RICE UNIVERSITY

Spatial Domain Decomposition and Model Reduction for Parabolic Optimal Control Problems

by

Kai Sun

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

Master of Arts

APPROVED, THESIS COMMITTEE:

Danny Sorensen, Chairman
Noah Harding Professor of Computational and Applied Mathematics

Matthias Heinkenschloss
Professor of Computational and Applied Mathematics

Athanasios Antoulas
Professor of Electrical and Computer Engineering

HOUSTON, TEXAS

APRIL, 2006
The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.
Abstract

Spatial Domain Decomposition and Model Reduction for Parabolic Optimal Control Problems

by

Kai Sun

In this thesis, we propose a spatial domain decomposition method and model reduction techniques for the solution of linear-quadratic parabolic optimal control problems. Such problems arise directly from many applications such as the data assimilation, circuit design and oil reservoir modeling. The motivation for this work is threefold. First, we attempt to address the storage issue in numerically solving the parabolic optimal control problem. Secondly, spatial domain decomposition leads to parallelism. Therefore, data can be decomposed uniformly by assigning subdomains to each processor. Finally, for large-scale problems, the subproblems on the subdomains are still very large. Model reduction techniques applied to the subproblems are expected to dramatically reduce the size of the subproblems and save computational time.
Acknowledgements

Chinese always say "he who teaches me for one day is my father for life".

In the beginning, I would like to gratefully acknowledge the enthusiastic supervision of Dr. Dan Sorensen. He is a great teacher, dutiful friend and kindly father. I could not have imagined having a better supervisor. And without his common-sense, knowledge, perceptiveness and guidance, I would never have finished this work.

Also, I would like to thank Dr. Matthias Heinkenschloss who gives me a lot of guidance and help. His knowledge in optimal control problem is a key to my work. I would like to thank Dr. Athanasios Antoulas who taught me system theory and be my thesis committee member. I also would like to thank my colleague, Quming Zhou, for his practical applications and technical discussions.

Next, there are all CAAM members and the friends I made at Rice University. Their friendship, teaching, helping and sharing made me a really happy person.

Finally, I have to say "thank-you"to: all my friends and family, wherever they are, particularly my wife, mother and father for your support.

May the grace and peace be with you forever!
Contents

Abstract ii

Acknowledgements iii

List of Figures vi

List of Tables vii

1 Introduction 1
   1.1 Problem Overview ..................................... 1
   1.2 Methods, Approaches and Contributions .................. 2
   1.3 Implementations and Applications ........................ 3
   1.4 Thesis Outline ........................................ 4

2 Example Problem 5
   2.1 Semi-Discretization .................................... 5
   2.2 Fully Discretization ................................... 6
   2.3 Optimality Conditions .................................. 7
   2.4 Standard Methods ..................................... 8
      2.4.1 Optimality Conditions Reformulation ................. 8
      2.4.2 Conjugate Gradient (CG) ............................ 9

3 Domain Decomposition Schur Complement Formulation 11
3.1 Introduction of Non-overlapping Domain Decomposition in Space ................................................. 12
  3.1.1 Notation and Terminology ........................................................................................................ 12
  3.1.2 Schur Complement .................................................................................................................... 14
3.2 Example Problem Reformulation .................................................................................................... 14
  3.2.1 Spatial Decomposition .............................................................................................................. 14
  3.2.2 Schur Complement Reformulation ............................................................................................ 16

4 Model Reduction ................................................................................................................................. 18
  4.1 Introduction to Linear Dynamical Systems ..................................................................................... 18
  4.2 Balanced Truncation Method .......................................................................................................... 22
  4.3 State Space Representation for the Full Problem ........................................................................... 23
  4.4 State Space Representation for the Subproblem ........................................................................... 24
  4.5 Model Reduction on Subdomains .................................................................................................. 26

5 Solution Algorithm ............................................................................................................................... 27
  5.1 GMRES ........................................................................................................................................ 27
  5.2 Neumann-Neumann Preconditioner .............................................................................................. 28

6 Numerical Results ................................................................................................................................. 30
  6.1 Optimal Control Problem No.1 ...................................................................................................... 30
    6.1.1 Coarse Mesh ............................................................................................................................ 31
    6.1.2 Fine Mesh ............................................................................................................................... 33
  6.2 Optimal Control Problem No.2 ...................................................................................................... 35
    6.2.1 Coarse Mesh ............................................................................................................................ 35
    6.2.2 Fine Mesh ............................................................................................................................... 36

7 Conclusions and Future Work ............................................................................................................... 39
  7.1 Summary and Conclusions ............................................................................................................. 39
  7.2 Future work ..................................................................................................................................... 40
# List of Figures

3.1 A Domain Decomposition Example ........................................ 12

6.1 Problem domain $\Omega$ .................................................. 31
6.2 Coarse finite element mesh on $\Omega$ .................................. 32
6.3 Controls and observations on coarse mesh .......................... 32
6.4 Controls and observations on coarse mesh .......................... 33
6.5 fine finite element mesh on $\Omega$ ...................................... 34
6.6 Controls and observations on fine mesh .............................. 34
6.7 Controls and observations on fine mesh .............................. 35
6.8 $f(x,t)$ on coarse mesh .................................................. 36
6.9 Approximate control and observation on coarse mesh ........... 37
6.10 $f(x,t)$ on coarse mesh .................................................. 37
6.11 Approximate control and observation on fine mesh ............. 38
List of Tables
Chapter 1

Introduction

1.1 Problem Overview

During the last years, optimal control problems have gained considerable attention. Some examples can be found in [2] [19] [20]. In this thesis, we shall present some ideas and computational results on these problems.

To illustrate our ideas, we consider the example problem

\[
\min_{u(x,t)} \frac{\alpha_1}{2} \int_0^T \int_{\Omega_0} (y(x,t) - z(x,t))^2 \, dx \, dt + \frac{\alpha_2}{2} \int_0^T \int_{\Omega_c} u^2(x,t) \, dx \, dt
\]

(1.1)

where \( y(x,t) \) satisfies the state equation:

\[
\partial_t y(x,t) - \Delta y(x,t) = f(x,t) + u(x,t), \quad x \in \Omega, \ t \in [0,T], \ u \in \chi_{\Omega_c}(x),
\]

(1.2)

\[
y(x,t) = 0, \quad \partial \Omega \times (0,T),
\]

(1.3)

\[
y(x,0) = y_0(x).
\]

(1.4)

Here, \( \Omega \subset \mathbb{R}^d, \ d = 1, 2, 3, \) is a spatial domain and \( \Omega_c, \Omega_0 \subset \Omega \) are open sets. The function \( \chi_{\Omega_c} \) is the indicator function on \( \Omega_c. \)

The functions \( z \in L^2(0,T; L^2(\Omega_0)) \) and \( f \in L^2(0,T; L^2(\Omega)) \) are given, and \( \alpha_1 \geq 0, \alpha_2 \geq 0 \) are given parameters. The problem has to be solved for \( y \in \mathcal{Y} \) and \( u \in \mathcal{U}, \) where
\[ \mathcal{V} = W(0, T) = \{ y : y \in L^2(0, T; H_0^1(\Omega)), y' \in L^2(0, T; H^{-1}(\Omega)) \} \quad (1.5) \]

and

\[ \mathcal{U} = L^2(0, T; L^2(\Omega_c)). \quad (1.6) \]

The optimal control problems are interesting and difficult. There are many applications [2] [19] [20] for them, such as data assimilation and circuit design. For instance, we consider the inverse problem of determining an arbitrary source in a time-dependent convective-diffusive transport equation, given a velocity field and pointwise measurements of the concentration. Applications that give rise to such problems include determination of groundwater or airborne pollutant sources from measurements of concentrations and identification of sources of chemical or biological attacks. In circuit design, the inverse problem of determining the arbitrary input source of electrical current in a power grid system with given measurements of the voltage also has nearly the same form.

