathbb{R}^n$ such that $x = Qz$ for the saddle point system

$$\begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}.$$

1: Compute $Q = I - {B_2}^T(B_2{B_2}^T)^{-1}B_2$ in form (i.e., not actually formulate Q explicitly)

2: Solve $\min \left\| \begin{bmatrix} AQ & {B_1}^T \end{bmatrix} \begin{bmatrix} z \\ y \end{bmatrix} - f \right\|_2$ by LSMR

3: $x \leftarrow Qz$

CHAPTER 4

SADDLE POINT PROBLEMS WITH RANK-DEFICIENT (2,1)-BLOCK MATRIX

In the previous chapter, we presented a solution method to solve the saddle point problem of the form (1.2) when the (2,1)-block matrix B_2 is of full rank. In this chapter, our focus is on the solution of the saddle point problem (1.2) when B_2 is a rank-deficient matrix. For most saddle point problems, B_2 is a full rank matrix but not all the time. Our main task here is to solve the system (1.2) with rank-deficient B_2 by using a projection matrix. Since B_2 is a rank-deficient matrix we are no longer able to use the same projection matrix that we used in Chapter 3. The main idea here is to construct a new projection matrix by using maximal linearly independent rows of B_2 and solve the system.

4.1 General Theory

Suppose that B_2 is not a full rank matrix and its rows can be permuted into the following partition

$$PB_2 = \begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix} \in \mathbb{R}^{m \times n}, \tag{4.1}$$

where $P \in \mathbb{R}^{m \times m}$ is a permutation matrix, $B_{21} \in \mathbb{R}^{l \times n}$ is a full rank matrix and $\mathrm{rank}(B_2) = l \leq m$. In particular, the rows of B_{21} are linearly independent.

Theorem 4.1.1. *Suppose that B_2 is not a full rank matrix. Permute B_2 as in (4.1), where $P \in \mathbb{R}^{m \times m}$ is a permutation matrix, $B_{21} \in \mathbb{R}^{l \times n}$ has full row rank such that* $\mathrm{rank}(B_2)$=$\mathrm{rank}(B_{21}) = l < m$ *and $B_{22} \in \mathbb{R}^{(m-l) \times n}$. Then every row of B_{22} can be* *written as a linear combination of B_{21}'s rows. This implies that there exists a matrix $C \in \mathbb{R}^{(m-l)\times l}$ such that $B_{22} = CB_{21}$.*

Proof. Let

$$\begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix} = \left[\begin{array}{c} [\quad b_1 \quad] \\ [\quad b_2 \quad] \\ \vdots \\ [\quad b_l \quad] \\ \hline [\quad b_{l+1} \quad] \\ [\quad \vdots \quad] \\ [\quad b_m \quad] \end{array}\right]$$

where $\{b_1, b_2, ..., b_l\}$ are the the row vectors of B_{21} and $\{b_{l+1}, b_{l+2}, ..., b_m\}$ are the row vectors of B_{22}. Since $\{b_{l+1}, b_{l+2}, ..., b_m\}$ are linearly dependent on the rows of B_{21} every row vector of B_{22} can be written as a linear combination of the vectors $\{b_1, b_2, ..., b_l\}$. Then there exist scalars $c_{l+1,1}, c_{l+1,2}, ..., c_{l+1,l}$, $c_{l+2,1}, c_{l+2,2}, ..., c_{l+2,l}$,..., $c_{m,1}, c_{m,2}, ..., c_{m,l}$ such that

$$\begin{aligned} b_{l+1} &= c_{l+1,1}b_1 + c_{l+1,2}b_2 + ... + c_{l+1,l}b_l, \\ b_{l+2} &= c_{l+2,1}b_1 + c_{l+2,2}b_2 + ... + c_{l+2,l}b_l, \\ &\vdots \\ &\vdots \\ b_m &= c_{m,1}b_1 + c_{m,2}b_2 + ... + c_{m,l}b_l. \end{aligned}$$

Equivalently,

$$B_{22} = \begin{bmatrix} c_{l+1,1} & c_{l+1,2} & \dots & c_{l+1,l} \\ \vdots & \vdots & & \vdots \\ c_{m,1} & c_{m,2} & \dots & c_{m,l} \end{bmatrix} \begin{bmatrix} [& b_1 &] \\ [& b_2 &] \\ & \vdots & \\ [& b_l &] \end{bmatrix}$$

$$= CB_{21}.$$

Hence, we have proved the claim. □

Theorem 4.1.2. *An $l \times l$ square matrix $B_{21}{B_{21}}^T$ is invertible, if $B_{21} \in \mathbb{R}^{l\times n}$ is a full row rank matrix (i.e., rank(B_{21}) $= l$).*

Proof. Similar to the proof of Theorem 3.1.2. □

Theorem 4.1.3. *A vector $x \in \mathbb{R}^n$ is in the null space of $B_2 \in \mathbb{R}^{m\times n}$ if and only if it can be written as*

$$x = \hat{Q}\hat{z} \tag{4.2}$$

where $\hat{Q} = I - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21} \in \mathbb{R}^{n\times n}$ and $\hat{z} \in \mathbb{R}^n$.

Proof. ($\Rightarrow$) Assume that $x \in \mathbb{R}^n$ is in the null space of $B_2 \in \mathbb{R}^{m\times n}$. Then

$$B_2 x = 0. \tag{4.3}$$

Multiply both sides of (4.3) by permutation matrix $P \in \mathbb{R}^{m\times m}$. Then we have

$$PB_2 x = 0. \tag{4.4}$$

By (4.1)

$$0 = PB_2 x \;\; = \;\; \begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix} x \tag{4.5}$$

$$= \begin{bmatrix} B_{21}x \\ B_{22}x \end{bmatrix}. \tag{4.6}$$

Hence

$$B_{21}x = 0 \quad \text{and} \quad B_{22}x = 0. \tag{4.7}$$

Also every $x \in \mathbb{R}^n$ can be written as

$$\begin{aligned} x &= x - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21}x + {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21}x \\ &= (I - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21})x + {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21}x. \end{aligned}$$

By (4.7) we have

$$\begin{aligned} x &= (I - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21})x \\ &= \hat{Q}x, \end{aligned}$$

where $\hat{Q} = I - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21} \in \mathbb{R}^{n\times n}$.

($\Leftarrow$) Suppose that $x \in \mathbb{R}^n$ takes the form

$$x = \hat{Q}\hat{z},$$

where $\hat{Q} = I - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21} \in \mathbb{R}^{n\times n}$ and $\hat{z} \in \mathbb{R}^n$. We also know that $B_{22} = CB_{21}$ for some $C \in \mathbb{R}^{(m-l)\times l}$ by Theorem 4.1.1. Then

$$PB_2\hat{Q}z = \begin{bmatrix} B_{21} \\ B_{22} \end{bmatrix} \hat{Q}z \tag{4.8}$$

$$= \begin{bmatrix} B_{21} \\ CB_{21} \end{bmatrix} \begin{bmatrix} I - {B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21} \end{bmatrix} z \tag{4.9}$$

$$= \begin{bmatrix} B_{21}I - B_{21}{B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21} \\ CB_{21}I - CB_{21}{B_{21}}^T(B_{21}{B_{21}}^T)^{-1}B_{21} \end{bmatrix} z \tag{4.10}$$

$$= \begin{bmatrix} B_{21} - (B_{21}{B_{21}}^T)(B_{21}{B_{21}}^T)^{-1}B_{21} \\ CB_{21} - C(B_{21}{B_{21}}^T)(B_{21}{B_{21}}^T)^{-1}B_{21} \end{bmatrix} z \tag{4.11}$$

$$= \begin{bmatrix} B_{21} - B_{21} \\ CB_{21} - CB_{21} \end{bmatrix} z \tag{4.12}$$

$$= 0z \tag{4.13}$$

$$= 0. \tag{4.14}$$

Hence x is in the null space of B_2. This completes the proof of the theorem. $\square$

Remember that the saddle point system (1.2) can be written as

$$Ax + {B_1}^T y = f,$$

$$B_2 x = 0.$$

By Theorem 4.1.3 the solution vector $x \in \mathbb{R}^n$ can be written as

$$x = \hat{Q}\hat{z}. \tag{4.15}$$

Substituting $x = \hat{Q}\hat{z}$ in $Ax + {B_1}^T y = f$, we obtain

$$A\hat{Q}\hat{z} + {B_1}^T y = f,$$

which can be expressed by

$$\begin{bmatrix} A\hat{Q} & {B_1}^T \end{bmatrix} \begin{bmatrix} \hat{z} \\ y \end{bmatrix} = f.$$

Therefore the saddle point problem is turned into

$$\min_{\hat{z},y} \left\| \begin{bmatrix} A\hat{Q} & {B_1}^T \end{bmatrix} \begin{bmatrix} \hat{z} \\ y \end{bmatrix} - f \right\|_2 , \tag{4.16}$$

which is a least squares problem whose coefficient matrix is in $\mathbb{R}^{n\times(n+m)}$. By solving the least squares problem the solution vectors $\hat{z}$ and y will be obtained. Once the solution $\hat{z}$ is obtained, x can be computed from the equation $x = \hat{Q}\hat{z}$.

