Title
Methods for PDE-constrained optimization

Permalink
https://escholarship.org/uc/item/287325bn

Author
Reed, Joseph Robert

Publication Date
2011

Peer reviewed|Thesis/dissertation
UNIVERSITY OF CALIFORNIA, SAN DIEGO

Methods for PDE-Constrained Optimization

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Joseph Robert Reed

Committee in charge:
Professor Philip E. Gill, Chair
Professor Randolph E. Bank
Professor Thomas Bewley
Professor Robert Bitmead
Professor Bo Li

2011
The dissertation of Joseph Robert Reed is approved, and it is acceptable in quality and form for publication on microfilm:

________________________________________

________________________________________

________________________________________

________________________________________

________________________________________

Chair

University of California, San Diego

2011
# TABLE OF CONTENTS

Signature Page .......................................................... iii

Table of Contents ....................................................... iv

List of Figures ........................................................... vii

List of Tables ............................................................ viii

Acknowledgements ........................................................ ix

Vita and Publications ................................................... x

Abstract .................................................................. xi

## Chapter 1 Introduction ............................................... 1
1.1 Overview ................................................................. 1
1.2 Notation and terminology ........................................... 1
1.3 Optimality conditions ............................................... 2
1.4 PDE-constrained optimization ..................................... 4
   1.4.1 Discretize-optimize ............................................. 5
   1.4.2 Problem structure ............................................... 7
1.5 Interior-point methods .............................................. 7
   1.5.1 Barrier methods .................................................. 8
   1.5.2 Inexact Newton methods ..................................... 9
   1.5.3 Merit functions .................................................. 10
   1.5.4 Line-search methods ......................................... 10
   1.5.5 Trust-region methods ......................................... 12
   1.5.6 Combining trust-region and line-search methods .......... 15
1.6 Other research ........................................................ 16
   1.6.1 Augmented Lagrangian methods ........................... 17
1.7 Contributions of this thesis ......................................... 17
1.8 Outline ................................................................ 18
1.9 Useful Results ........................................................ 19

## Chapter 2 Interior methods ......................................... 20
2.1 Overview ................................................................. 20
2.2 Background on barrier-SQP methods ............................ 20
2.3 A primal-dual augmented Lagrangian ............................. 22
   2.3.1 Scaling in the PDE context ................................. 23
2.4 Doubly-augmented systems ......................................... 25
2.5 Primal-dual augmented Lagrangian methods ................... 26
   2.5.1 Regularized primal-dual SQP method ....................... 26
   2.5.2 Using an approximate Hessian ............................... 27
   2.5.3 The inner-iterations ........................................... 28
LIST OF FIGURES

Figure 6.1: State variable predicted by PLTMG for Problem 1 . . . . . . . . . . . . 84
Figure 6.2: Control variable predicted by PLTMG for Problem 1 . . . . . . . . . . . . 84
Figure 6.3: Lagrange multiplier predicted by PLTMG for Problem 1 . . . . . . . . . . 85
Figure 6.4: State variable predicted by PLTMG for Problem 2 . . . . . . . . . . . . 85
Figure 6.5: Control variable predicted by PLTMG for Problem 2 . . . . . . . . . . . . 86
Figure 6.6: Lagrange multiplier predicted by PLTMG for Problem 2 . . . . . . . . . . 86
Figure 6.7: State variable predicted by PLTMG for Problem 3 . . . . . . . . . . . . 87
Figure 6.8: Control variable predicted by PLTMG for Problem 3 . . . . . . . . . . . . 87
Figure 6.9: Lagrange multiplier predicted by PLTMG for Problem 3 . . . . . . . . . . 88
LIST OF TABLES

Table 6.1: Problem 1 preconditioner results ........................................ 79
Table 6.2: Problem 2 preconditioner results ........................................ 79
Table 6.3: Problem 3 preconditioner results ........................................ 80
Table 6.4: PLTMG results ................................................................. 83
Table 6.5: Optimization results for Problem 1 ..................................... 83
Table 6.6: Optimization results for Problem 2 ..................................... 83
ACKNOWLEDGEMENTS

After writing all of this technical material, I thought that acknowledging those that helped me along the way would be easy. The truth is, I hardly know where to begin. Words cannot express how grateful I am to my advisor, Dr. Philip E. Gill, for his patience and encouragement during my time at UCSD. I came into graduate school knowing very little applied mathematics. The fact that I am writing this acknowledgement is due in large part to his ability to express the beauty and power of optimization in everyday life. I would also like to thank Dr. Randy E. Bank for his efforts in explaining the inner-workings of PLTMG and his keen insight into partial differential equations and the finite-element method. I could not have gone through this process without the remaining members of my committee: Dr. Bo Li, Dr. Thomas Bewley, and Dr. Robert Bitmead.

This dissertation could not have been written without the financial support of the UCSD Department of Mathematics. I would like to thank Lois Stewart for guiding me through the final steps of the graduation process. I would like to thank Michael Ferry and James Hall for putting up with me as an office mate and Jeremy Greene for suggesting I take MATH 271.

I would not be where I am today without the love and support of my family and friends. Thank you Mom, Dad, Tim, Grandma Burke, and Jason. I do not deserve what you have given me. I would like to dedicate this dissertation to my wife Erica and daughter Ronia. We have all gone through quite a bit the last few months. I will never be able to thank you enough for your patience and support during this long process. All I can say is that I will be the best husband and father I can be. Erica, you add a spark to my life that I cannot describe. I love you and am excited to start our next big adventure.
<table>
<thead>
<tr>
<th>Year</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2006</td>
<td>B. S. in Mathematics <em>cum laude</em>, California Polytechnic State University, San Luis Obispo</td>
</tr>
<tr>
<td>2006-2011</td>
<td>Graduate Teaching Assistant, University of California, San Diego</td>
</tr>
<tr>
<td>2011</td>
<td>Ph. D. in Mathematics, University of California, San Diego</td>
</tr>
</tbody>
</table>
ABSTRACT OF THE DISSERTATION

Methods for PDE-Constrained Optimization

by

Joseph Robert Reed
Doctor of Philosophy in Mathematics
University of California San Diego, 2011

Professor Philip E. Gill, Chair

The KKT systems arising in nonlinearly constrained optimization problems may not have correct inertia, and therefore must be modified to avoid convergence to nonoptimal KKT points. Matrix factorizations can determine the inertia of a general symmetric matrix but are too costly in the PDE context.