Our linear-quadratic parabolic optimal control problem (1.1)-(1.4) is a special case (without an advection term) of such inverse problems. These problems also arise as subproblems in Newton or SQP methods for the solution of nonlinear parabolic optimal control problems, such as fluid [1], glass cooling [24], steel cooling [26], semiconductor processing [21] and flow control [9].

### 1.2 Methods, Approaches and Contributions

The linear-quadratic optimal control problems usually are solved by standard methods like GMRES or CG. The numerical solution of such problems is difficult due to the large amount of storage requirements arising from the coupling of states, adjoints and controls.

Recently, [12] [13] [14] [15] presented a non-overlapping spatial domain decomposition method for this class of optimal control problems. This method introduces
optimization level parallelism into the solution approach and reduce the amount of permanent storage required.

Domain decomposition has been proposed as a means to solve large-scale problems. However, these can still cost too much time and storage. Based on the method proposed by Heinkenschloss and Sorensen, we further apply model reduction technique to the subproblems arising in the domain decomposition method. "Model reduction seeks to replace a large-scale dynamical system of differential or difference equations by a system of substantially lower dimension that has nearly the same response characteristics."[16] By applying this technique, we save both storage and CPU time for solving the problems.

This thesis extends domain decomposition method with a model reduction technique to the solution of two dimensional linear-quadratic parabolic optimal control problems. We present numerical results for this approach and compare the results with the standard Conjugate Gradient approach.

1.3 Implementations and Applications

The implementation of our approach contains finite element method, domain decomposition method, model reduction technique and preconditioning technique. Currently, all the codes have been written in MATLAB. They work for 2-D problems. The long term objective is to solve 3-D problem, rewrite the codes in C/C++ (using some mathematical software packages, such as Trilinos) and implement them on some parallel machine. We applied the codes to some real problems arising in circuit design. The computational results indicate this approach is competitive with existing methods on these problems.
1.4 Thesis Outline

- Chapter 1 is an overview of this thesis.

- Chapter 2 gives a brief description of our optimization problems with optimality condition and existing standard methods.

- Chapter 3 introduces spatial domain decomposition Schur complement formulation to the example problem. Most materials in this chapter based on Dr. Heinkenschloss’ approach.

- Chapter 4 gives an overview of model reduction technique first. Then it shows how this technique can be applied to the subproblems arising in domain decomposition method.

- Chapter 5 presents our fully detailed algorithms for solving the example problem.

- Chapter 6 gives the computational results.

- Chapter 7 gives conclusions and describes future work.
Chapter 2

Example Problem

In this chapter, we mainly focus on the example problem (1.1)-(1.4). The optimality condition for the discretized example problem will be derived. In the end, we shall give a briefly description of the existing straightforward methods.

2.1 Semi-Discretization

Let $V^h \subset H^1_0(\Omega)$, $U^h \subset L^2(\Omega_c)$ be finite dimensional subspaces with bases $\psi_1, \ldots, \psi_n$ and $\mu_1, \ldots, \mu_m$, respectively. We approximate the states and controls by $y_h \in H^1(0, T; V^h)$ and $u_h \in L^2(0, T; U^h)$ defined as

\[ y_h = \sum_{i=1}^{n} y_i(t) \psi_i, \quad u_h = \sum_{j=1}^{m} u_j(t) \mu_j. \tag{2.1} \]

In our computation, we use linear conforming finite elements in space to discretize the states $y$ and the controls $u$. We can define $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $Q \in \mathbb{R}^{n \times n}$,
$M \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ as follows:

\begin{align*}
A_{ij} &= \int_\Omega \nabla \psi_i^T \nabla \psi_j \, dx, \\
B_{ij} &= -\int_\Omega \psi_i \mu_j \, dx, \\
Q_{ij} &= \int_{\Omega_o} \psi_i \psi_j \, dx, \\
M_{ij} &= \int_\Omega \psi_i \psi_j \, dx, \\
R_{ij} &= \int_{\Omega_c} \mu_i \mu_j \, dx.
\end{align*}

(2.2)\quad (2.3)\quad (2.4)\quad (2.5)\quad (2.6)

We replace $z$ by its piecewise linear interpolant. Our finite element semi-discretization of the optimal control problem (1.1)-(1.4) leads to a large-scale linear quadratic problem of the form

\begin{align*}
\min \quad & \frac{\alpha_1}{2} \int_0^T (y(t) - z(t))^T Q (y(t) - z(t)) \, dt + \frac{\alpha_2}{2} \int_0^T u(t)^T R u(t) \, dt, \\
\text{s.t.} \quad & My'(t) + Ay(t) + Bu(t) = f(t), \quad t \in (0, T), \\
& y(0) = y_0,
\end{align*}

(2.7)\quad (2.8)\quad (2.9)

where $z(t) = \begin{pmatrix} z_1(t) & z_2(t) & \cdots & z_n(t) \end{pmatrix}^T$,

\begin{align*}
z_i(t) &= \begin{cases} 
z(x_i, t) & x_i \text{ is a vertex in } \Omega_o, \\
0 & \text{otherwise.}
\end{cases}
\end{align*}

(2.10)

2.2 Full Discretization

Furthermore, by using a given uniformly spaced set of time points \( \{ t_k \}_{k=0}^N \) on \([0, T]\) and the Backward-Euler method, the discretized problem (2.7) - (2.9) becomes

\begin{align*}
\min \quad & \frac{\alpha_1}{2} h \sum_{k=1}^N (y_k - z_k)^T Q (y_k - z_k) + \frac{\alpha_2}{2} h \sum_{k=1}^N u_k^T R u_k, \\
\text{s.t.} \quad & My_k - y_{k-1} h + Ay_k + Bu_k = f_k,
\end{align*}

(2.11)\quad (2.12)
with given $y_0$. Here $y_k = y(t_k)$, $h = \frac{T}{N}$ and $t_k = kh$. We rewrite the problem as

$$\min \quad \frac{\alpha_1}{2} h \sum_{k=1}^{N} (y_k - z_k)^T Q (y_k - z_k) + \frac{\alpha_2}{2} h \sum_{k=1}^{N} u_k^T R u_k,$$ \hspace{1cm} (2.13)

s.t. \quad (M + hA)y_k - My_{k-1} + hBu_k = hf_k. \hspace{1cm} (2.14)

Now we have a quadratic programming problem with equality constrains.

### 2.3 Optimality Conditions

The Lagrangian function of the optimization problem (2.13) - (2.14) is defined as:

$$\mathcal{L}(y, u, p) = \frac{\alpha_1}{2} h \sum_{k=1}^{N} (y_k - z_k)^T Q (y_k - z_k) + \frac{\alpha_2}{2} h \sum_{k=1}^{N} u_k^T R u_k$$

$$- \sum_{k=1}^{N} p_k^T ((M + hA)y_k - My_{k-1} + hBu_k - hf_k).$$ \hspace{1cm} (2.15)

The optimality conditions of the optimization problem are given by

$$\frac{\partial \mathcal{L}(y, u, p)}{\partial y_k} = 0,$$ \hspace{1cm} (2.16)

$$\frac{\partial \mathcal{L}(y, u, p)}{\partial u_k} = 0,$$ \hspace{1cm} (2.17)

$$\frac{\partial \mathcal{L}(y, u, p)}{\partial p_k} = 0.$$ \hspace{1cm} (2.18)

That is

$$M^T p_{k+1} - (M^T + hAT)p_k + \alpha_1 hQy_k = \alpha_1 hQz_k, \quad k = 1, ..., N - 1,$$ \hspace{1cm} (2.19)

$$-(M^T + hAT)p_N + \alpha_1 hQy_N = \alpha_1 hQz_N,$$ \hspace{1cm} (2.20)

$$\alpha_2 Ru_k - B^T p_k = 0, \quad k = 1, 2, ..., N,$$ \hspace{1cm} (2.21)

$$(M + hA)y_k - My_{k-1} + hBu_k = hf_k, \quad k = 1, 2, ..., N.$$ \hspace{1cm} (2.22)
2.4 Standard Methods

In the previous chapter, we mentioned the example problem could be solved by some straightforward methods. Here, we show two different approaches, GMRES and Conjugate Gradient.