Algorithm 4.1 Rank-Deficient Saddle Point Problem (RDSPP)

Given any initial solution $\begin{bmatrix} \hat{z}_0 \\ y_0 \end{bmatrix} \in \mathbb{R}^{n+m}$, this algorithm computes a minimal residual solution to the least squares system $\min \left\| \begin{bmatrix} A\hat{Q} & {B_1}^T \end{bmatrix} \begin{bmatrix} \hat{z} \\ y \end{bmatrix} - f \right\|_2$ and computes the solution vector $x \in \mathbb{R}^n$ such that $x = \hat{Q}\hat{z}$ for the saddle point system

$$\begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}$$

1: Find B_{21}

2: Compute $\hat{Q} = I - {B_{2_1}}^T (B_{21} {B_{21}}^T)^{-1} B_{21}$ in form (i.e., not actually formulate Q explicitly)

3: Solve $\min \left\| \begin{bmatrix} A\hat{Q} & {B_1}^T \end{bmatrix} \begin{bmatrix} \hat{z} \\ y \end{bmatrix} - f \right\|_2$ by LSMR

4: $x \leftarrow \hat{Q}\hat{z}$

4.2 Finding B_{21}

In this section we briefly explain how to find linearly independent rows of (2,1)-block matrix B_2. For a matrix B_2 with rank l , the first l rows of B_2 may not be linearly independent. To find out the linearly independent rows of B_2, we use the QR factorization of ${B_2}^T$ with column pivoting [5].

The QR factorization of ${B_2}^T$ with column pivoting computes the factorization

$$
\begin{aligned}
{B_2}^T P_\pi &= Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix} && (4.17) \\
&= \begin{bmatrix} \hat{b}_1 \ldots \hat{b}_m \end{bmatrix}, && (4.18)
\end{aligned}
$$

where $Q \in \mathbb{R}^{n \times n}$ is an orthogonal matrix, $R_{11} \in \mathbb{R}^{l \times l}$ is a nonsingular and upper triangular matrix and, $P_\pi \in \mathbb{R}^{m \times m}$ is a permutation matrix.

Suppose that for some k, Householder matrices $H_1, \ldots, H_{k-1}$ and permutation matrices $P_{\pi_1}, \ldots, P_{\pi_{k-1}}$ are computed such that

$$
\begin{aligned}
(H_{k-1} \ldots H_1){B_2}^T(P_{\pi_1} \ldots P_{\pi_{k-1}}) &= R^{(k-1)} && (4.19) \\
&= \begin{bmatrix} {R_{11}}^{(k-1)} & {R_{12}}^{(k-1)} \\ 0 & {R_{22}}^{(k-1)} \end{bmatrix}, && (4.20)
\end{aligned}
$$

where ${R_{11}}^{(k-1)}$ is a nonsingular and upper triangular matrix. Suppose that

$$
{R_{22}}^{(k-1)} = \begin{bmatrix} {u_k}^{(k-1)}, \ldots, {u_m}^{(k-1)} \end{bmatrix} \tag{4.21}
$$

is a column partitioning and let $i \geq k$ be the smallest index such that

$$
\left\| {u_i}^{(k-1)} \right\|_2 = \max \left\{ \left\| {u_k}^{(k-1)} \right\|_2, \ldots, \left\| {u_m}^{(k-1)} \right\|_2 . \right\} \tag{4.22}
$$

If $\left\| u_i{}^{(k-1)} \right\|_2 = 0$, we should stop the calculation. If $\left\| u_i{}^{(k-1)} \right\|_2 > 0$, we determine the permutation matrix $P_\pi \in \mathbb{R}^{m\times m}$ by swapping the p-th and k-th columns and determine Householder matrix H_k such that $R^{(k)} = H_k R^{(k-1)} P_{\pi_k}$ then $R^{(k)}(k+1:n,k) = 0$. Once we finish calculating the k-th step, we check if $\frac{|u_{kk}|}{|u_{11}|}$ <tol, where tol(tolerance) is 10^{-12}. If $\frac{|u_{kk}|}{|u_{11}|}$ >tol, $\hat{b}_k$ is a column of ${B_{21}}^T$. Then the matrix ${B_{21}}^T$ will be

$$ {B_{21}}^T = \begin{bmatrix} \hat{b}_1 & \hat{b}_2 & \ldots \hat{b}_{k-1} & \hat{b}_k \end{bmatrix}, \tag{4.23} $$

where $\hat{b}_1$, $\hat{b}_2$, ... ,$\hat{b}_{k-1}$, $\hat{b}_k$ are linearly independent columns of ${B_{21}}^T$. Then the matrix B_{21} will be

$$ B_{21} = \begin{bmatrix} [& \hat{b}_1^T &] \\ [& \hat{b}_2^T &] \\ & \vdots & \\ [& \hat{b}_{k-1}^T &] \\ [& \hat{b}_k^T &] \end{bmatrix}. \tag{4.24} $$

CHAPTER 5

NUMERICAL RESULTS

In this chapter, we show some numerical results that illustrate the performances of the projected method for saddle point problems. The numerical experiments show the comparison of the convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using projection matrices for both full rank B_2 and rank-deficient B_2.

The linear system has the form

$$\begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix},$$

where $A \in \mathbb{R}^{n \times n}$, $B_1, B_2 \in \mathbb{R}^{m \times n}$, $f \in \mathbb{R}^n$ with $n \geq m$. Initial guess is taken to be the zero vector for all tests.

All numerical results shown in this chapter were run using Matlab version R2017b (9.3.0). We have taken all the testing matrices from SuiteSparse matrix collection, formerly the University of Florida sparse matrix collection [47]. These matrices with their generic properties are shown in Table 5.1 and Table 5.2. These two tables give, for each matrix, m the number of rows in B_2, n the number of columns in B_2, number of nonzero entries and their sources.

First, we will discuss the results for full rank B_2 and then the results of rank-deficient B_2.

5.1 Full Rank B_2

In this section, we show some numerical results that illustrate the convergence of GMRES and LSMR applied to the least squares problem after using the projection matrix for the full row rank case. We report the relative residual

$$\frac{\left\|b - \mathcal{A}z\right\|_2}{\left\|b\right\|_2}$$

for Algorithm 3.1, where the system is solved after using projection matrix and for Algorithm 2.2 where the original system $\mathcal{A}z = b$ is solved. Some of the examples in this section have singular coefficient matrices. We check the consistency of the system $\mathcal{A}z = b$ by calculating the rank of the coefficient matrix, $\mathcal{A}$ and the rank of the augmented matrix $[\mathcal{A}, b]$. In all our examples, rank($\mathcal{A}$)=rank($[\mathcal{A}, b]$).

The example matrices in Table 5.1 have the form

$$\mathcal{A} = \begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix}$$

and n represents the number of columns in B_2 and m is the number of the rows in B_2. m is relatively small to n for each example. The size of each testing matrix is $(n+m) \times (n+m)$. The (2,1)-block matrix B_2 for each testing matrix has full row rank.