In PDE-constrained optimization, KKT systems are generally solved with preconditioned iterative methods that are unable to detect whether the current matrix has correct inertia. Moreover, the preconditioners assume the existence of a preconditioner for the underlying PDE.

Methods are discussed that solve the constrained problem by minimizing a sequence of smooth primal-dual merit functions. The Newton equations are solved approximately with a variant of the preconditioned conjugate-gradient (PCG) method that naturally determines when the regularized KKT system for the constrained problem has incorrect inertia. Convergence is accelerated with a sparsity exploiting preconditioner that implicitly defines a positive-definite system.

The preconditioning strategy is entirely algebraic and is based on an incomplete factorization of an equivalent symmetric indefinite system. It explicitly takes advantage of dual regularization, and in the PDE constrained context, does not require a preconditioner for the underlying PDE.
Chapter 1

Introduction

1.1 Overview

The focus of this dissertation is on primal interior-point methods for nonlinearly constrained optimization problems of the form:

$$\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad c_i(x) = 0, \quad i \in \mathcal{E} \\
& \quad c_i(x) \geq 0, \quad i \in \mathcal{I},
\end{align*}$$

(1.1)

where \( c(x) \) is an \( m \)-vector of constraint functions with \( i \)-th component \( c_i(x) \), and \( \mathcal{E} \) and \( \mathcal{I} \) form a disjoint partition of the index set \( \{1, 2, \ldots, m\} \) with \( |\mathcal{E}| = m_\mathcal{E} \) and \( |\mathcal{I}| = m_\mathcal{I} \). It is assumed that the objective function \( f : \mathbb{R}^n \mapsto \mathbb{R} \) and constraint functions \( c_i : \mathbb{R}^n \mapsto \mathbb{R} \) are twice-continuously differentiable. Moreover, the inequality constraints are assumed to be linear and define finite bound constraints on a subset of the variables \( x \).

1.2 Notation and terminology

The \( i \)-th component of an arbitrary vector \( v \) is denoted \([v]_i\) or \( v_i \), depending on the context. The \( ij \)-th component of an arbitrary matrix \( M \) is denoted \([M]_{ij}\), or \( m_{ij} \). The inertia of a symmetric matrix \( K \) is the integer triple \( \text{In}(K) = (a_+, a_-, a_0) \), giving the number of positive, negative, and zero eigenvalues. The vectors \( e \) and \( e_j \) denote the column vector of ones and the \( j \)-th column of the identity matrix \( I \). The dimensions of \( e, e_j, \) and \( I \) are defined by the context. Given two vectors, \( a \) and \( b \), of the same length, the \( i \)-th component of the vector \( a \cdot b \) is \( a_i b_i \).
The quantities $g(x)$, $H(x)$, and $J(x)$ denote the gradient $\nabla f(x)$, Hessian $\nabla^2 f(x)$, and Jacobian $c'(x)$. The Hessian of $c_i(x)$ is denoted $H_i(x)$. It is often the case that these functions must be evaluated at a specific point $x_k$, and in this case the following notation may be used: $f_k = f(x_k)$, $g_k = g(x_k)$, $c_k = c(x_k)$, and $J_k = J(x_k)$. In some situations, in will make sense to shorten this notation further to $f$, $g$, $c$, and $J$, respectively. For convenience the vector of equality and inequality constraints may be denoted $c_E(x)$ and $c_I(x)$. In the equality-constrained context, the quantities $m_E$, $c_E(x)$, and $J_E(x)$ are denoted by $m$, $c(x)$, and $J(x)$. The Jacobian of the inequality constraints is always denoted by $J_I$.

The algorithms described here use a two-level structure of “major” and “minor” iterates (also called “outer” and “inner” iterates), where the outer iterates are generated from a subproblem solved by an iterative procedure. Unless stated otherwise, a local solution to problem (1.1) is denoted by $x^*$ and $\{x_k\}$ denotes a sequence of subproblem solutions intended to approach $x^*$. Each $x_k$ is a member of a sequence of inner-iterates $\{x_j\}$. This notation will be used with respect to other two-level sequences as well. Vector pairs or triples are occasionally abbreviated by $v$, so, for instance,

$$v \triangleq (x, \pi), \quad v_k^* \triangleq (x_k^*, \pi_k^*), \quad \text{and} \quad v^* \triangleq (x^*, \pi^*).$$

A vector $y$ is considered $O(\mu)$ (denoted $y = O(\mu)$) for $\mu > 0$ if there exists a constant $\kappa > 0$ independent of $\mu$ such that $\|y\| \leq \kappa \mu$, with respect to some norm.

### 1.3 Optimality conditions

The Lagrangian function for problem (1.1) is defined as

$$L(x, \pi, z) = f(x) - c_E(x)^T \pi - c_I(x)^T z, \quad (1.2)$$

where $x$ is the vector of primal-variables, $\pi \in \mathbb{R}^m$ is the vector of Lagrange multipliers, and $z \in \mathbb{R}^{m_I}$ is the vector of reduced-costs. Because the inequality constraints are linear, the Hessian of the Lagrangian with respect to $x$ is

$$H(x, \pi) = H(x) - \sum_{i \in E} \pi_i H_i(x). \quad (1.3)$$

In general, finding a global minimizer of problem (1.1) is NP-hard. Because of this, most algorithms in smooth optimization attempt to satisfy problem-dependent optimality conditions that hold at a local minimizer. When describing optimality conditions for problem (1.1), a particularly important quantity is the active set defined at a point feasible with respect to the inequalities.
Definition 1.3.1 (Active set). Let \( x \) satisfy \( c_I(x) \geq 0 \). The active set is defined as

\[
A_I(x) = \{ i \in I : c_i(x) = 0 \}.
\]

The matrix of rows of \( J_I \) corresponding to the active constraints is denoted \( J_{AI} \), and the matrix of active constraint gradients at \( x \) is

\[
J_A = \begin{pmatrix} J_{AI} \\ J_E(x) \end{pmatrix}.
\]

Optimality conditions are intimately tied to the following first-order Karush-Kuhn-Tucker (KKT) condition,

Definition 1.3.2 (First-order KKT point). \((x^*, \pi^*, z^*)\) is a first-order KKT point for (1.1) if the following conditions hold:

\[
c_E(x^*) = 0, \quad c_I(x^*) \geq 0 \tag{1.4}
\]

\[
g(x^*) = J_E(x^*)^T \pi^* + J_I^T z^* \tag{1.5}
\]

\[
z^* \geq 0 \tag{1.6}
\]

\[
z^* \cdot c_I(x^*) = 0. \tag{1.7}
\]

Additional regularity assumptions are necessary to ensure that first-order KKT conditions and local minimizers are related. These assumptions are referred to as constraint qualifications. The two most common constraint qualifications are the linear independence constraint qualification (LICQ), and the Mangasarian-Fromovitz constraint qualification (MFCQ) [48].