2.4.1 Optimality Conditions Reformulation

This is a reformulation of the optimality condition as a linear equations. By letting

\[
\Lambda = \begin{pmatrix}
\alpha_2 hR & 0 & -hB^T \\
0 & \alpha_1 hQ & -M^T - hA^T \\
-hB & -M - hA & 0
\end{pmatrix}, \quad \Phi = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & M^T \\
0 & 0 & 0
\end{pmatrix},
\]

(2.23)

and

\[
w_k = \begin{pmatrix}
u_k \\
y_k \\
p_k
\end{pmatrix}, \quad k = 1, \ldots, N,
\]

(2.24)

\[
b_1 = \begin{pmatrix}0 \\
hf_1 + My_0 \\
\alpha_1 hQz_1 \end{pmatrix}, \quad b_k = \begin{pmatrix}0 \\
hf_k \\
\alpha_1 hQz_k \end{pmatrix}, \quad k = 2, \ldots, N.
\]

(2.25)

The optimality conditions (2.19) - (2.22) can be reformulated as a symmetric linear system:

\[
\begin{pmatrix}
\Lambda & \Phi \\
\Phi^T & \Lambda
\end{pmatrix}
\begin{pmatrix}
w_1 \\
\vdots \\
w_N
\end{pmatrix}
= \begin{pmatrix}
b_1 \\
\vdots \\
b_N
\end{pmatrix}.
\]

(2.26)

It is a sparse indefinite symmetric linear system, which may be solved by iterative methods like SYMMLQ method and minimum residual method. The solution to this system gives the solution of approximated optimal control problem (2.13) - (2.14).
2.4.2 Conjugate Gradient (CG)

As the number of the states and controls, the number of time steps $N$ increasing, the size of the linear system (2.26) increases very rapidly. For instance, for a problem with $N$ states and 1 control with 400 time steps, the size of the linear system (2.26) will be $400 \times (2N + 1)$. This is a small problem. When higher accuracy and (or) 3D is required, the problems become much larger and soon become intractable.

Now, let us introduce the classical CG method which only focuses on the controls. The CG method is an effective method for unconstrained convex optimization problems. In order to use CG, we must show that the example problem is convex and can be transformed to an unconstrained problem.

After a discretization in time one arrives at a minimization problem

$$
\min_u \hat{f}(u),
$$

where the objective function is of the form

$$
\hat{f}(u) = f(y, u) = \frac{\alpha_1}{2} \sum_{k=1}^{N} (y_k - z_k)^T Q(y_k - z_k) h + \frac{\alpha_2}{2} \sum_{k=1}^{N} u_k^T R u_k h
$$

and $y(u)$ is the solution of an implicit linear equation

$$
c(y, u) = \begin{pmatrix}
c_1(y, u) \\
c_2(y, u) \\
\vdots \\
c_N(y, u)
\end{pmatrix} = 0,
$$

where

$$
c_k(y, u) = (M + hA)y_k - My_{k-1} + hBu_k - hf_k.
$$

Since this is a linear-quadratic form, it is quite straightforward to calculate the gradient $\nabla \hat{f}(u)$ and the hessian $\nabla^2 \hat{f}(u)$. One can easily verify that

$$
\nabla \hat{f}(u) = -c_u(y(u), u)^T c_y(y(u), u)^T \nabla_y f(y(u), u) + \nabla_u f(y(u), u)
$$
and

\[
\nabla^2 \hat{f}(u) = c_u^T(y, u)c_y^{-T}(y, u)\nabla_{yy}\mathcal{L}(y, u, \lambda)c_y^{-1}(y, u)c_u(y, u) + \nabla_{uu}\mathcal{L}(y, u, \lambda),
\]

(2.32)

where \(\mathcal{L}(y, u, \lambda)\) is the Lagrangian function.

By using the gradient and hessian formulas, we can use the iterative method such as Conjugate Gradient method to solve this problem as an unconstrained problem.

In this chapter, we have shown how to discretize the example problem and how to solve it by the straightforward methods. However, these methods still lack the capability to solve large-scale problems. In next chapter, we shall present domain decomposition method for large-scale problems.
Chapter 3

Domain Decomposition Schur Complement Formulation

The computational cost for solving optimal control problems by the standard methods is still expensive. Therefore, we shall introduce a domain decomposition method as a means to overcome this cost. The basic idea of domain decomposition is to split a large problem into several much smaller subproblems. The key is to obtain subproblems that are very cheap to solve along with a cost effective reconstruction technique.

Domain decomposition methods have been applied previously to linear-quadratic time dependent optimal control problems. They split into time domain decomposition methods [8] [17] [18] [10] and spatial domain decomposition methods [5] [6] [7] [12]. Like [5] [6] [7] [12], the approach introduced in this paper is also based on a decomposition of the spatial domain. The resulting subproblems are smaller linear-quadratic parabolic optimal control problems posed on a spatial-subdomain-time cylinder. The difference between the approaches [5] [6] [7] and the approach [12] lies in the way the subdomain problems are coupled and in the solution method for the coupled subdomain problems.
3.1 Introduction of Non-overlapping Domain Decomposition in Space

Domain decomposition methods refer to a collection of techniques which revolve around the principle of divide-and-conquer. Such methods have been primarily developed for solving Partial Differential Equations over regions in two or three dimensions.

3.1.1 Notation and Terminology

To show the idea of domain decomposition, we consider the problem of solving the Laplace Equation on a rectangular shaped domain partitioned as shown in Figure 3.1.1. Domain decomposition attempts to solve the problem on the entire domain

$$\Omega = \bigcup_{i=1}^{n} \Omega_i,$$

by solving the smaller subdomain problems and combining this in a consistent way to construct a solution on the original full domain. When partitioning a problem,

![Diagram of a 2x2 grid with subdomains $\Omega_1$ and $\Omega_2$.](image)

Figure 3.1: A Domain Decomposition Example.

it is common to use graph representations. Since the subproblems obtained from a given partitioning will eventually be mapped into distinct processors, there are some
restrictions regarding the type of partitioning allowed. For example, in Element-By-Element finite element techniques, it may be desirable to map elements into processors instead of vertices. In this case, the restriction means no element should be split between two subdomains, i.e., all information related to a given element is mapped to the same processor. These partitionings are termed element-based non-overlapping.

We assume that the following problem is to be solved

\[ \Delta u = f \text{ in } \Omega \]  \hspace{1cm} (3.1)
\[ u = u_\Gamma \text{ on } \Gamma = \partial \Omega \]  \hspace{1cm} (3.2)

Domain decomposition methods are all implicitly or explicitly based on different ways of handling the unknown at the interfaces. From the PDE point of view, if the value of the solution is known at the interfaces between the different subdomains, these values could be used in Dirichlet-type boundary conditions and we will obtain \( s \) uncoupled Poisson equations. We can then solve these equations to obtain the value of the solution at the interior points. If the whole domain is discretized by either finite elements or finite difference techniques, then this is easily translated into the resulting linear system.