Matrix	n	m	nonzero	application
`lshape1`	353	98	3807	statistics
`maxwell3`	1504	48	8474	electromagnetic
`maxwell4`	6080	198	34698	electromagnetic
`lshape4`	7544	238	44652	statistics
`navierstokesN16`	1472	51	36352	incompressible flow
`stokesN8`	352	27	3256	computational fluid dynamics
`dynamicSoaringProblem_1`	363	284	5367	optimal control
`ncvxqp1`	7110	73	44398	optimization problem

Table 5.1. Testing Matrices with Full Rank B_2

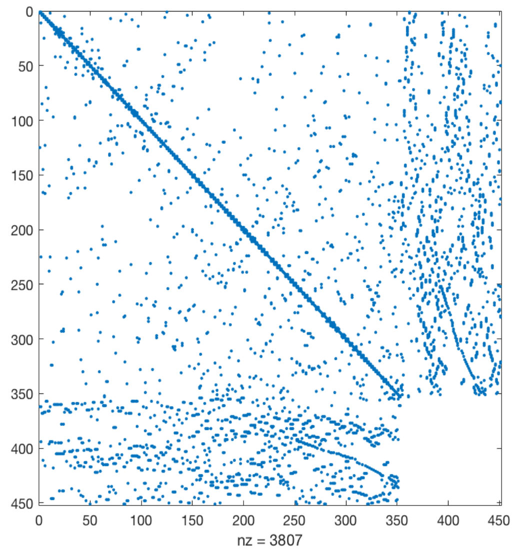

Figure 5.1. Sparsity pattern of $\mathcal{A}$ formed by `lshape1`. Size: 451×451, n=353, m=98, number of nonzero entries=3807, condition number=6.3461e+03, rank(B_2)=98.

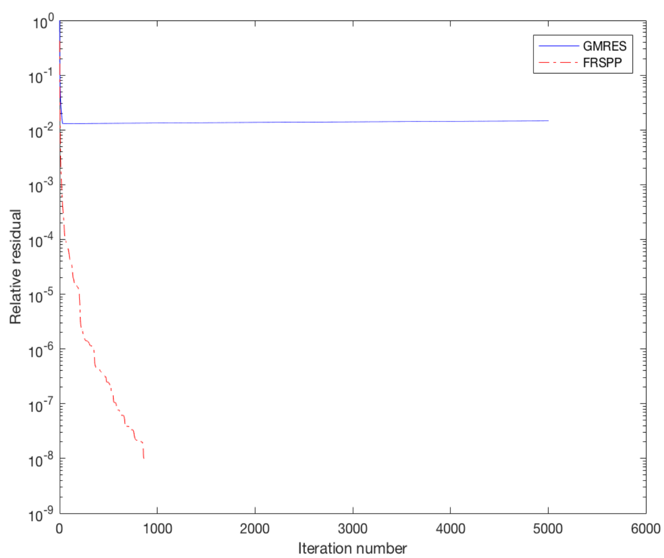

Figure 5.2. Relative residual *vs*. iteration number for `lshape1`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

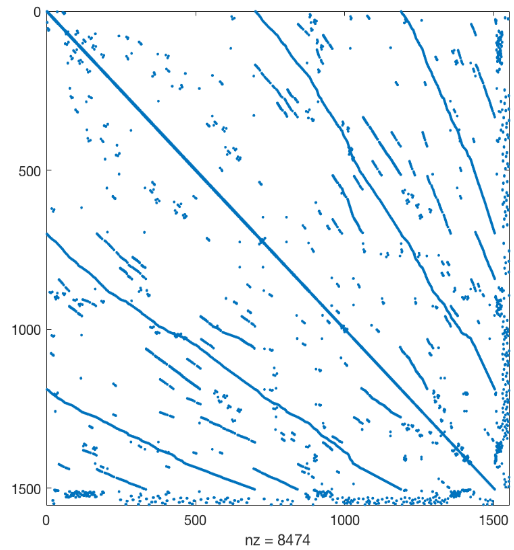

Figure 5.3. Sparsity pattern of $\mathcal{A}$ formed by `maxwell3`. Size: 1552×1552, n=1504, m=48, number of nonzero entries=8474, condition number=2.9829e+21, rank(B_2)=48.

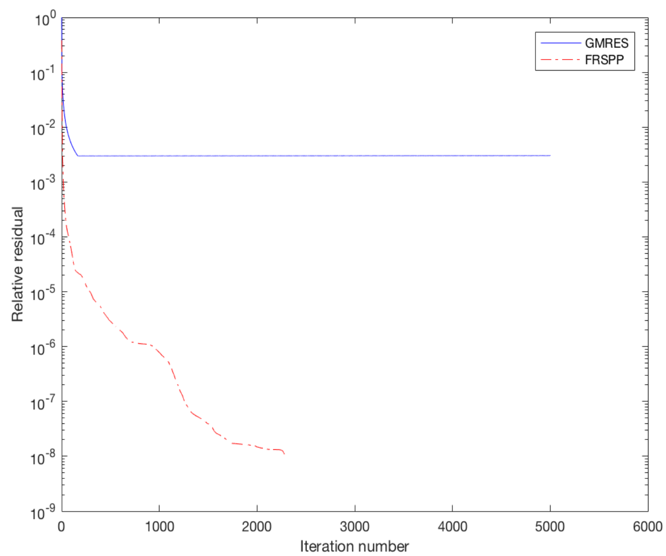

Figure 5.4. Relative residual *vs.* iteration number for `maxwell3`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

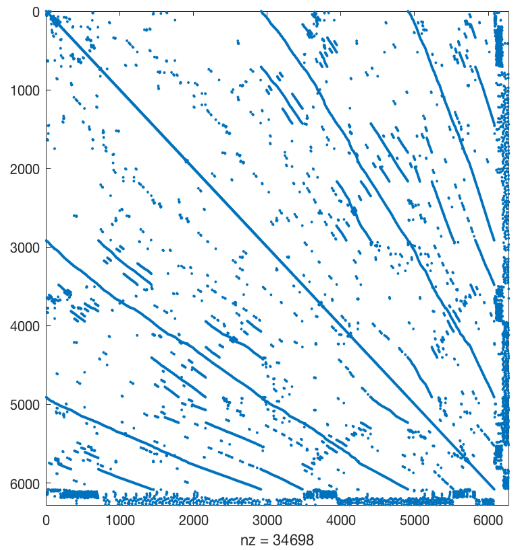

Figure 5.5. Sparsity pattern of $\mathcal{A}$ formed by `maxwell4`. Size: 6278×6278, n=6080, m=198, number of nonzero entries=34698, condition number=Inf, rank(B_2)=198.

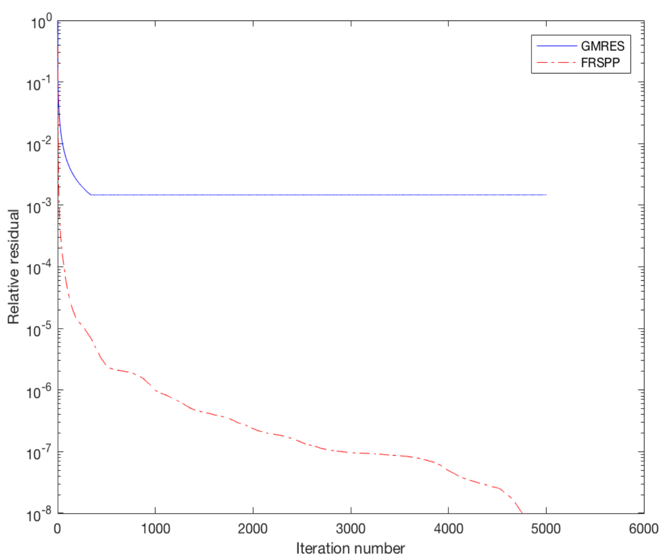

Figure 5.6. Relative residual *vs.* iteration number for `maxwell4`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

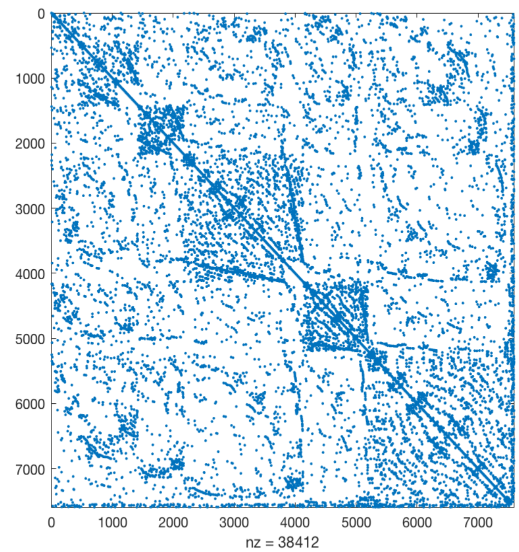

Figure 5.7. Sparsity pattern of $\mathcal{A}$ formed by `lshape4`. Size: 7782×7782, n=7544, m=238, number of nonzero entries=44652, condition number=Inf, rank(B_2)=238.