Definition 1.3.3 (LICQ). The LICQ holds at a feasible point \( x \) if \( J_A(x) \) has full row rank.

Definition 1.3.4 (MFCQ). The MFCQ holds at a feasible point \( x \) if \( J_E(x) \) has full row rank and there is a vector \( p \in \text{null}(J_E(x)) \) such that \( J_{AI} p \geq 0 \).

First-and-second order optimality conditions can now be stated.

Theorem 1.3.1 (First-order necessary conditions). If \( x^* \) is a local minimizer for which the MFCQ holds, then there exist \( \pi^* \) and \( z^* \) such that \((x^*, \pi^*, z^*)\) is a first-order KKT point.

Theorem 1.3.2 (Second-order necessary conditions). If \( x^* \) is a local minimizer for which the LICQ holds, then there exist \( \pi^* \) and \( z^* \) such that \((x^*, \pi^*, z^*)\) is a first-order KKT point and \( p^T H(x^*, \pi^*) p \geq 0 \) for all directions \( p \in \text{null}(J_E(x^*)) \) with \( J_{AI} p \geq 0 \).
Theorem 1.3.3 (Second-order sufficient conditions). Suppose the MFCQ holds at \( x^* \) and that \( (x^*, \pi^*, z^*) \) is a first-order KKT point. If there exists \( \omega > 0 \) such that \( p^T H(x^*, \pi^*) p \geq \omega \| p \|^2 \) for all \( p \in \text{null}(J_E(x^*)) \) and \( J_A p \geq 0 \), then \( x^* \) is a local minimizer.

Note that if \( A(x^*) = \emptyset \), then the conditions for the Hessian in the second-order conditions can be replaced by the condition that the reduced-Hessian defined as

\[
H_Z(x^*, \pi^*) = Z(x^*)^T H(x^*, \pi^*) Z(x^*),
\]

(1.8)
is positive definite, where \( Z(x^*) \) is the \( n \times (m - n) \) matrix whose columns form a basis for \( \text{null}(J(x^*)) \). Before discussing optimization algorithms, the previous conditions will be framed in the PDE constrained optimization context.

1.4 PDE-constrained optimization

Of particular interest are problems with constraints that come from a spatial discretization of a partial differential equation (PDE) on a domain \( \Omega \) with boundary \( \partial \Omega \). Here, it is assumed that \( \Omega \subset \mathbb{R}^2 \). These problems may be formulated in a finite-dimensional setting, compatible with problem (1.1).

The classical formulation of a PDE-constrained optimization with point-wise control constraints takes the form:

\[
\begin{align*}
\text{minimize} & \quad f(y, u) \\
\text{subject to} & \quad C(y, u) = 0 \quad \text{on} \quad \Omega \subset \mathbb{R}^2 \\
& \quad b_l \leq u \leq b_u.
\end{align*}
\]

(1.9)
The variable \( y \) belongs to a space of real-valued functions defined on \( \Omega \), and is referred to as the state variable. Depending on the problem, \( u \) may belong to a function space, or may even be a vector of parameters. Either way, it will be referred to as the control variable. The space of state and control variables is denoted \( \mathcal{Y} \) and \( \mathcal{U} \), and the state equation \( C : \mathcal{Y} \times \mathcal{U} \to \mathbb{R} \) defines the PDE. The boundary conditions are implicit in the state equation and are assumed to be of Dirichlet, Neumann, or mixed type on all or some portion of \( \partial \Omega \). The objective function is assumed quadratic in the state and control, and takes the form:

\[
f(y, u) = \frac{1}{2} \| y - y_t \|^2_{\mathcal{Y}} + \frac{\eta}{2} \| u \|^2_{\mathcal{U}},
\]

(1.10)
where \( y_t \in \mathcal{Y} \) is considered the “true” solution, and \( \eta (\eta > 0) \) is a regularization parameter that keeps the problem well-posed. The norms defining (1.10) are generally induced from
inner-products,
\[ (\cdot, \cdot)_Y : Y \times Y \mapsto \mathbb{R} \quad \text{and} \quad (\cdot, \cdot)_U : U \times U \mapsto \mathbb{R}. \]  
(1.11)

As an example of a state equation, consider the Poisson problem
\[ \nabla^2 y = 0 \quad \text{on} \quad \Omega \quad \text{subject to} \quad y = u \quad \text{on} \quad \partial \Omega, \]  
(1.12)

where the closure of \( \Omega \subset \mathbb{R}^2 \) is the closed unit disk and \( y : \partial \Omega \mapsto \mathbb{R} \) is square-integrable. Then \( \mathcal{C}(\hat{y}, \hat{u}) = 0 \) means that \( \hat{y} \) solves (1.12) with \( u = \hat{u} \). Therefore, the state variable is the solution to the state equation, parameterized by the control variable. An algorithm for PDE-constrained optimization “tunes” the control variable within its allowable range until the state variable gets as close to \( y_t \) as possible.