Assume that the problem associated with domain shown in Figure 3.1.1 is discretized with centered differences. We can label the nodes by subdomains. Please note that the interior nodes are labeled subdomain by subdomain. Then the interface nodes are labeled last. As a result, for a general partitioning into \( s \) subdomains, the linear system associated with the problem has the following structure:

\[
\begin{pmatrix}
A_1 & E_1 \\
A_2 & E_2 \\
\vdots & \vdots \\
A_s & E_s \\
F_1 & F_2 & \cdots & F_s & A_\Gamma
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\vdots \\
x_s \\
x_\Gamma
\end{pmatrix}
=
\begin{pmatrix}
f_1 \\
f_2 \\
\vdots \\
f_s \\
g
\end{pmatrix}
\hspace{1cm} (3.3)
\]

where each \( x_i \) represents the subvector of unknowns that are interior to subdomain \( \Omega_i \) and \( y \) represents the vector of all interface unknowns. It is useful to express the
above system in the simpler form,

\[ A \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix} \quad \text{with} \quad A = \begin{pmatrix} B & E \\ F & A_\Gamma \end{pmatrix}. \quad (3.4) \]

### 3.1.2 Schur Complement

Consider the linear system written in the form (3.3), in which \( B \) is assumed to be nonsingular. From the first equation the unknown \( x \) can be expressed as

\[ x = B^{-1}(f -Ey). \quad (3.5) \]

Upon substituting this into the second equation, the following reduced system is obtained:

\[ (A_\Gamma - FB^{-1}E)y = g - FB^{-1}f. \quad (3.6) \]

The matrix

\[ A_\Gamma - FB^{-1}E \quad (3.7) \]

is called the Schur complement matrix associated with the \( y \) variable. If this matrix can be formed and the linear system (3.6) can be solved, all the interface variables \( y \) will become available. Once these variables are known, the remaining unknowns can be computed, via (3.5). Because of the particular structure of \( B \), observe that any linear system solution with it decouples in \( s \) separate systems. The parallelism in this situation arises from this natural decoupling.

### 3.2 Example Problem Reformulation

#### 3.2.1 Spatial Decomposition

We divide \( \Omega \) into nonoverlapping subdomains \( \Omega_i, \quad i = 1, \ldots, s \). We define the interface of \( i \)th subdomain as \( \Gamma_i = \partial \Omega_i \setminus \partial \Omega \) and the whole interface as \( \Gamma = \bigcup_{i=1}^{s} \Gamma_i \). To simplify
the presentation, we assume that the number of subdomains is two, i.e. \( s = 2 \). Everything can be generalized to \( s > 2 \) subdomains.

After a suitable reordering of row and columns based on domain decomposition, the stiffness matrix can be written as

\[
A = \begin{pmatrix}
A_1 & E_1 \\
A_2 & E_2 \\
F_1 & F_2 & A_\Gamma
\end{pmatrix}.
\]  
(3.8)

Since \( M \) is a diagonal matrix,

\[
M = \begin{pmatrix}
M_1 \\
M_2 \\
M_\Gamma
\end{pmatrix}.
\]  
(3.9)

The vectors \( y_0, y_k \) and \( p_k \) can be partitioned into

\[
y_0 = \begin{pmatrix}
y_0^{(1)} \\
y_0^{(2)} \\
y_0^{(\Gamma)}
\end{pmatrix}, \quad y_k = \begin{pmatrix}
y_k^{(1)} \\
y_k^{(2)} \\
y_k^{(\Gamma)}
\end{pmatrix}, \quad p_k = \begin{pmatrix}
p_k^{(1)} \\
p_k^{(2)} \\
p_k^{(\Gamma)}
\end{pmatrix}.
\]  
(3.10)

We assume that all control points and observation points lie in the interior of the subdomains. This implies that we can partition \( B, Q \) and \( z_k \) as follows

\[
B = \begin{pmatrix}
B_1 & 0 \\
0 & B_2
\end{pmatrix}, \quad Q = \begin{pmatrix}
Q_1 & 0 & 0 \\
0 & Q_2 & 0 \\
0 & 0 & 0
\end{pmatrix}, \quad z_k = \begin{pmatrix}
z_k^{(1)} \\
z_k^{(2)} \\
0
\end{pmatrix}.
\]  
(3.11)

And the matrix \( R \) and control \( u_k \) can be decomposed analogously into

\[
R = \begin{pmatrix}
R_1 & 0 \\
0 & R_2
\end{pmatrix}, \quad u_k = \begin{pmatrix}
u_k^{(1)} \\
u_k^{(2)}
\end{pmatrix}.
\]  
(3.12)

The assumption that all control points and all observation points lie in the interior of the subdomains is only made to simplify the presentation and the initial implementation of domain decomposition algorithm. The approach does not require this assumption [12].
3.2.2 Schur Complement Reformulation

Now, let insert (3.8) - (3.12) back to (2.19) - (2.22) to obtain

\[
\begin{pmatrix}
M_1^T \\
M_2^T \\
M_T^T
\end{pmatrix}
\begin{pmatrix}
p_{k+1}^{(1)} \\
p_{k+1}^{(2)} \\
p^{(\Gamma)}_{k+1}
\end{pmatrix}
- \begin{pmatrix}
M_1^T \\
M_2^T \\
M_T^T
\end{pmatrix}
+ h \begin{pmatrix}
A_1^T & F_1^T \\
A_2^T & F_2^T \\
E_1^T & E_2^T & A_T^T
\end{pmatrix}
\begin{pmatrix}
p_k^{(1)} \\
p_k^{(2)} \\
p^{(\Gamma)}_k
\end{pmatrix}
+ \alpha_1 h \begin{pmatrix}
Q_1 & 0 & 0 \\
0 & Q_2 & 0 \\
0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
y_k^{(1)} \\
y_k^{(2)} \\
y_k^{(\Gamma)}
\end{pmatrix}
= \alpha_1 h \begin{pmatrix}
Q_1 & 0 & 0 \\
0 & Q_2 & 0 \\
0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
z_k^{(1)} \\
z_k^{(2)} \\
0
\end{pmatrix},
\]

\[k = 1, 2, ..., N - 1, \tag{3.13}\]

\[
- \begin{pmatrix}
M_1^T \\
M_2^T \\
M_T^T
\end{pmatrix}
+ h \begin{pmatrix}
A_1^T & E_1 \\
A_2^T & E_2 \\
F_1 & F_2 & A_T
\end{pmatrix}
\begin{pmatrix}
p_N^{(1)} \\
p_N^{(2)} \\
p^{(\Gamma)}_N
\end{pmatrix}
+ \alpha_1 h \begin{pmatrix}
Q_1 & 0 & 0 \\
0 & Q_2 & 0 \\
0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
y_N^{(1)} \\
y_N^{(2)} \\
y_N^{(\Gamma)}
\end{pmatrix}
= \alpha_1 h \begin{pmatrix}
Q_1 & 0 & 0 \\
0 & Q_2 & 0 \\
0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
z_N^{(1)} \\
z_N^{(2)} \\
0
\end{pmatrix}, \tag{3.14}
\]

\[
\alpha_2 \begin{pmatrix}
R_1 & 0 \\
0 & R_2
\end{pmatrix}
\begin{pmatrix}
u_k^{(1)} \\
u_k^{(2)}
\end{pmatrix}
- \begin{pmatrix}
B_1^T & 0 & 0 \\
0 & B_2^T & 0
\end{pmatrix}
\begin{pmatrix}
p_k^{(1)} \\
p_k^{(2)} \\
p^{(\Gamma)}_k
\end{pmatrix}
= 0, \quad k = 1, ..., N \tag{3.15}
\]
\[
\begin{pmatrix}
M_1 \\
M_2 \\
M_\Gamma
\end{pmatrix}
+ h \begin{pmatrix}
A_1 & E_1 \\
A_2 & E_2 \\
F_1 & F_2 & A_\Gamma
\end{pmatrix}
\begin{pmatrix}
y_k^{(1)} \\
y_k^{(2)} \\
y_k^{(\Gamma)}
\end{pmatrix}
- \begin{pmatrix}
M_1 \\
M_2 \\
M_\Gamma
\end{pmatrix}
\begin{pmatrix}
y_{k-1}^{(1)} \\
y_{k-1}^{(2)} \\
y_{k-1}^{(\Gamma)}
\end{pmatrix}
+ h \begin{pmatrix}
B_1 & \\
0 & B_2 \\
0 & 0
\end{pmatrix}
\begin{pmatrix}
u_k^{(1)} \\
u_k^{(2)}
\end{pmatrix}
= h \begin{pmatrix}
f_k^{(1)} \\
f_k^{(2)} \\
f_k^{(\Gamma)}
\end{pmatrix},
\]
\[k = 1, \ldots, N.\]  