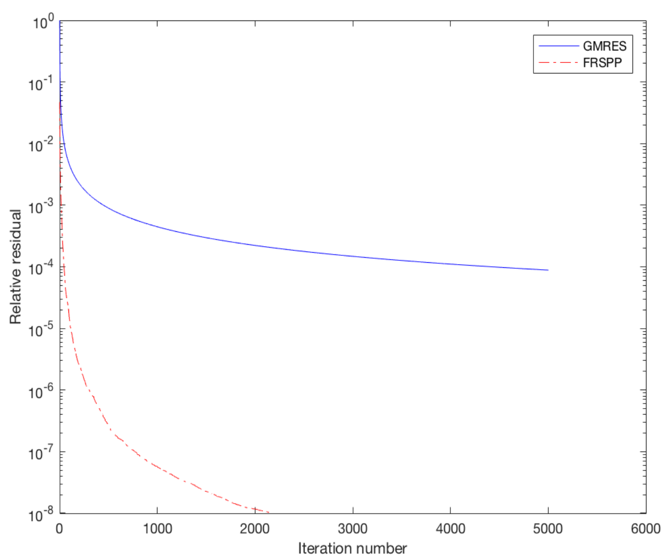

Figure 5.8. Relative residual *vs.* iteration number for `lshape4`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

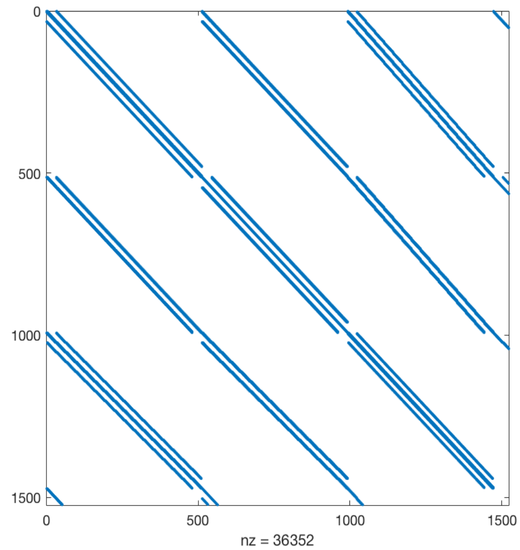

Figure 5.9. Sparsity pattern of $\mathcal{A}$ formed by `navierstokesN16`. Size: 1523×1523, n=1472, m=51, number of nonzero entries=36352, condition number=7.4027e+04, rank(B_2)=51.

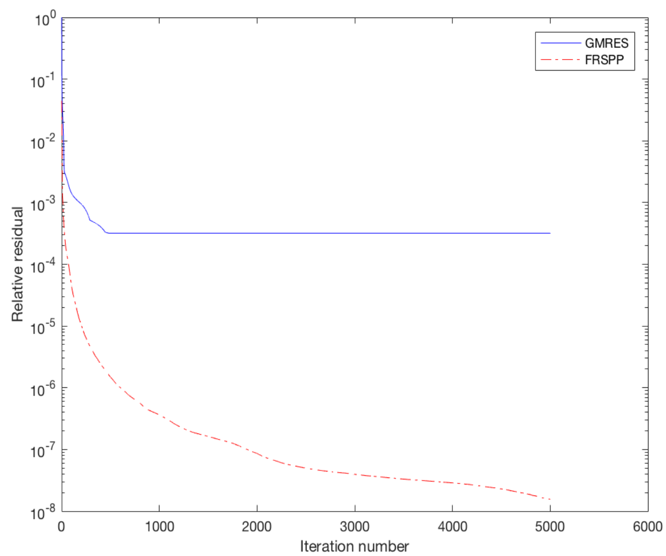

Figure 5.10. Relative residual *vs*. iteration number for `navierstokesN16`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

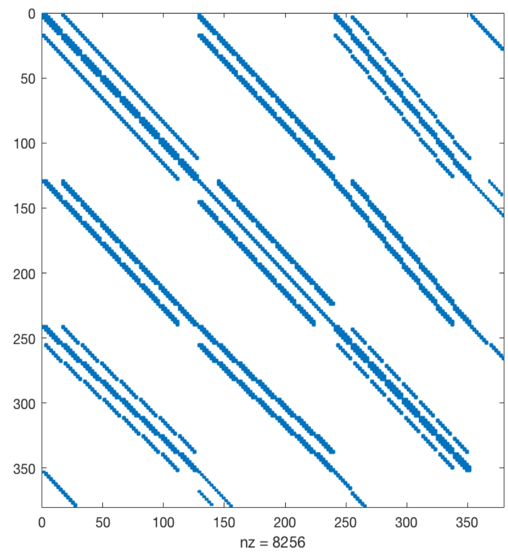

Figure 5.11. Sparsity pattern of $\mathcal{A}$ formed by `stokesN8`. Size: 379×379, n=352, m=27, number of nonzero entries=3256, condition number=7.5977e+04, rank(B_2)=27.

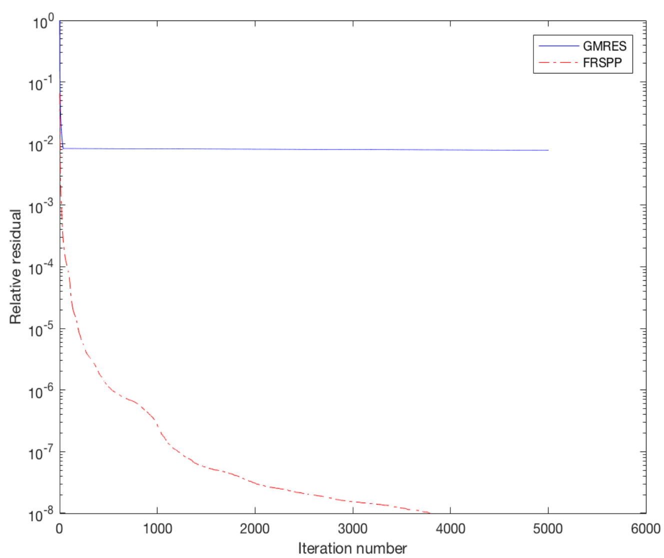

Figure 5.12. Relative residual *vs.* iteration number for `stokesN8`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

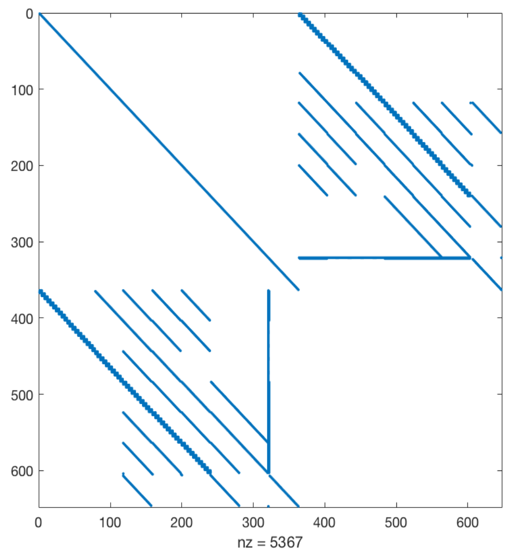

Figure 5.13. Sparsity pattern of $\mathcal{A}$ formed by `dynamicSoaringProblem_1`. Size: 647×647, n=363, m=284, number of nonzero entries=5367, condition number=3.0853e+05, rank(B_2)=284.