1.4.1 Discretize-optimize

The first step in expressing problem (1.9) as a finite-dimensional nonlinear program is to “weaken” the state equation, giving a set of variational equalities. The process used to put the PDE into a variational form depends on the problem. As an example, consider the boundary value problem
\[ \nabla^2 y = u_1 \quad \text{subject to} \quad y = u_d \quad \text{on} \quad \Gamma_d, \quad \partial_n y = u_n \quad \text{on} \quad \Gamma_n, \]  
(1.13)

where \( \partial \Omega = \Gamma_n \cup \Gamma_d \) is a decomposition of the boundary into its Neumann and Dirichlet components. The operator \( \partial_n \) denotes the normal-derivative. The control variable is selected from \( \{u_1, u_d, u_n\} \), and will not be specified as it does not affect the computation. To proceed further, consider the Sobolev space \( H^1(\Omega) \) defined as
\[ H^1(\Omega) = \left\{ \varphi : \Omega \mapsto \mathbb{R} : \varphi, \varphi' \in L^2(\Omega) \right\}, \]
where \( \varphi' \) is the generalized derivative of \( \varphi \). The solution, \( y^* \), to (1.13) also satisfies the variational equation
\[ c_E(y^*, u; \varphi) \triangleq a(y^*, \varphi) - \langle u_n, \varphi \rangle + (u_1, \varphi) = 0, \quad \text{for all} \quad \varphi \in H^1_{0, \Gamma_d}(\Omega), \]  
(1.14)

where in the current context,
\[ H^1_{0, \Gamma_d}(\Omega) = \left\{ \varphi \in H^1(\Omega) : \varphi = 0 \quad \text{on} \quad \Gamma_d \right\}, \]
and
\[ a(\phi, \varphi) = \int_\Omega \nabla \phi \cdot \nabla \varphi, \quad (u_n, \varphi) = \int_{\Gamma_n} u_n \varphi, \quad \text{and} \quad (u_1, \varphi) = \int_\Omega u_1 \varphi. \]  
(1.15)
In (1.14), \( H_{0,R_d}^1(\Omega) \) is referred to as the space of test functions. Suppose that \( y^* \) satisfies (1.13) and consider the decomposition:

\[ y^* = y_0 + u_d, \quad \text{where} \quad y_0 \in H_{0,R_d}^1(\Omega). \]  \hfill (1.16)

This leads to the Galerkin method for computing \( y^* \): find \( y_0 \in H_{0,R_d}^1(\Omega) \) satisfying

\[ c_E(y_0 + u_d, u; \varphi) = 0 \quad \text{for all} \quad \varphi \in H_{0,R_d}^1(\Omega). \]  \hfill (1.17)

The Galerkin problem is discretized by requiring that (1.17) holds for a finite-dimensional subspace of the space of test functions. In the finite-element method, the subspace is constructed as a span of compactly supported basis functions derived from a spatial discretization of \( \Omega \). Suppose the discrete test space, denoted by \( V_h \), consists of \( m_E \) basis functions \( V_h = \{ \varphi_i \}_{i=1}^{m_E} \). Then the discrete Galerkin problem is the following: find \( y_0 \in V_h \) such that

\[ c(y_0 + u_d, u; \varphi_i) = 0 \quad \text{for} \quad i = 1 : m_E. \]  \hfill (1.18)

If the control variable comes from a function space, it can also be “projected” onto a finite-dimensional subspace of the control space denoted by \( U_h = \{ \phi_i \}_{i=1}^{n_u} \). Expressing \( y_0 \in V_h \) and \( u \in U_h \) as a linear combination of basis functions leads to the vector-valued function \( c_E : \mathbb{R}^n \rightarrow \mathbb{R}^{m_E} \) with \( i \)-th component:

\[ c_i(x) \triangleq c(y, u; \varphi_i) \quad \text{for} \quad i \in \mathcal{E}, \]  \hfill (1.19)

where \( x = (x_y, x_u) \),

\[ y = \sum_{i=1}^{n_y} [x_y]_i \varphi_i, \quad \text{and} \quad u = \sum_{i=1}^{n_u} [x_u]_i \phi_j. \]  \hfill (1.20)

In this case \( m_E = n_y \). This technique of expressing the states and controls as a linear combination of basis functions leads to an objective function of the form:

\[ f(x) = \frac{1}{2}(x_y - x_t)^T M_y (x_y - x_t) + \frac{\eta}{2} x_u^T M_u u, \]  \hfill (1.21)

where \( x_t \in \mathbb{R}^{n_y} \) is the vector of coefficients defining \( y_t \). The matrices \( M_y \) and \( M_u \) are symmetric, positive definite mass matrices defined component-wise by,

\[ [M_y]_{ij} = (\varphi_i, \varphi_j)_y \quad \text{and} \quad [M_u]_{ij} = (\phi_i, \phi_j)_{ut}. \]

Point-wise control constraints can also be expanded in terms of the control-space basis, giving the finite-dimensional problem

\[
\begin{align*}
\text{minimize} & \quad f(x_y, x_u) \\
\text{subject to} & \quad c_E(x_y, x_u) = 0 \\
& \quad c_I(x_y, x_u) \geq 0,
\end{align*}
\]  \hfill (1.22)
where the vector of inequality constraints is
\[ c_I(x_y, x_u) = \begin{pmatrix} 0 & I \\ 0 & -I \end{pmatrix} \begin{pmatrix} x_y \\ x_u \end{pmatrix} - \begin{pmatrix} \ell \\ -u \end{pmatrix}. \]

### 1.4.2 Problem structure

In a slight abuse of terminology, the components of \(x_y\) and \(x_u\) are referred to as state and control variables. Partitioning \(x\) is this way yields a natural partitioning of \(g(x)\), \(H(x)\), \(H_i(x)\), and \(H(x, \pi)\). The gradient and Hessian are
\[
g(x) = \begin{pmatrix} M_y(x_y - x_t) \\ \eta M_u x_u \end{pmatrix} \quad \text{and} \quad H(x) = \begin{pmatrix} M_y & 0 \\ 0 & \eta M_u \end{pmatrix}.
\]
The Jacobian of \(c_E\) is written,
\[
J_E(x) = \begin{pmatrix} J_y(x) \\ J_u(x) \end{pmatrix},
\]
where the subscripts \(y\) and \(u\) represent derivatives with respect to the state and control variables. The Jacobian of \(c_I\) may be partitioned in a similar way. If the bound constraints apply only to the control variables, then
\[
J(x) = \begin{pmatrix} J_y(x) & J_u(x) \\ 0 & J_I \end{pmatrix}.
\]
Similarly, the Hessian of the \(i\)-th equality constraint and Hessian of the Lagrangian with respect to \(x\) can be partitioned as
\[
H_i(x) = \begin{pmatrix} [H_i]_{yy}(x) & [H_i]_{yu}(x) \\ [H_i]_{uy}(x) & [H_i]_{uu}(x) \end{pmatrix} \quad \text{and} \quad H(x, \pi) = \begin{pmatrix} H_{yy}(x, \pi) & H_{yu}(x, \pi) \\ H_{uy}(x, \pi) & H_{uu}(x, \pi) \end{pmatrix}.
\]
Because the problem in function space is discretized with the finite-element method, \(J_y(x)\) and \(H_{yy}(x, \pi)\), are sparse matrices with sparsity pattern governed by the finite-element mesh. When \(u\) is a function, the matrices \(J_u(x), H_{yu}(x, \pi)\) and \(H_{uu}(x, \pi)\) are also sparse.