(3.16)

If we only look at the first two rows of (3.13) - (3.16), it leads to

\[M_i^T p_{k+1}^{(i)} - (M_i + hA_i)^T p_{k}^{(i)} + \alpha_i hQ_{i} y_{k}^{(i)} = \alpha_i hQ_{i} z_{k}^{(i)} + hF_i^T p_{k}^{(\Gamma)}, \quad k = 1, \ldots, N - 1,\]

(3.17)

\[-(M_i + hA_i)^T p_{N}^{(i)} + \alpha_i hQ_{i} y_{N}^{(i)} = \alpha_i hQ_{i} z_{N}^{(i)} + hF_i^T p_{N}^{(\Gamma)},\]

(3.18)

\[\alpha_2 R_i u_{k}^{(i)} - B_i^T p_{k}^{(i)} = 0, \quad k = 1, \ldots, N,\]

(3.19)

\[(M_i + hA_i)y_{k}^{(i)} - M_i y_{k-1}^{(i)} + hB_i u_{k}^{(i)} = h f_k^{(i)} - hE_i y_{k}^{(\Gamma)}, \quad k = 1, \ldots, N.\]

(3.20)

If we only look at the last row of (3.13) - (3.16), we have the interface coupling conditions:

\[M_\Gamma^T p_{k+1}^{(\Gamma)} - (M_\Gamma + hA_\Gamma)^T p_{k}^{(\Gamma)} - hE_1^T p_{k}^{(1)} - hE_2^T p_{k}^{(2)} = 0, \quad k = 1, \ldots, N - 1,\]

(3.21)

\[-(M_\Gamma + hA_\Gamma)^T p_{N}^{(\Gamma)} - hE_1^T p_{N}^{(1)} - hE_2^T p_{N}^{(2)} = 0,\]

(3.22)

\[(M_\Gamma + hA_\Gamma)y_{k}^{(\Gamma)} - M_\Gamma y_{k-1}^{(\Gamma)} + hF_1 y_{k}^{(1)} + hF_2 y_{k}^{(2)} = h f_k^{(\Gamma)}, \quad k = 1, \ldots, N.\]

(3.23)

Here, we call the interface conditions (3.21) - (3.23) the **Schur Complement** problem.

In this chapter, we used the the domain decomposition idea reformulate the example problem as a Schur Complement problem coupled with interface conditions. The the following chapters, we will shall how to solve this problem.
Chapter 4

Model Reduction

In previous chapter, we convert an example problem into a Schur Complement problem. In order to solve this problem, we have to do many iterations of computation on each subdomain. Here, we introduce a model reduction technique to speed up the computation. This result greatly reduces the dimension of the subdomain boundary interface conditions.

In this chapter, we shall give a brief introduction of system theory and balanced truncation model reduction. Next, we show how to incorporate the model reduction technique into our Schur complement problem.

4.1 Introduction to Linear Dynamical Systems

Model reduction for linear dynamical systems consider the following systems:

\[
\Sigma = \begin{cases} 
\dot{x}(t) = Ax(t) + Bu(t) \\
y(t) = Cx(t) + Du(t)
\end{cases} \tag{4.1}
\]

or for simplicity:

\[
\Sigma := \begin{pmatrix} A & B \\ C & D \end{pmatrix} \tag{4.2}
\]
where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, $D \in \mathbb{R}^{p \times m}$ are given matrices, $m$ and $p$ are the number of inputs and outputs respectively, $u(t) \in \mathbb{R}^m$ is the given vector of inputs, $y(t) \in \mathbb{R}^p$ is the unknown vector of outputs, $x(t) \in \mathbb{R}^n$ is the unknown vector of internal variables, $n$ is the dimension of the system.

With some modifications, the definitions and results for continuous linear systems can be applied to discrete linear systems,

$$\sum \begin{cases} x(t + \Delta t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$$  \hspace{1cm} (4.3)

where $t \in \mathbb{R}$.

It is not difficult to get the solution of the differential equation (4.1),

$$x(t) = e^{(t-t_0)A}x(t_0) + \int_{t_0}^{t} e^{(t-\tau)A}Bu(\tau)d\tau.$$  \hspace{1cm} (4.4)

Thus, the output in time domain is

$$y(t) = Ce^{(t-t_0)A}x(t_0) + C\int_{t_0}^{t} e^{(t-\tau)A}Bu(\tau)d\tau + Du(t).$$  \hspace{1cm} (4.5)

Stability of the system is an important concept, see [4]. The system is stable if for any $\varepsilon > 0$, there exists $\delta > 0$ such that if $\|x(0)\| < \delta$ then $\|x(t)\| < \varepsilon$ for all $t > 0$. Otherwise the system is unstable. The origin system is asymptotically stable if it is stable, and if there exists $\delta' > 0$ such that if $\|x(0)\| < \delta'$ then $\lim_{t \to \infty} \|x(t)\| = 0$.

**Theorem 4.1.1** Linear system (4.1) is asymptotically stable $\iff$ all eigenvalues of $A$ have negative real parts.

Given two arbitrary vectors $x(t_0)$ and $x_f$ in state space, if there exists the input vector $u(t)$ that will take the initial state $x(t_0)$ to the final state $x(T) = x_f$, then the system will be called reachable. If for all possible known input and output vectors $u(t)$ and $y(t)$, we can determine the value of the initial state $x(t_0)$, then the system is called observable. You can use grammians to study the controllability and observability properties of state-space models and for model reduction. The controllability
grammian is defined by
\[ \mathcal{P} = \int_{0}^{\infty} e^{At}BB^{T}e^{At}dt \]  (4.6)
and the observability grammian by
\[ \mathcal{Q} = \int_{0}^{\infty} e^{At}C^{T}Ce^{At}dt. \]  (4.7)

Under assumption that the system is stable, it is well-known [3] [4] that the reachability grammian \( \mathcal{P} \) and observability grammian \( \mathcal{Q} \) are the unique solutions to Lyapunov equations (4.8) and (4.9), both \( \mathcal{P} \) and \( \mathcal{Q} \) are symmetric positive definite.

\[ A\mathcal{P} + \mathcal{P}A^{T} + BB^{T} = 0 \]  (4.8)
and
\[ A^{T}\mathcal{Q} + QA + C^{T}C = 0. \]  (4.9)

The cross grammian \( \mathcal{X} \) for square systems \( \Sigma \) \( (m = p) \) is defined as the solution \( \mathcal{X} \) of the sylvester equation
\[ AX + XA + BC = 0. \]  (4.10)
If \( A \) is stable, the solution of this equation can be represented as
\[ \mathcal{X} = \int_{0}^{\infty} e^{At}BCe^{At}dt. \]  (4.11)

**Proposition 4.1.2** The system is reachable if and only if \( \mathcal{P} \) is positive definite. The system is observable if and only if \( \mathcal{Q} \) is positive definite.