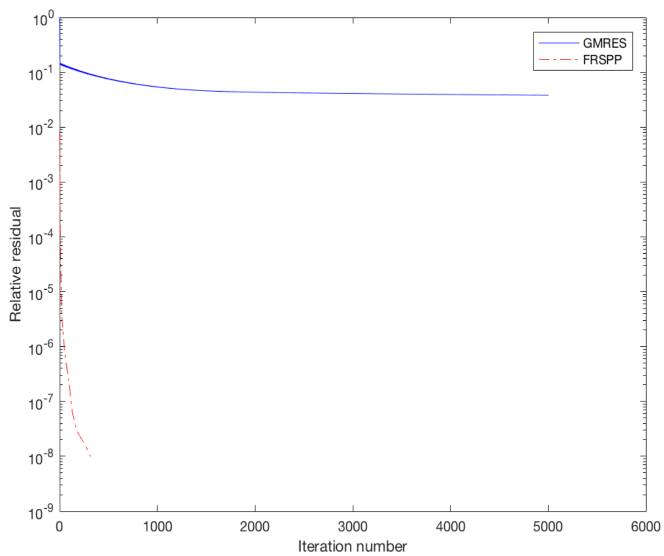

Figure 5.14. Relative residual *vs.* iteration number for `dynamicSoaringProblem_1`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

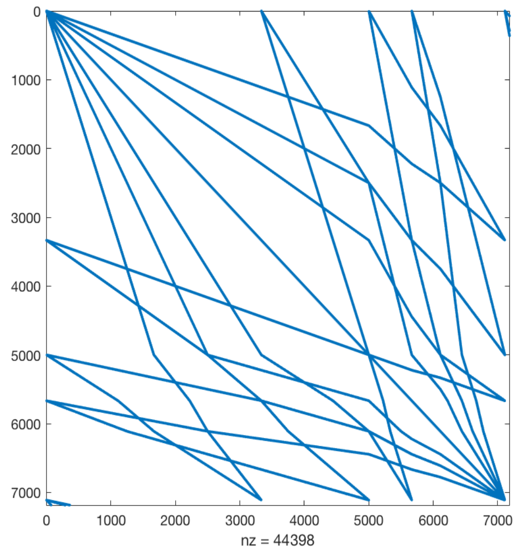

Figure 5.15. Sparsity pattern of $\mathcal{A}$ formed by `ncvxqp1`. Size: 7183×7183, n=7110, m=73, number of nonzero entries=44398, condition number=5.5473e+22, rank(B_2)=73.

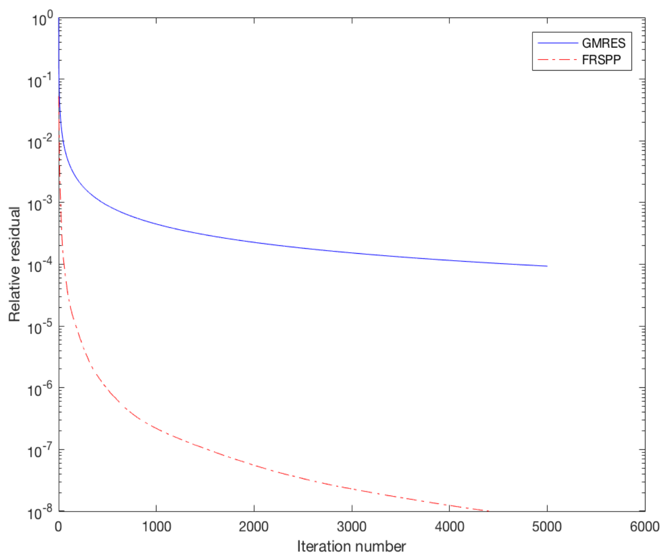

Figure 5.16. Relative residual *vs*. iteration number for `ncvxqp1`. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix for full rank B_2.

5.2 Rank-Deficient B_2

In this section, we show some numerical results that illustrate the convergence of GMRES and LSMR applied to the least squares problem after using the projection matrix for the rank-deficient case. We report the relative residual

$$\frac{\left\|b - \mathcal{A}z\right\|_2}{\left\|b\right\|_2}$$

for the method, LSMR, where the system is solved after using projection matrix and for the method, GMRES, where the original system $\mathcal{A}z = b$ is solved. Since the system is rank-deficient we check if it has a solution. We check the consistency of the system $\mathcal{A}z = b$ by calculating the rank of the coefficient matrix, $\mathcal{A}$ and the rank of the augmented matrix $[\mathcal{A}\ b]$. In all our examples, rank($\mathcal{A}$)=rank($[\mathcal{A}\ b]$).

The example matrices in Table 5.2 represent the (2,1)-block matrix B_2. In this table n represents the number of columns in B_2 and m is the number of the rows in B_2. We assign random sparse matrices for $A \in \mathbb{R}^{n\times n}$ and $B_1 \in \mathbb{R}^{n\times m}$ to form the saddle point matrix

$$\mathcal{A} = \begin{bmatrix} A & {B_1}^T \\ B_2 & 0 \end{bmatrix}.$$

Here, m is relatively small to n for each example. The size of each testing matrix $\mathcal{A}$ is $(n+m)\times(n+m)$. The (2,1)-block matrix B_2 for each testing matrix is rank-deficient. The rank of each matrix B_2 is given in the sparsity pattern of $\mathcal{A}$ in each figure.

Matrix	n	m	nonzero	application
`Maragal_1`	31	14	234	least squares problem
`Maragal_2`	555	350	4582	least squares problem
`Maragal_3`	1690	860	20130	least squares problem
`GL6_D_6`	469	201	2642	combinatorial problem
`GL7d11`	1019	60	1678	combinatorial problem
`GL7d26`	2798	305	8273	combinatorial problem

Table 5.2. Rank-Deficient Testing Matrices B_2

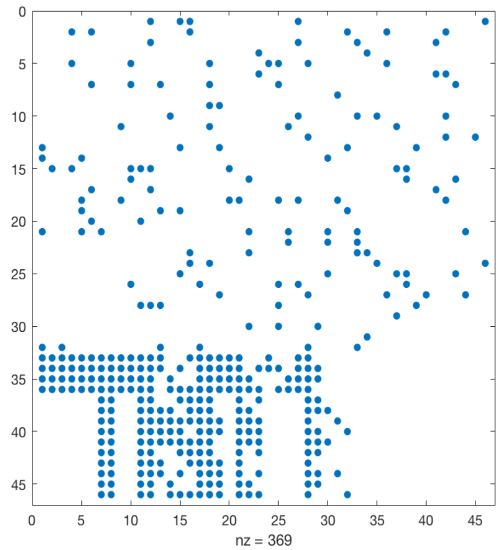

Figure 5.17. Sparsity pattern of $\mathcal{A}$ formed by random matrix $A \in \mathbb{R}^{32\times 32}$, random matrix $B_1 \in \mathbb{R}^{32\times 14}$ and `Maragal_1`$=B_2 \in \mathbb{R}^{14\times 32}$. Size: 46×46, n=32, m=14, number of nonzero entries=234, condition number= Inf, rank(B_2)=10.

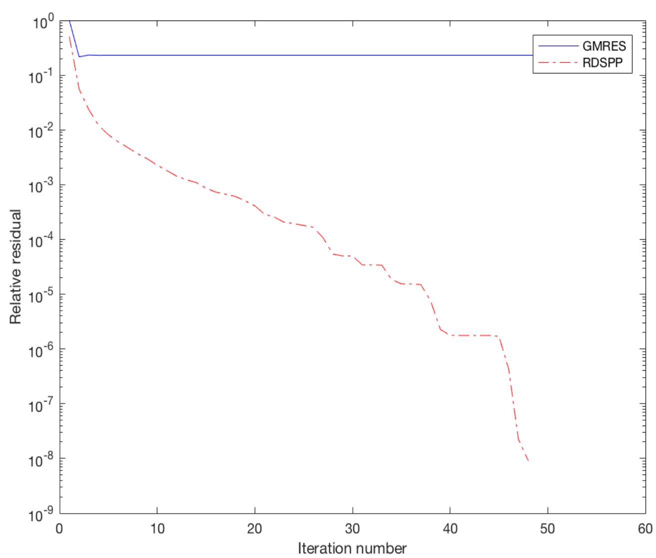

Figure 5.18. Relative residual *vs.* iteration number for $\mathcal{A}$ formed by random A, B_1 and `Maragal_1`$=B_2$. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix to the system with rank-deficient B_2.

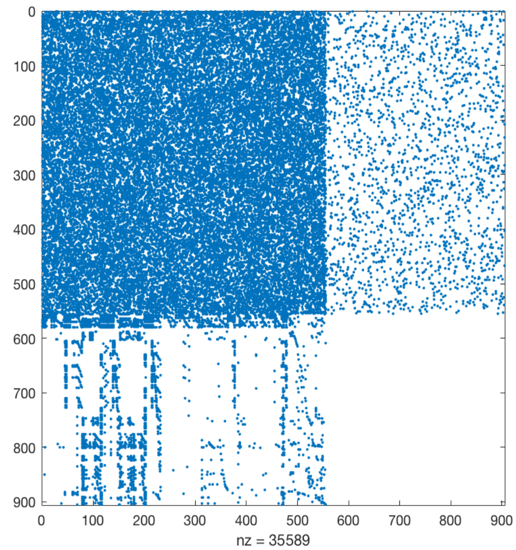

Figure 5.19. Sparsity pattern of $\mathcal{A}$ formed by random matrix $A \in \mathbb{R}^{555\times 555}$, random matrix $B_1 \in \mathbb{R}^{555\times 350}$ and `Maragal_2`=$B_2 \in \mathbb{R}^{350\times 555}$. Size: 905×905, n=555, m=350, number of nonzero entries=4582, condition number= Inf, rank(B_2)=172.