### 1.5 Interior-point methods

In this section we develop interior-point methods in the context of PDE constrained optimization. To begin, observe that finding a KKT point is equivalent to finding \((x^*, \pi^*, z^*)\)
satisfying $c_I(x^*) > 0$, $z^* > 0$, and

$$F(x^*, \pi^*, z^*) = \begin{pmatrix} g(x^*) - J_\mathcal{E}(x^*)^T \pi^* - J_\mathcal{T}^T z^* \\ c(x^*) \\ z^* \cdot c_I(x^*) \end{pmatrix} = 0. \quad (1.23)$$

Primal-dual interior-point methods find $(x^*, \pi^*, z^*)$ as a limit of iterates $\{ (x_k, \pi_k, z_k) \}$ such that $c_I(x_k) > 0$ and $z_k > 0$. The $k$-th member of this sequence is obtained by fixing $\mu_k > 0$ and applying Newton’s method to the perturbed optimality conditions

$$F_{\mu_k}(x, \pi, z) = \begin{pmatrix} g(x) - J_\mathcal{E}(x)^T \pi - J_\mathcal{T}^T z \\ c(x) \\ z \cdot c_I(x) - \mu_k w \end{pmatrix} = 0, \quad (1.24)$$

where $w \in \mathbb{R}^m$ is a fixed vector of weights. Applying Newton’s method to (1.24) introduces a sequence of minor iterates. The exact solution to (1.24) is denoted by $(x(\mu_k), \pi(\mu_k), z(\mu_k))$ and lies on the so-called central path. Interior-point methods attempt to follow the central path over a sequence of barrier parameters $\{ \mu_k \}$ converging to 0. Under certain conditions, for sufficiently small $\bar{\mu}$, the central path is a differentiable function on $(0, \bar{\mu})$ with

$$\lim_{\mu \to 0^+} (x(\mu), \pi(\mu), z(\mu)) \to (x^*, \pi^*, z^*).$$

Therefore, the sequence of interior-point iterates will converge to $(x^*, \pi^*, z^*)$ if they remain sufficiently close to the central path.

### 1.5.1 Barrier methods

Rather than work with the perturbed KKT conditions (1.24), the methods proposed here approximately solve a sequence of equality-constrained barrier subproblems

$$\min_{x \in \mathbb{R}^n} f_\mu(x) \overset{\triangle}{=} f(x) - \mu \sum_{i \in I} w_i \ln(c_i(x)) \quad (1.25)$$

subject to $c_\mathcal{E}(x) = 0$.

The barrier sequential quadratic programming (SQP) approach for problem (1.25) applies Newton’s method to the first-order optimality conditions. The Newton equations come from the stationarity conditions for the barrier Lagrangian,

$$L_\mu(x, \pi) = f_\mu(x) - c_\mathcal{E}(x)^T \pi. \quad (1.26)$$
If $S$ and $W$ are defined as $\text{diag}(c_I(x))$ and $\text{diag}(w)$, the gradient of $f_\mu$, and the Hessian of $L_\mu$ with respect to $x$ are
\[
g_\mu(x) = g(x) - \mu J_T^T S^{-1} w \quad \text{and} \quad H_\mu(x, \pi) = H(x, \pi) + \mu J_T^T S^{-2} W J_T.
\]
The symmetrized Newton equations for the zero-finding problem
\[
F_\mu(x, \pi) = \begin{pmatrix} g_\mu(x) - J_\mathcal{E}(x)^T \pi \\ c_\mathcal{E}(x) \end{pmatrix} = 0
\]
may be written as
\[
\begin{pmatrix} H_\mu(x, \pi) & J_\mathcal{E}(x)^T \\ J_\mathcal{E}(x) & -D \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta \pi \end{pmatrix} = - \begin{pmatrix} g_\mu(x) - J_\mathcal{E}(x)^T \pi \\ c_\mathcal{E}(x) \end{pmatrix},
\]
with $D = 0$. The matrix associated with this linear system is called a KKT, or saddle-point matrix and will be denoted by $K_\mu(x, \pi)$ in the barrier SQP case. If $\text{In}(K_\mu(x, \pi)) = (n,m,0)$, then $K_\mu(x, \pi)$ is said to have correct inertia.

1.5.2 Inexact Newton methods

In the large-scale setting, solving the Newton equations (1.27) exactly over a sequence $\{(x_j, \pi_j)\}$ is prohibitively expensive and may not be efficient when far from the solution. The inexact Newton method of Dembo, Eisenstat and Steihaug [10] computes an approximate solution to the Newton equations. The $j$-th step of an inexact Newton for finding a KKT point of the barrier subproblem computes $s_j$ such that
\[
F'_\mu(v_j) s_j = -F_\mu(v_j) + r_j, \quad \text{where} \quad \frac{\| r_j \|}{\| F_\mu(v_j) \|} \leq \tau_j
\]
and the elements $\{\tau_j\}$ define a forcing sequence designed to control the rate of convergence. Given $s_j$, the $(j+1)$-st estimate for the KKT point of problem (1.25) is $v_{j+1} = v_j + s_j$. The following theorem describes how the forcing sequence influences the convergence rate of $\{v_j\}$ to $v^*$.

Theorem 1.5.1. Assuming that the inexact Newton iterates $\{v_j\}$ converge to a KKT point $v_j^*$, then $v_j \to v^*$ at a superlinear rate if $\tau_j \to 0$. 