The unit impulse \( \delta : [0, \infty) \rightarrow \mathbb{R} \) is defined as \( \delta(t) = 0 \) for \( t \neq 0 \), and \( \int_{-\infty}^{\infty} \delta(t)dt = 1. \) By using the sifting property of impulse, if \( a < \tau < b \) and \( f(t) \) is continuous at \( t = \tau \), then
\[ \int_{a}^{b} f(t)\delta(t-\tau)dt = f(\tau). \]  (4.12)

By (4.12), equation (4.5) becomes
\[ y(t) = \int_{0}^{t} (Ce^{(t-\tau)}A + D\delta(t-\tau))u(t)d\tau 
\]
\[ = \int_{0}^{t} h(t-\tau)u(\tau)d\tau. \]  (4.14)
This has the form of a convolution integral, and \( h(t) \) is called the impulse response matrix, also called weighting matrix, which has the form

\[
h(t) = Ce^{tA}B + D\tau(t). \tag{4.15}
\]

The Hankel operator \( \mathcal{H} \) is defined as

\[
\mathcal{H} : \mathcal{L}^m(R_-) \hookrightarrow \mathcal{L}^p(R_+), \quad \mathcal{H}(u)(t) = \int_{-\infty}^{0} h(t - \tau)u(\tau)d\tau, \quad t \geq 0. \tag{4.16}
\]

Now we can define another two important quantities of the system.

**Definition 4.1.3** The Hankel singular values of the stable system \( \Sigma \), denoted by \( \sigma_1, \sigma_2, \ldots, \sigma_r \), are the singular values of the Hankel operator \( \mathcal{H} \) defined by (4.16). The Hankel norm of \( \Sigma \) is the largest singular value of \( \mathcal{H} \),

\[
\|\Sigma\|_{\mathcal{H}} = \sigma_{\max}(\mathcal{H}) = \sigma_1. \tag{4.17}
\]

**Theorem 4.1.4** [3] The Hankel singular values of a reachable, observable and stable linear system \( \Sigma \) are the square roots of the eigenvalues of the products of gramians \( \mathcal{P} \) and \( \mathcal{Q} \):

\[
\sigma_i = \sqrt{\lambda_i(\mathcal{P}\mathcal{Q})}, \quad i = 1, 2, \ldots, n. \tag{4.18}
\]

We close this section by presenting the following theorem.

**Theorem 4.1.5** Let \( \mathcal{P} \) and \( \mathcal{Q} \) denote the reachability gramian and the observability gramian of a stable linear system.

1. The minimal energy required to steer the state of the system from 0 to \( x_\tau \) is:

\[
x^*\mathcal{P}^{-1}x_\tau.
\]

2. The maximal energy produced by observing the output of the system whose initial state is \( x_0 \) is:

\[
x^*\mathcal{Q}x_0.
\]
3. The states which are difficult to reach (i.e. require large amounts of energy) are in the span of those eigenvectors of $\mathcal{P}$ which correspond to small eigenvalues.

4. The states which are difficult to observe (i.e. produce small amounts of energy) are in the span of those eigenvectors of $\mathcal{Q}$ which correspond to small eigenvalues.

Generally, [16] "Model reduction seeks to replace a large-scale dynamical system of differential or difference equations by a system of substantially lower dimension that has nearly the same response characteristics." There are many methods [3] to reduce the model order. Among them, the balanced truncation method has an excellent theory behind it. The balanced truncation method has two attractive features. The first is that it has a priori error bound. The second is that if the original system is stable, the reduced system will also be stable. This theorem reveals the basic idea of the balanced truncation model reduction which is to eliminate eliminating the states which are both difficult to reach and observe.

4.2 Balanced Truncation Method

The fundamental idea of model reduction is to produce a lower dimensional system of the same type

$$
\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{B}u(t),
\hat{y}(t) = \hat{C}\hat{x}(t) + Du(t),
$$

(4.19)

that will approximate the outputs of the original system (4.1), i.e. $\frac{\|y - \hat{y}\|}{\|u\|} \leq \varepsilon$.

The widely used model reduction approach for linear dynamical systems is balanced truncation which was introduced by [22].

From Theorem 4.1.5, we know that if $\mathcal{P} = \mathcal{Q} = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$, then a state which is difficult to reach is also difficult to observe. It is desirable to eliminate such states from the original system. The basic idea of balanced truncation is to find a realization for system (4.1) so that in the new coordinate system both $\mathcal{P}$ and $\mathcal{Q}$ are
diagonal and equal, then we can truncate the system by keeping r largest Hankel singular values.

**Definition 4.2.1** The stable system \( \Sigma \) is balanced if and only if \( \mathcal{P} = \mathcal{Q} \). \( \Sigma \) is principle-axis balanced if and only if

\[
\mathcal{P} = \mathcal{Q} = \Sigma := \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n).
\] (4.20)

In this thesis, we use the term balanced to mean principle-axis balanced for simplification. The balanced truncation problem is to find a balancing transformation matrix, then truncate the gramians. [3] lists several numerically stable implementations of balanced truncation method.

Balanced truncation method has a good error bound (see the following theorem). The \( \mathcal{H}_\infty \) norm of the error system is bounded by 2 times the sum of the neglected Hankel singular values. So in general, the reduced system has a good accuracy in approximating the original system.

**Theorem 4.2.2** [3] Suppose \( \sigma_i \), \( i = 1, 2, \ldots, n \) be the Hankel singular values of the system, then the \( \mathcal{H}_\infty \)-norm of the error system obtained from balanced truncation method is bounded by twice the summation of the neglected Hankel singular values,

\[
\| \Sigma - \Sigma_{\text{red}} \|_{\mathcal{H}_\infty} \leq 2(\sigma_{k+1} + \cdots + \sigma_n).
\] (4.21)

### 4.3 State Space Representation for the Full Problem

Our finite element semi-discretization of the optimal control problem (1.1)-(1.4) already gives the state space representation of the full problem,

\[
My'(t) = -Ay(t) - Bu(t) + f(t).
\] (4.22)
In the objective function (2.7), there is \((y(t) - z(t))^TQ(y(t) - z(t))\). Since \(Q\) is a semidefinite matrix, we can rewrite that part of the objective function (2.7) as

\[
(y(t) - z(t))^TQ(y(t) - z(t)) = (Q^{1/2}(y(t) - z(t)))^T(Q^{1/2}(y(t) - z(t)))
\]

\[
=\|Q^{1/2}y(t) - Q^{1/2}z(t)\|_2^2
\]

Thus, what we really want is \(Q^{1/2}y(t)\).

Now, the state space representation becomes

\[
\Sigma \begin{cases}
My'(t) = -Ay(t) - Bu(t) + f(t) \\
\dot{z}(t) = Q^{1/2}y(t)
\end{cases}
\]

4.4 State Space Representation for the Subproblem

By the domain decomposition method, we have reformed the optimality conditions of our example problem into the systems on the subdomains

\[
M_i^{T}p_{k+1}^{(i)} - (M_i + hA_i)^T p_k^{(i)} + \alpha_1 hQ_iy_k^{(i)} = \alpha_1 hQ_i z_k^{(i)} + hF_i^{T}p_{k}^{(T)},
\]

\[
-(M_i + hA_i)^{T} p_{N}^{(i)} + \alpha_1 hQ_iy_N^{(i)} = \alpha_1 hQ_i z_N^{(i)} + hF_i^{T}p_{N}^{(T)},
\]

\[
\alpha_2 R_i u_k^{(i)} - B_i^{T} p_k^{(i)} = 0
\]

\[
(M_i + hA_i)^{T} y_k^{(i)} - M_i y_{k-1}^{(i)} + hB_i u_k^{(i)} = h_{f_k}^{(i)} - hE_i y_k^{(T)}.
\]

and into the interface coupling condition

\[
M_{\Gamma}^{T}p_{k+1}^{(\Gamma)} - (M_{\Gamma} + hA_{\Gamma})^{T} p_k^{(\Gamma)} - hE_1^{T}p_k^{(1)} - hE_2^{T}p_k^{(2)} = 0,
\]