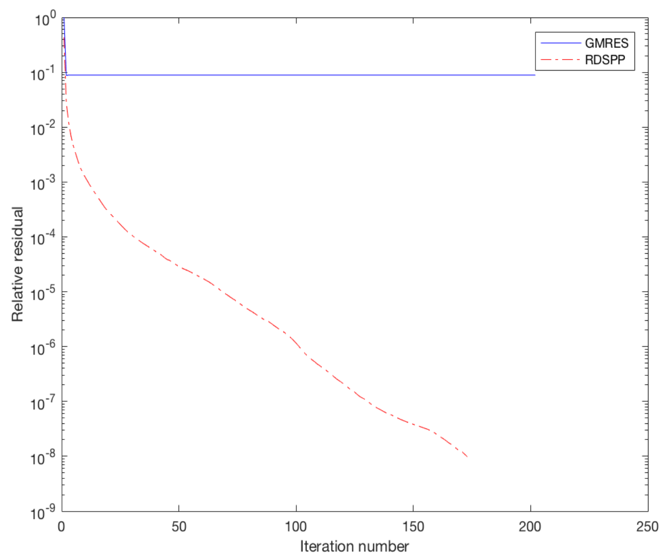

Figure 5.20. Relative residual *vs*. iteration number for $\mathcal{A}$ formed by random A, B_1 and `Maragal_2`=B_2. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix to the system with rank-deficient B_2.

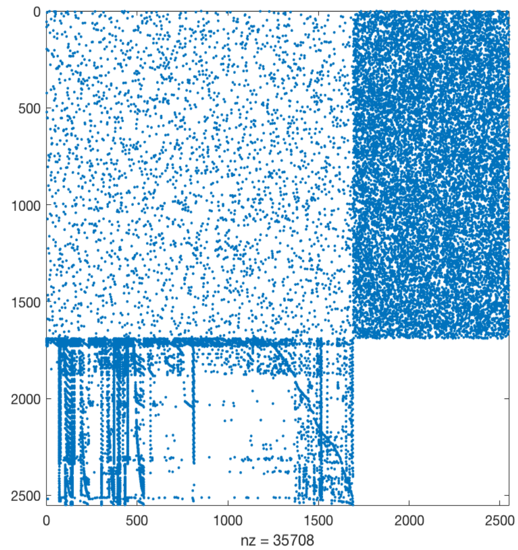

Figure 5.21. Sparsity pattern of $\mathcal{A}$ formed by random matrix $A \in \mathbb{R}^{1690\times1690}$, random matrix $B_1 \in \mathbb{R}^{1690\times860}$ and `Maragal_3`$=B_2 \in \mathbb{R}^{860\times1690}$. Size: 2550×2550, n=1690, m=860, number of nonzero entries=20130, condition number= Inf, rank(B_2)=613.

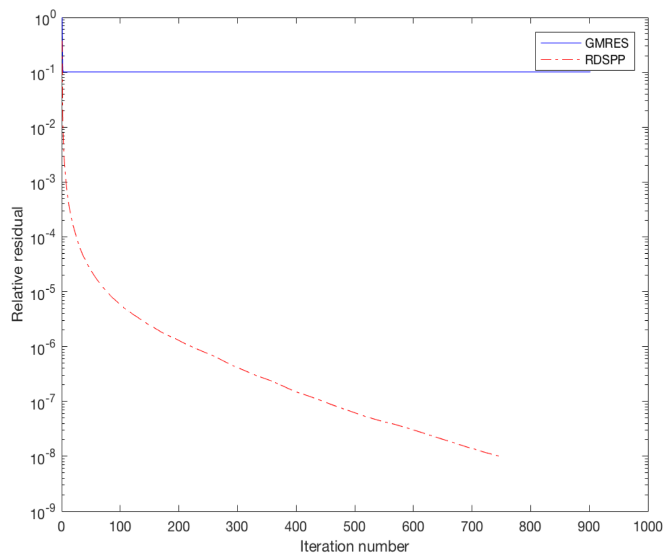

Figure 5.22. Relative residual *vs.* iteration number for $\mathcal{A}$ formed by random A, B_1 and `Maragal_3`$=B_2$. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix to the system with rank-deficient B_2.

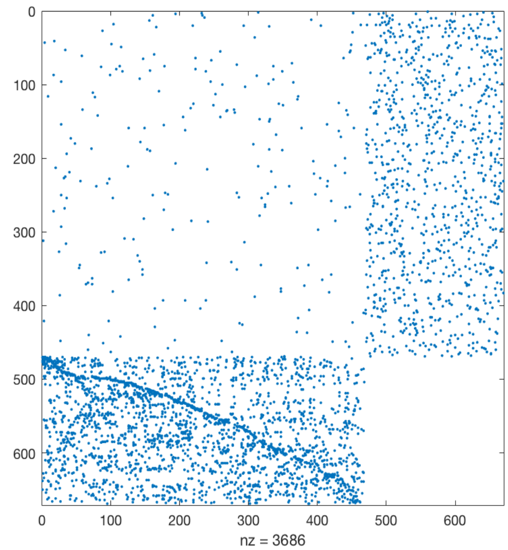

Figure 5.23. Sparsity pattern of $\mathcal{A}$ formed by random matrix $A \in \mathbb{R}^{469\times 469}$, random matrix $B_1 \in \mathbb{R}^{469\times 201}$ and GL6_D_6$=B_2 \in \mathbb{R}^{201\times 469}$. Size: 670×670, n=469, m=201, number of nonzero entries=2642, condition number= Inf, rank(B_2)=156.

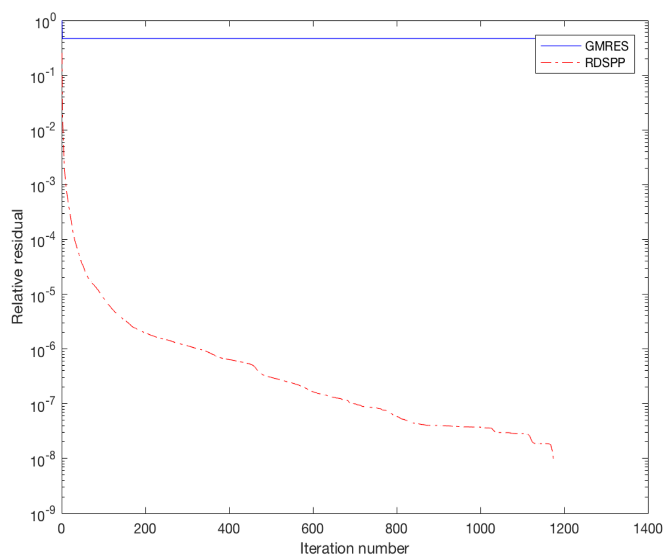

Figure 5.24. Relative residual *vs.* iteration number for $\mathcal{A}$ formed by random A, B_1 and GL6_D_6$=B_2$. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix to the system with rank-deficient B_2.

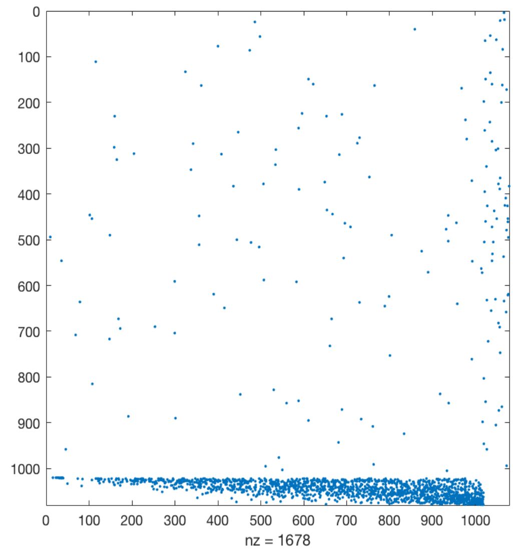

Figure 5.25. Sparsity pattern of $\mathcal{A}$ formed by random matrix $A \in \mathbb{R}^{1019\times1019}$, random matrix $B_1 \in \mathbb{R}^{1019\times60}$ and `GL7d11`$=B_2 \in \mathbb{R}^{60\times1019}$. Size: 1079×1079, n=1019, m=60, number of nonzero entries=1678, condition number= Inf, rank(B_2)=59.