When iterative solvers are used to compute $s_k$, inexact Newton methods also carry the designation truncated Newton methods. If the inertia of $K_\mu(x_k, \pi_k)$ is correct, then any Krylov based solver can be used, including MINRES, SYMMLQ, and LSQR [53, 55, 54]. These algorithms are intended for solving symmetric indefinite systems and cannot be used to deal with nonconvex optimization problems. The Hestenes-Steifel conjugate-gradient (CG) [41] method is capable of detecting indefiniteness. A large part of this research develops methods that apply this positive definite solver to indefinite equations.

1.5.3 Merit functions

Newton’s method is not globally convergent. Global convergence of $\{v_k\}$ can be achieved when an appropriate merit function is used. Merit functions offer a way of applying tools from unconstrained optimization to the constrained setting.

In the barrier SQP context, a merit function measures the progress being made in finding a first-order KKT point for a given barrier subproblem. Perhaps the most obvious merit function in this case is

$$
M_\mu(v) = \frac{1}{2} \| F_\mu(x, \pi) \|^2.
$$

This is considered a first-order merit function because it is does not distinguish between a local minimizer and nonoptimal KKT point. For this reason, it is not particularly well-suited for nonconvex problems. Other, so-called second-order merit functions are able to guide the optimization through regions of nonconvexity and toward local minimizers rather than saddle-points or local maximizers.

1.5.4 Line-search methods

A line-search method uses two phases to obtain $v_{k+1}$ from $v_k$. The first phase finds a search direction $s_k$ that moves toward optimality for the constrained problem. The next phase computes a step-length $\alpha_k$ such that

$$
M(v_k + \alpha_k s_k) < M(v_k).
$$

The $(k + 1)$-st iterate is then defined as $v_{k+1} = v_k + \alpha_k s_k$.

For efficiency, it is important that $\alpha_k$ not be the exact minimizer along the search direction $s_k$. Therefore, most line-search methods are based on satisfying a sufficient decrease condition such as the Armijo condition:

$$
M(v_k + \alpha_k s_k) \leq M(v_k) + \eta_k \alpha_k \nabla M(v_k)^T s_k,
$$

(1.30)
where $\eta_s$ lies in the interval $(0, \frac{1}{2})$. Another condition often imposed on $\alpha_k$ is

$$|\nabla \mathcal{M}(v_k + \alpha_k s_k)^T s_k| \leq \eta_w |\nabla \mathcal{M}(v_k)^T s_k|,$$  \hspace{1cm} (1.31)

with Wolfe parameter $\eta_w$ ($0 < \eta_s < \eta_w < 1$). Conditions (1.30) and (1.31) comprise the Wolfe conditions. These are actually considered the strong Wolfe conditions, but will not be referred to as such. Some of the most sophisticated line-search methods use polynomial interpolation to determine $\alpha_k$ that satisfy the Wolfe conditions [25, 28, 50].

**Second-order line-search**

Line-search methods using the Wolfe conditions are designed to converge to first-order points, meaning $v_k \rightarrow v^*$ with $\nabla \mathcal{M}(v^*) = 0$. As the merit functions considered here are twice-differentiable it seems reasonable to incorporate second-order information so that the line-search may move through regions of indefiniteness that may contain saddle-points and local maximizers.

Second-order line-search methods are based on the concept of a descent pair $(p_k, d_k)$ such that one of the following conditions hold:

- $\nabla \mathcal{M}(v_k)^T p_k \leq 0$, $\nabla \mathcal{M}(v_k)^T d_k \leq 0$, and $d_k^T \nabla^2 \mathcal{M}(v_k)d_k < 0$,
- $\nabla \mathcal{M}(v_k)^T p_k < 0$, $\nabla \mathcal{M}(v_k)^T d_k \leq 0$, and $d_k^T \nabla^2 \mathcal{M}(v_k)d_k \leq 0$.

Given a descent pair $(p_k, d_k)$, $v_{k+1}$ is chosen as a point on the path

$$v(\alpha) = v_k + s_k(\alpha) \quad \text{where} \quad s_k(\alpha) = \omega_p(\alpha)p_k + \omega_d(\alpha)d_k,$$  \hspace{1cm} (1.32)

and $\omega_p(\alpha)$ and $\omega_d(\alpha)$ are scalar-valued functions such that $\omega_p(0) = \omega_d(0) = 0$. Two common choices for these weight functions are

$$\omega_p(\alpha) = \omega_d(\alpha) = \alpha \quad \text{and} \quad \omega_p(\alpha) = \alpha^2, \ \omega_d(\alpha) = \alpha.$$

The step-length $\alpha_k$ is chosen to satisfy the sufficient decrease condition

$$\mathcal{M}(v_k + \alpha_k s_k) \leq \mathcal{M}(v_k) + \eta_s \mathcal{Q}_k^-(\alpha s_k),$$

and

$$|\nabla \mathcal{M}(v_k + \alpha_k s_k)^T s_k| \leq \eta_w |\nabla \mathcal{M}(v_k)^T s_k|,$$

where $\mathcal{Q}_k^-$ is the quadratic function

$$\mathcal{Q}_k^-(s) = \nabla \mathcal{M}(v_k)^T s + \frac{1}{2} \min \left(0, s^T B(v_k)s\right).$$  \hspace{1cm} (1.33)

The matrix $B(v_k)$ serves as an approximation to the Hessian $\nabla^2 \mathcal{M}(v_k)$. 

1.5.5 Trust-region methods

Line-search methods can be characterized as “direction first” methods. A pure trust-region method, on the other hand, uses a “direction second” philosophy. The search direction at each step is calculated from a constrained quadratic subproblem that places an \textit{a priori} bound on the length of the step. The $k$-th quadratic subproblem is given by

$$
\minimize_{s \in \mathbb{R}^{n+m}} Q_k(s) \text{ subject to } \|s\| \leq \delta_k,
$$

where $\delta_k > 0$ is the trust-region radius, $\|\cdot\|$ is the trust-region norm, and $Q_k(s)$ is a quadratic model of the change in $M(v)$ at $v_k$ in the $s$-direction defined by,

$$
Q_k(s) = \nabla M(v_k)^T s + \frac{1}{2} s^T B(v_k) s.
$$

Again, $B(v_k)$ represents an approximation to the exact Hessian of $M(v)$. This thesis will deal with elliptic trust-region norms of the form $\|v\|_B^2 = v^T \tilde{B} v$ with $\tilde{B}$ positive definite. Another common choice for the trust-region norm is $\|\cdot\|_\infty$, in which case (1.34) is a bound constrained quadratic program (QP).