\[
-(M_{\Gamma} + hA_{\Gamma})^{T} p_{N}^{(\Gamma)} - hE_1^{T}p_N^{(1)} - hE_2^{T}p_N^{(2)} = 0,
\]

\[
(M_{\Gamma} + hA_{\Gamma})y_k^{(\Gamma)} - M_{\Gamma} y_{k-1}^{(\Gamma)} + hF_1 y_k^{(1)} + hF_2 y_k^{(2)} = h_{f_k}^{(\Gamma)}.
\]
The system on each subdomain is the discretized form of
\[ M_i^T \dot{p}^{(i)}(t) = A_i^T p^{(i)}(t) + E_i p^{(i)}(t) + \alpha_1 Q_i \left( z^{(i)}(t) - y^{(i)}(t) \right), \quad p^{(i)}(T) = 0, \tag{4.34} \]
\[ \alpha_2 R_i u^{(i)}(t) = B_i^T p^{(i)}(t), \tag{4.35} \]
\[ M_i \dot{y}^{(i)}(t) = -A_i y^{(i)}(t) - B_i u^{(i)}(t) - F_i y^{(s)}(t) + f^{(i)}(t), \quad y^{(i)}(0) = y_0^{(i)}. \tag{4.36} \]

This gives us that the system (4.34) - (4.36) can be split into the following two systems [16]:
\[ -M_i^T \dot{p}^{(i)}(t) = -A_i^T p^{(i)}(t) - E_i p^{(i)}(t) + Q_i^{\frac{1}{2}} (\alpha_1 Q_i^{\frac{1}{2}} y^{(i)}(t)) - \alpha_1 Q_i z^{(i)}(t), \tag{4.37} \]
\[ q^{(i)}(t) = -B_i^T p^{(i)}(t), \tag{4.38} \]
\[ r^{(i)}(t) = -F_i^T p^{(i)}(t), \tag{4.39} \]
\[ p^{(i)}(T) = 0, \tag{4.40} \]
and
\[ M_i \dot{y}^{(i)}(t) = -A_i y^{(i)}(t) - B_i u^{(i)}(t) - F_i y^{(s)}(t) + f^{(i)}(t), \tag{4.41} \]
\[ v^{(i)}(t) = -E_i^T y^{(i)}(t), \tag{4.42} \]
\[ w^{(i)}(t) = Q_i^{\frac{1}{2}} y^{(i)}(t), \tag{4.43} \]
\[ y^{(i)}(0) = y_0^{(i)}. \tag{4.44} \]

We view \( p^{(s)}(t), \alpha_1 Q_i^{\frac{1}{2}} y^{(i)}(t) \) as the inputs and \( q^{(i)}(t), r^{(i)}(t) \) as the outputs of the system (4.37) - (4.40). The state space representation of (4.37) - (4.40) is
\[
\Sigma_i^s = \left( \begin{array}{c|c}
-A_i^T & -E_i \\
-B_i^T & Q_i^{\frac{1}{2}} \\
-F_i & 0
\end{array} \right). \tag{4.45}
\]

We view \( u^{(i)}(t), y^{(s)}(t) \) as the inputs into the system (4.41) - (4.44) and \( v^{(i)}(t), w^{(i)}(t) \) as the outputs. The state space representation of (4.41) - (4.44) is
\[
\Sigma_i^s = \left( \begin{array}{c|c}
-A_i & -E_i \\
-B_i & -F_i^T \\
-E_i^T & Q_i^{\frac{1}{2}} \\
0 & 0
\end{array} \right). \tag{4.46}
\]
4.5 Model Reduction on Subdomains

Based on the discussion above in this section, we can apply balanced truncation technique to the system $\Sigma_i^{\Delta}$. That is to find the matrices $W$ and $V$ for the projection

\[ -\hat{A}_i^T = W^T (-A_i^T) V, \]  
\[ \begin{bmatrix} -\hat{E}_i & \hat{Q}_i \end{bmatrix} = W^T \begin{bmatrix} -E_i & Q_i^{\frac{1}{2}} \end{bmatrix}, \]  
\[ \begin{bmatrix} -\hat{B}_i^T \\ -\hat{F}_i \end{bmatrix} = \begin{bmatrix} -B_i^T \\ -F_i \end{bmatrix} V. \]  

This is the model reduction for $\Sigma_i^{\Delta}$. The reduced representation is

\[ \Sigma_i^s = \begin{pmatrix} -\hat{A}_i^T & -\hat{E}_i & \hat{Q}_i \\ -\hat{B}_i^T & 0 \end{pmatrix}. \]

Since $Q_i$ is symmetric positive definite, $Q_i^{\frac{1}{2}}$ is symmetric. Now, we automatically get a reduced representation for the system $\Sigma_i^s$.

\[ \hat{\Sigma}_i^s = \begin{pmatrix} -\hat{A}_i & -\hat{B}_i & -\hat{F}_i^T \\ -\hat{E}_i^T & 0 \end{pmatrix}. \]

In this chapter, we presented an introduction to model reduction. Then, we derived the reduced interface conditions for our Schur complement problem. Through a model reduction of the interface conditions, we could reduce the computational cost of the subdomain problems.
Chapter 5

Solution Algorithm

5.1 GMRES

Section 3.2.2 of this thesis has already given an outline of our solution algorithm. We present a preconditioned GMRES algorithm to solve the Schur complement problem (3.21) - (3.23).

Now, we introduce extraction matrix $I_T^i$. It is a matrix with zero or one entries that extracts out a vector (whole interface) the subvector (interface belongs to $i$th subdomain). Now the Schur Complement can be written as

$$\sum_{i=1}^{s} (I_T^i)^T S_i I_T^i (y^{(r)}, p^{(r)}) = r,$$

where $S_i$ is defined as

$$S_i((y^{(r)})^{(i)}, (p^{(r)})^{(i)}) = \begin{pmatrix}
((M_T^i) + h(A_T^i)) (y^{(r)})^{(i)} - M_T^i (y_0^{(r)})^{(i)} + hF_i y_1^{(i)} \\
(M_T^i)^T (p_2^{(r)})^{(i)} - ((M_T^i) + h(A_T^i))^T (p_1^{(r)})^{(i)} - hE_T^r p_1^{(i)} \\
\vdots \\
((M_T^i) + h(A_T^i))(y_N^{(r)})^{(i)} - M_T^i (y_{N-1}^{(r)})^{(i)} + hF_i y_N^{(i)} \\
((M_T^i) + h(A_T^i))^T (p_N^{(r)})^{(i)} - hE_T^r p_N^{(i)}
\end{pmatrix},$$

(5.2)
\((y_k^{(\Gamma)})^i\) and \((p_k^{(\Gamma)})^i\) are defined as

\[
(y_k^{(\Gamma)})^i = I_T^i y_k^{(\Gamma)}, \tag{5.3}
\]

\[
(p_k^{(\Gamma)})^i = I_T^i p_k^{(\Gamma)}. \tag{5.4}
\]

In (5.2), \(y^{(i)}\) and \(p^{(i)}\) solve the subdomain problem (3.17) - (3.20) for given \(y^{(\Gamma)}\) and \(p^{(\Gamma)}\). Now, the linear operator equation (5.1) can be solved by GMRES. The core step in each iteration is how to evaluate \((I_T^i)^T S_i I_T^i (y_{\Gamma}(t), p_{\Gamma}(t))\) for arbitrary \(y_{\Gamma}\) and \(p_{\Gamma}\). This can be done by

- solve subdomain problem for \(y^{(i)}\) and \(p^{(i)}\) on each subdomain;

- evaluate \((I_T^i)^T S_i I_T^i (y_{\Gamma}(t), p_{\Gamma}(t))\) by (5.2).