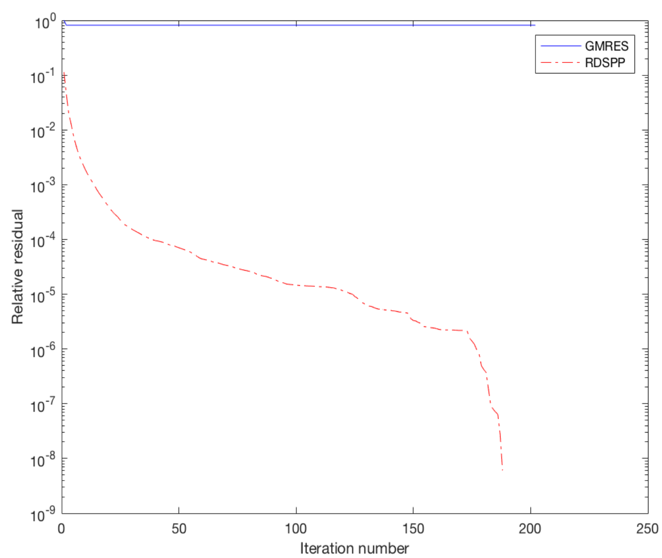

Figure 5.26. Relative residual *vs.* iteration number for $\mathcal{A}$ formed by random A, B_1 and `GL7d11`$=B_2$. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using the projection matrix to the system with rank-deficient B_2.

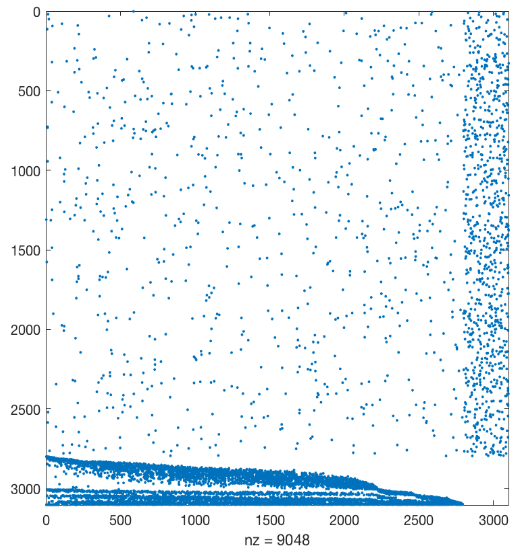

Figure 5.27. Sparsity pattern of $\mathcal{A}$ formed by random matrix $A \in \mathbb{R}^{2798\times 2798}$, random matrix $B_1 \in \mathbb{R}^{2798\times 305}$ and $\texttt{GL7d26}^T = B_2 \in \mathbb{R}^{305\times 2798}$. Size: 3003×3003, n=2798, m=305, number of nonzero entries=8273, condition number= Inf, rank(B_2)=273.

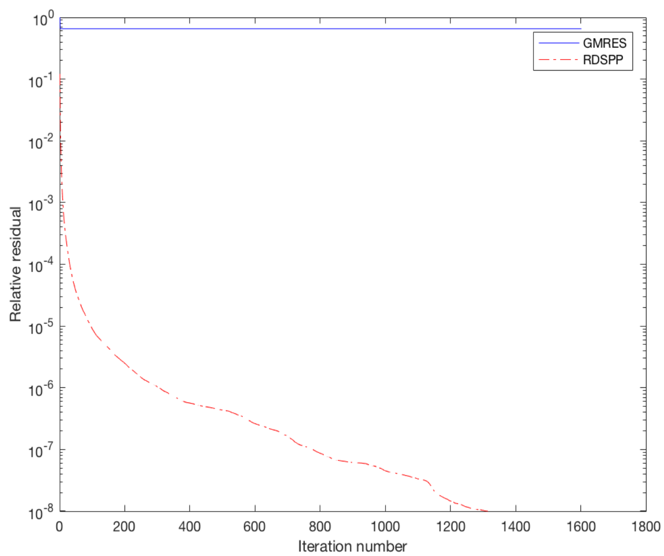

Figure 5.28. Relative residual *vs.* iteration number for $\mathcal{A}$ formed by random A, B_1 and $\texttt{GL7d26}^T = B_2$. Convergence of GMRES applied to the whole problem and LSMR applied to the least squares problem after using theprojection matrix to the system with rank-deficient B_2.

Our experiments show that our projection method for both full rank and rank-deficient B_2 has very good convergence. Our method works even for singular $\mathcal{A}$ and A. As illustrated in the results the system does not have to have a full rank to get good results by using our method. We did not use re-orthogonalization or preconditioning for solving the problem. Using a preconditioning is our future research for solving the saddle point system.

CHAPTER 6

CONCLUSION

In this thesis, we have investigated the iterative solutions of large and sparse saddle point systems of the form (1.2) by using a projection technique. The main contribution of this thesis is the that the presented technique can be applied to large class of saddle point problems. In other words, the technique does not necessarily require a specific form of block matrices except the (2,2)-block matrix in the saddle point matrix being 0-matrix.

In Chapter 3, we presented a solution method for full rank B_2. The main idea of the chapter was constructing a projection matrix by using full row rank matrix B_2 and then transforming the original problem into a least squares problem. Since the number of rows in B_2 is relatively small compared to the number columns in B_2 applying a projection matrix $Q = I - {B_2}^T (B_2 {B_2}^T)^{-1} B_2$ is not an expensive calculation. By using this technique the original problem is transformed into a least squares problem. Then the least squares problem is solved by using LSMR which is one of the Krylov subspace iterative method for solving the underdetermined systems. Numerical results show that the projection method converges faster than GMRES.

In Chapter 4, we worked on the rank-deficient B_2. Since B_2 is a rank-deficient matrix, $B_2 {B_2}^T$ is not an invertible matrix. Therefore, we cannot construct the same projection matrix that we use in Chapter 3. To build a different projection matrix we use only the maximal number of linearly independent rows of B_2, which we call B_{21}. Applying the projection matrix $\hat{Q} = I - {B_{21}}^T (B_{21} {B_{21}}^T)^{-1} B_{21}$ is not numerically expensive since the number of rows of B_{21} is very small. We use the projection matrix to transform the original problem to a least squares problem, then we solve the system by LSMR. It is numerically shown that our method is faster than the GMRES applied to the original system in numerical experiments.

It has been demonstrated that the projection method for saddle point systems with full rank or rank-deficient (2,1)-block has very good convergence in comparison to GMRES applied to the whole system.

REFERENCES

[1] M. Benzi, G. H. Golub, and J. Liesen. *Numerical solution of saddle point problems.* Acta Numerica, Vol. 14, pp.1-137, 2005.

[2] J. Nocedal and S. J. Wright. *Numerical Optimization.* Second edition, Springer, New York, 2006.

[3] Y. Saad. *Iterative methods for sparse linear systems.* Second edition, SIAM, Philadelphia, 2003.

[4] A. Greenbaum. *Iterative Methods for Solving Linear Systems.* First edition, SIAM, Philadelphia, 1997.

[5] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Third edition John Hopkins University Press, Baltimore, 1996.

[6] W. E. Arnoldi. *The principle of minimized iterations in the solution of the matrix eigenvalue problem.* Quarterly of Applied Mathematics, Vol. 9, No. 17, pp. 17-29, 1951.

[7] D. Braess and W. Dahmen. *A cascadic multigrid algorithm for the Stokes equations.* Numer. Math., Vol. 82, No. 2, pp. 179-191, 1999.

[8] V. Simoncini and M. Benzi. *Spectral properties of the Hermitian and skew Hermitian splitting preconditioner for saddle point problems.* SIAM J. Matrix Anal. Appl., Vol. 26, No. 2, pp. 377-389, 2004.

[9] Y. Saad and M. H. Schultz. *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems.* SIAM J. Sci. Stat. Comput., Vol. 7, No. 3 pp. 856-869, 1986.