The trust-region grows or shrinks, depending on the ratio of actual versus predicted change in $M(v)$,

$$
\rho_k = \frac{M(v_k + s_k) - M(v_k)}{Q_k(s_k)}.
$$

When $\rho_k$ is large, $k$ is a “successful” iterate and $v_{k+1} = v_k + s_k$. On the other hand, if $\rho_k$ is small, then $v_{k+1}$ is set to $v_k$ and iteration $k$ is “unsuccessful”. Strategies for increasing or decreasing the trust-region radius are implementation-dependent, but are designed not to impede the strong local convergence properties of Newton’s method. As the trust-region radius changes, the trust-region subproblem must be solved with the updated radius.

The following trust-region algorithm uses the norm of the current search direction in updating the radius.
Algorithm 1.1 Basic trust-region algorithm

Choose constants $0 < \eta_1 < \eta_2 < 1$, $0 < \gamma_c < 1 < \gamma_e$;  
k = 0; $\delta_k = 1$; $v_k = (x_0, \pi_0)$;  
while not converged do  
Compute $s_k$ as an approximate solution of $\min \{Q_k(s) : \|s\| \leq \delta_k\}$;  
$r_k = (M(v_k + s_k) - M(v_k))/Q_k(s_k)$;  
if $r_k \geq \eta_1$ then  
$x_{k+1} = x_k + s_k$;  
if $r_k \geq \eta_2$ then $\delta_{k+1} = \max \{\delta_k, \gamma_c\|s_k\|\}$ else $\delta_{k+1} = \delta_k$;  
else [ unsuccessful step ]  
$x_{k+1} = x_k; \delta_{k+1} = \gamma_c\|s_k\|$;  
end if  
k = $k + 1$;  
end do

In this implementation, an iterate is said to be “successful” when $r_k \geq \eta_1$, “very successful” when $r_k \geq \eta_2$, and “unsuccessful” when $r_k < \eta_1$. Moreover, the trust-region is expanded at very successful iterates.

Two-norm trust-region method

This section emphasizes the structure of a single trust-region subproblem and therefore $B(v_k)$ is denoted by $B$, $g$ denotes the gradient $\nabla M(v_k)$, and the two-norm trust-region subproblem is

$$\minimize_{s \in \mathbb{R}^{n+m}} Q(s) \text{ subject to } \|s\|_2 \leq \delta,$$  \hspace{1cm} (1.37)

with $Q(s) = g^T s + \frac{1}{2}s^T Bs$. Unlike many problems in optimization, a global minimizer of the two-norm trust-region subproblem may be characterized completely.

**Theorem 1.5.2.** Let $\delta > 0$ be a given constant. A vector $s^*$ is a global minimizer of the two-norm trust-region subproblem with radius $\delta$ if and only if $\|s^*\|_2 \leq \delta$ and there exists a unique $\sigma^* \geq 0$ such that

$$(B + \sigma^*I)s^* = -g \quad \text{and} \quad \sigma^*(\delta - \|s^*\|_2) = 0,$$

with $B + \sigma^*I$ positive semidefinite. Moreover, if $B + \sigma^*I$ is positive-definite, then the global minimizer is unique.
Theorem 1.5.2 is proved by Gay [21] and Sorenson [60]. It forms the basis for the Levenberg-Marquardt algorithm used to solve nonlinear least-squares problems, from which trust-region methods are derived. Modern trust-region methods are based in large part on the work of Powell [58, 57] and Dennis and Mei [11].

Both direct and iterative methods may be used to solve the two-norm trust-region subproblem. Direct methods seek \((\sigma^*, s^*)\) by working with the optimality conditions of Theorem 1.5.2. Most iterative approaches work with the constrained optimization problem, rather than the optimality conditions.

One well-known direct method is due to Moré and Sorensen [49]. The Moré-Sorensen algorithm applies a safeguarded Newton method to find a nonnegative scalar \(\sigma\) that makes \(B + \sigma I\) positive definite and solves the secular equation \(\phi(\sigma) = 0\), where
\[
\phi(\sigma) = \frac{1}{\|s(\sigma)\|_2^2} - \frac{1}{\delta}, \quad \text{and} \quad s(\sigma) = -(B + \sigma I)^{-1}g.
\]
(1.38)
To ensure that \(B + \sigma I\) is positive definite, the algorithm makes extensive use of partial Cholesky factorizations, making it impractical for some large problems. Despite this drawback, it can be used to develop second-order trust-region methods.

Many iterative methods for solving problem (1.37) are based on the dog-leg method of Powell [58]. This method constructs a polygonal path \(s(\alpha)\) such that the univariate function \(Q(s(\alpha))\) is strictly decreasing. The computed solution either lies inside the trust-region, or is defined by the point at which \(s(\alpha)\) crosses the trust-region boundary. As a result these methods do not always yield an accurate estimate when \(s^*\) lies on the trust-region boundary.

Convergence of trust-region methods

Powell shows that if each trust-region step satisfies \(\|s_k\|_2 \leq \delta_k\) and
\[
Q(s_k) \leq -\frac{1}{2}\|g_k\|_2 \min \{\delta_k, \|g_k\|_2/\|B_k\|_2\},
\]
(1.39)
then Algorithm 1.1 converges to a stationary point of \(M\).