### 5.2 Neumann-Neumann Preconditioner

The above GMRES generally converges slowly. Usually, iterative methods are used in combination with some kind of preconditioning to improve the condition number or cluster the eigenvalues of the problem in order to achieve accelerated convergence. Here, we present a Neumann-Neumann preconditioner for the Schur complement problem.

We introduce \(D_{\Gamma}^i\) be positive diagonal matrices such that

\[
\sum_{i=1}^g (I_T^i)^T D_{\Gamma}^i = I. \tag{5.5}
\]

The preconditioner (3.1) is defined by approximating the inverse of the sum of local Schur Complement problems by the weighted sum of the inverses:

\[
P_{NN} = \sum_{i=1}^g (I_T^i)^T D_{\Gamma}^i S_i^{-1} D_{\Gamma}^i I_T^i. \tag{5.6}
\]

This NN preconditioner has very good properties with respect to solving PDE. In [25], it is reported that the number of iterations required by the NN algorithm is
bounded independently of the mesh diameter. However, it does depend on the PDE and relative sizes of the subdomains. The convergence of the Neumann-Neumann preconditioner for a small number of subdomains is quite good. The convergence rate decreases rapidly for a large number of subdomains.

The behavior of the NN preconditioner for our optimal control problems will be shown in the following section. For four subdomains cases, our tests show that most problems converge within 20 iterations.

One may note that applying $S_i^{-1}$ is the most computationally expensive part for the NN preconditioner. To overcome this, we might apply pseudoinverse. We could generate a coarse grid and define $\hat{S}_i$ on it as we did in (5.2). Then $S_i^{-1}$ could be approximated by $\hat{S}_i^{-1}$. This pseudoinverse idea will be investigated in the future.
Chapter 6

Numerical Results

In this section, we show the solutions of the optimal control problem (1.1) - (1.4) on a rectangle domain $\Omega = [0, 1] \times [0, 1]$.

6.1 Optimal Control Problem No.1

In this problem, $\Omega_c$ is defined by $\bigcup_{i=1}^4 \Omega_c^i$, where $\Omega_c^1 = (0, \frac{1}{8}) \times (0, \frac{1}{8})$, $\Omega_c^2 = (0, \frac{1}{8}) \times (\frac{7}{8}, 1)$, $\Omega_c^3 = (\frac{7}{8}, 1) \times (0, \frac{1}{8})$ and $\Omega_c^4 = (\frac{7}{8}, 1) \times (\frac{7}{8}, 1)$. $\Omega_o$ is defined by $\bigcup_{i=1}^4 \Omega_o^i$, where $\Omega_o^1 = (\frac{1}{4}, \frac{3}{8}) \times (\frac{1}{4}, \frac{3}{8})$, $\Omega_o^2 = (\frac{1}{4}, \frac{3}{8}) \times (\frac{5}{8}, \frac{3}{4})$, $\Omega_o^3 = (\frac{5}{8}, \frac{3}{4}) \times (\frac{1}{4}, \frac{3}{8})$ and $\Omega_o^4 = (\frac{5}{8}, \frac{3}{4}) \times (\frac{5}{8}, \frac{3}{4})$.

We set

$$f(x, t) = 0, \quad (6.1)$$

and compute $z$ as the solution of (1.2)-(1.4) with $u$ given by

$$u(x, t) = \begin{cases} \frac{1}{2\pi} t + 10\sin(t) & x \in \Omega_c^1, \\ \frac{1}{2\pi} t + 2\cos(t) & x \in \Omega_c^2, \\ 6\cos(t) & x \in \Omega_c^3, \\ 12\sin(t) & x \in \Omega_c^4. \end{cases} \quad (6.2)$$
6.1.1 Coarse Mesh

We discretize the spatial domain $\Omega$ into a coarse finite element mesh of 128 triangles, and split the mesh into four subdomains. Here, the time step is $\frac{\pi}{8}$, and the time interval is $[0, 4\pi]$.

With functions $f(x, t)$ and $u(x, t)$, we can calculate the exact observation. By using this observation, we solve the optimal control problem by using our domain decomposition algorithm.

Here, we show the controls and observations on each subdomain (by different color) in Fig 6.3. The solid lines are exact controls and responses. The dot lines are approximated controls and corresponding observations. In this figure, one can see that the control and observation were approximated very well. Next, let us see the error on both control and observation.

The error of our algorithm is given in Fig 6.4. From this figure, the relative error for both control and observation is very small.
Figure 6.2: Coarse finite element mesh on $\Omega$.

Figure 6.3: Controls and observations on coarse mesh.
Figure 6.4: Controls and observations on coarse mesh.

6.1.2 Fine Mesh

Now, we refine the mesh into a mesh with 512 triangles. Then, by using the same setting, we solve the problem again.

Here, we show the controls and observations on each subdomain (by different color) in Fig 6.6. The solid lines are exact controls and responses. The dot lines are approximated controls and corresponding observations. In this figure, one can see that the control and observation were approximated very well. Next, let us see the error on both control and observation.

The error of our algorithm is given in Fig 6.7. From this figure, the relative error for both control and observation is very small.

From this problem, one can see that our domain decomposition algorithm solves this optimal control problem with high accuracy.
Figure 6.5: fine finite element mesh on $\Omega$.

Figure 6.6: Controls and observations on fine mesh.
6.2 Optimal Control Problem No.2

In this problem, we still use the same $\Omega$, $\Omega_c$ and $\Omega_o$. The function $f(x, t)$ is given by

$$f(x, t) = \cos\left(\frac{\pi}{\sqrt{2}} d(x)\right) + \sin(t) + 1,$$  \hspace{1cm} (6.3)

where $d(x) = \|x - \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}\|_2$.

This time, there will be no control given. We want to control the observation by letting

$$z(x, t) = 1, \quad x \in \Omega_o.$$  \hspace{1cm} (6.4)

6.2.1 Coarse Mesh

Here, we still use the same coarse mesh as before. On this mesh, we show the function $f(x, t)$ at four particular time points.

Now, we can solve the optimal control problem by using $z(x, t) = 1$. In Fig 6.9,
we show the approximate control and corresponding observation. You can see that the approximate observation fit $z(x, t) = 1$ quite well.

### 6.2.2 Fine Mesh

Here, we still use the same fine mesh as before. On this mesh, we also show the function $f(x, t)$ at four particular time points.

Now, we can solve the optimal control problem by using $z(x, t) = 1$. In Fig 6.9, we show the approximate control and corresponding observation. You can see that the approximate observation fit $z(x, t) = 1$ quite well.
Figure 6.9: Approximate control and observation on coarse mesh.

Figure 6.10: $f(x, t)$ on fine mesh.
Figure 6.11: Approximate control and observation on fine mesh.
Chapter 7

Conclusions and Future Work

7.1 Summary and Conclusions

We presented a domain-decomposition approach for a class of optimal control problems. By using spatial domain decomposition method, the optimality conditions are decomposed into subproblems coupling with interface conditions. We used preconditioned GMRES method to solve the problems. The tests indicate that our approach is competitive to the standard methods for the problems with small number of controls.

For the large-scale problems, the storage requirement of the standard methods will be prohibitively large. Since our domain decomposition method only focuses on the interface state variables, the storage requirement only depends on the dimension of the Schur complement problem, which is much smaller than the dimension of the original problem. Through model reduction, the computational cost may be further reduced. Therefore, our domain decomposition method has the capability to handle large-scale problems.
7.2 Future work

Now, all the codes only work for two dimensional problems. The next thing to be done is the implementation and the integration of model reduction algorithm into our domain decomposition method. The long term objective is to solve 3-D problem, rewrite the codes in C/C++ (using some mathematical software packages, such as Trilinos) and implement them in parallel environment.
Bibliography