[10] C. C. Paige and M. A. Saunders. *Solution of sparse indefinite systems of linear equations.* SIAM J. Numer. Anal., Vol. 12, No. 4, pp. 617-629, 1975.

[11] C. C. Paige and M. A. Saunders. *LSQR: An algorithm for sparse linear equations and sparse least squares.* ACM Trans. Math. Software, Vol. 8, No. 1, pp. 43-71, 1982.

[12] D. C. Fong and M. A. Saunders. *LSMR: An iterative algorithm for sparse least-squares problems.* SIAM J. Sci. Comput., Vol. 33, No. 5, pp. 2950-2971, 2011.

[13] H. M. Markowitz and A. F. Perold. *Sparsity and piecewise linearity in large portfolio optimization problems, in Sparse Matrices and Their Uses.* Academic Press, New York, 1981.

[14] A. Battermann and E. W. Sachs. *Block preconditioners for KKT systems arising in PDE-governed optimal control problems.* Birkhaäuser Basel, pp. 1-18, 2001.

[15] J. H. Bramble and J. E. Pasciak. *A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems.* Mathematics of Computations, Vol. 50, No. 181, pp. 1-17, 1988

[16] H. Q. Nguyen. *Domain decomposition methods for linear-quadratic elliptic optimal control problems.* PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2004.

[17] J. Liesen, E. de Sturler, A. Sheffer, Y. Aydin and C. Siefert. *Preconditioners for indefinite linear systems arising in surface parameterization, in Proc.* 10th International Meshing Round Table, pp. 71-81, 2001.

[18] S. Turek. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach.* Vol. 6 of Lecture Notes in Computational Science and Engineering, Springer, Berlin, 1999.

[19] P. Wesseling. *Principles of Computational Fluid Dynamics.* Vol. 29 of Springer Series in Computational Mathematics, Springer, Berlin, 2001.

[20] P. E. Gill, W. Murray and M. H. Wright. *Practical Optimization.* Academic Press Inc., London, 1981.

[21] A. Battermann and M. Heinkenschloss. *Preconditioners for Karush-Kuhn-Tucker matrices arising in the optimal control of distributed systems.* Birkhaäuser, pp. 15-32, 1998.

[22] S. J. Wright. *Primal Dual Interior Point Methods.* SIAM, Philadelphia, PA, 1997.

[23] I. Perugia, V. Simoncini and M. Arioli. *Linear algebra methods in a mixed approximation of magnetostatic problems.* SIAM J. Sci. Comput. Vol. 21, No. 3, pp. 1085-1101, 1999.

[24] J. T. Betts. *Practical Methods for Optimal Control Using Nonlinear Programming.* SIAM, Philadelphia, 2001.

[25] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations.* First edition, Springer, Berlin, 1994.

[26] W. Leontief, F. Duchin and D. B. Szyld. *New approaches in economic analysis.* Science Vol. 228, No. 4698, pp. 419-422, 1985.

[27] H. Elman and D. Silvester. *Fast nonsymmetric iterations and preconditioning for Navier-Stokes equations.* SIAM J. Sci. Comput. Vol. 17, No. 1, pp. 33-46, 1996.

[28] L. Zhu, A. Beaudoin, and S. Macewan. *A study of kinetics in stress relaxation of AA 5182.* In Proceedings of TMS Fall 2001: Microstructural Modeling and Prediction During Thermomechanical Processing, pp. 189-199, 2001.

[29] E. L. Hall. *Computer Image Processing and Recognition.* Academic Press, New York, 1979.

[30] R. Glowinski. *Numerical Methods for Nonlinear Variational Problems.* Springer Series in Computational Physics, Springer, New York, 1984.

[31] Z.-H. Cao. *A note on constraint preconditioning for nonsymmetric indefinite matrices.* SIAM J. Matrix Anal. Appl. Vol. 24, No. 1, pp.121-125, 2002.

[32] C. Keller, N. I. M. Gould and A. J. Wathen. *Constraint preconditioning for indefinite linear systems.* SIAM J. Matrix Anal. Appl. Vol. 21, No. 4, pp. 1300-1317, 2000.

[33] H. M. Markowitz. *Portfolio Selection: Efficient Diversification of Investments.* Wiley, New York, 1959.

[34] D. J. Silvester and A. J. Wathen. *Fast iterative solution of stabilised Stokes systems II: Using general block preconditioners.* SIAM J. Numer. Anal., Vol. 31, No.5, pp. 1352-1367, 1994

[35] K. J. Arrow, L. Hurwicz and H. Uzawa. *Studies in Linear and Nonlinear Programming.* Stanford University Press, Stanford, CA, 1958.

[36] R. W. Freund. *Model reduction methods based on Krylov subspaces.* Acta Numerica, Vol. 12, pp. 267-319, 2003.

[37] R. A. Horn and C. R. Johnson. *Matrix analysis.* Second edition, Cambridge University Press, New York, 2013.

[38] C. Farhat, K. Pierson , and M. Lesoinne. *The second generation of FETI methods and their applications to the parallel solution of large scale linear and geometrically nonlinear structural analysis problems.* Computer Methods in Applied Mechanics and Engineering, Vol. 184, No. 2-4, pp. 333-374, 2000.

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

[40] M. F. Murphy, G. H. Golub, and A. J. Wathen. *A note on preconditioning for indefinite linear systems.* SIAM J. Sci. Comput.,Vol. 21, No. 6, pp. 1969-1972, 2000.

[41] I. C. F. Ipsen. *A note on preconditioning nonsymmetric matrices.* SIAM J. Sci. Comput., Vol. 23, No. 3, pp. 1050-1051, 2001.

[42] D. Braess. *Finite elements. Theory, Fast Solvers, and Applications in Solid Mechanics.* Second edition, Cambridge University Press, Cambridge, 2001.

[43] M. Benzi, M. J. Gander and G. H. Golub. *Optimization of the Hermitian and skew-Hermitian splitting iteration for saddle-point problems.* BIT Numerical Mathematics, Vol. 43, No. 5, pp. 881-900, 2003.

[44] X. Chen. *On preconditioned Uzawa methods and SOR methods for saddle point problems.* J. Comput. Appl. Math. Vol. 100, No.2, pp. 207-224, 1998

[45] M. R. Hestenes and E. Stiefel. *Conjugate Gradients for Solving Linear Systems.* Journal of Research of the National Bureau of Standards, Vol. 49, No. 6, 1952.

[46] G. Golub and W. Kahan. *Calculating the singular values and pseudo-inverse of a matrix*. SIAM Ser. B Numer. Anal., Vol. 2, No. 2, 1965.

[47] T. A. Davis. *SuiteSparse Matrix Collection*, https://sparse.tamu.edu/.

[48] O. Axelsson and M. Neytcheva. *Preconditioning methods for linear systems arising in constrained optimization problems.* Numerical Linear Algebra With Applications, Vol. 10, No. 1, pp. 3-31, 2003.

[49] C. G. Cullen An Introduction to Numerical Linear Algebra Volume 1, PWS Publishing Company, 1994.

[50] W. Zulehner. *Analysis of iterative methods for saddle point problems: a unified approach.* Mathematics of Computations, Vol. 71, No. 238, pp. 479-505, 2002.

[51] E. de Sturler and J. Liesen. *Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems, Part I: Theory.* SIAM J. Sci. Comput., Vol. 26, No. 5, pp. 1598-1619, 2005.

[52] V. Simoncini. *Block triangular preconditioners for symmetric saddle-point problems.* Applied Numerical Mathematics, Vol. 49, No. 1, pp. 63-80, 2004.

[53] M. Benzi and G. H. Golub. *A preconditioner for generalized saddle point problems.* SIAM J. Matrix Anal. Appl. Vol. 26, No. 1, pp. 20-41, 2004.

[54] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary Value Problems.* Studies in Mathematics and Its Applications Vol. 15, North Holland, Amsterdam, 1983.

[55] S. Axler Linear Algebra Done Right Second edition, Springer, New York, 1997.

BIOGRAPHICAL STATEMENT

Gul Karaduman was born in Ankara, Turkey, in 1985. She received her Bachelor of Science degree in Mathematics from Ankara University, Master of Science degree in Mathematics Education from Gazi University and Master of Art in Mathematics from Boston University in 2007, 2009, and 2012 respectively. She joined the Ph.D. program in Mathematics at the University of Texas at Arlington in the fall of 2012.

Gul's research interest includes numerical linear algebra, numerical analysis, number theory, and cryptography.