Theorem 1.5.3. Suppose \(M : D \subseteq \mathbb{R}^{n+m} \mapsto \mathbb{R}\) is continuously differentiable on an open convex set \(D\). Let \(\{v_k\} \subset D\) be a sequence of iterates generated by Algorithm 1.1. Assume that the trust-region step satisfies (1.39) and that \(\{\|B_k\|_2\}\) is bounded above. If \(M\) is bounded below in \(D\), then either \(\liminf_{k \to \infty} \|\nabla M(v_k)\|_2 = 0\) or some \(v_k\) satisfies the algorithm’s convergence criterion and the algorithm terminates.
If $\mathcal{M}$ is uniformly continuous on $\mathcal{D}$, then the results can be strengthened so that 
\[ \lim_{k \to \infty} \|\mathcal{M}(v_k)\|_2 = 0. \]
First-order trust-region methods do not explicitly use second-derivatives and therefore cannot be guaranteed to converge to a local minimizer. When the Moré-Sorensen algorithm is used in Algorithm 1.1, a sequence $\{v_k\}$ is produced with a limit point $v^*$ satisfying the following properties: $\nabla \mathcal{M}(v^*) = 0$ and $\nabla^2 \mathcal{M}(v^*)$ is positive semidefinite or $\nabla \mathcal{M}(v_j) = 0$ and $\nabla^2 \mathcal{M}(v_j)$ is positive semidefinite for some finite index $j$.

### 1.5.6 Combining trust-region and line-search methods

The bottleneck in trust-region methods is the solution of the trust-region subproblem. At unsuccessful iterates, $v_k$ is not changed and another trust-region subproblem must be solved with a smaller trust-region radius. Although a trust-region step may result in an unsuccessful iteration, it may define a suitable line-search step. Moreover, the line-search step-length may be used to update the trust-region radius. Algorithms that combine line-search and trust-region methods have been considered by Toint [62], Nocedal and Yuan [52], Gertz [22, 23], and Gertz and Gill [24]. Adding a line-search to a trust-region algorithm also makes sense in the inequality constrained setting when $\mathcal{M}(v)$ includes a barrier term. In this case, progress can be made when the trust-region step is infeasible.

If the values of the $\mathcal{M}(v)$ and $Q^-(s)$ on the rays $v(\alpha) = v_k + \alpha s_k$ and $s(\alpha) = \alpha s_k$ are denoted by 
\[ M_k(\alpha) \triangleq \mathcal{M}(v_k + \alpha s_k) \quad \text{and} \quad q_k(\alpha) \triangleq Q^-(\alpha s_k), \]
then the following algorithm adds a Wolfe line-search to a trust-region method (see Gertz [22]).
Algorithm 1.2 Wolfe trust-region algorithm

Specify constants $0 < \eta_s < \eta_w < 1 < \nu$;

$k = 0; \quad \delta_k = 1; \quad v_0 = (x_0, \pi_0)$;

while not converged do

Compute $s_k$ as an approximate solution of $\min \{ Q_k(s) : \| s \|_2 \leq \delta_k \}$;

Find $\alpha_k$ such that,

$M_k(\alpha_k) \leq M_k(0) + \eta_s q_k(\alpha_k) \quad \text{and} \quad |M_k'(\alpha_k)| \leq \eta_w |q_k'(\alpha_k)|$

$x_{k+1} = x_k + \alpha_k s_k$;

Choose $\delta_{k+1} \in [\alpha_k \| s_k \|_2, \alpha_k \nu \| s_k \|_2]$;

$k = k + 1$;

end do

1.6 Other research

Interior-point methods for constrained optimization problems were proposed by Frisch [20] and Fiacco and McCormick in the 1960s [15]. A barrier method for problem (1.1) solves a sequence of penalty-barrier functions

$P_\mu(x) = f(x) + \frac{1}{2\mu} \| c(x) \|^2$.

Analogous to primal-dual interior-point methods, the idea is to compute a sequence $\{ x_k^* \}$ of approximate minimizers over a sequence of decreasing barrier parameters $\{ \mu_k \} \to 0$. The two-norm term has the effect of penalizing infeasibility of the equality-constraints. Murray [51] and Loostma [46] show independently that the Hessian of these “quadratic-penalty” functions become increasingly ill-conditioned as $\mu \to 0$. This, together with the adoption of more sophisticated methods such as augmented Lagrangian and SQP methods, led to a lack of interest in interior-point methods. Interest in barrier methods was revived in 1984 when Karmarkar announced the creation of a polynomial-time algorithm for linear programming problems [44]. Although Karmarkar’s method was shown to be formally equivalent to classical barrier methods [27], Karmarkar’s work helped initiate the so-called “interior-point revolution” and led to a number of new algorithms.
1.6.1 Augmented Lagrangian methods

Augmented Lagrangian methods for problem (1.25) involve approximately minimize a sequence of primal augmented Lagrangian functions

\[ L(\mu; x, \pi, \rho) = f(\mu; x) - c(x)^T \pi + \frac{\rho}{2} \| c(x) \|^2_2, \]

where \( \rho \) is a positive penalty-parameter and \( \pi \) is a Lagrange multiplier estimate. This function was proposed independently by Hestenes [42] and Powell [56] to solve equality-constrained problems. Augmented Lagrangian methods operate under the assumption that a sufficiently large penalty parameter creates a local minimizer for the augmented Lagrangian function. A sequence of pairs \( \{ (x_k, \pi_k) \} \) is computed with k-th member defined by,

\[ x_k \approx \arg\min_{x \in \mathbb{R}^n} L(\mu; x, \pi_k, \rho_k), \]  

where \( \{ \rho_k \} \) is a sequence of penalty parameters chosen to ensure that the Hessian of \( L(\mu) \) is locally positive definite. If \( \rho \) is fixed, it can be shown that \( x_k \to x^* \) only if \( \pi_k \to \pi^* \).

The Lagrange multiplier estimates serve as parameters rather than optimization variables, and are updated as the iterations proceed. First-order Lagrange multiplier estimates are defined by the formula

\[ \pi(x; \pi^e, \rho) = \pi^e - \rho c(x), \]

and are based on the observation that the equations

\[ 0 = \nabla_x L(\mu; x, \pi_k, \rho_k) = g(\mu; x) - J^T \pi_k, \quad \text{with} \quad \pi_k \triangleq \pi(x_k; \pi^e_k, \rho_k) \]  

define the stationarity conditions for problem (1.25). First-order multiplier updates define linearly convergent augmented Lagrangian methods at best. Quadratically convergent augmented Lagrangian algorithms require second-order multiplier estimates. More information on augmented Lagrangian methods may be found in Bertsekas [6].

1.7 Contributions of this thesis

Methods are proposed for solving a sequence of equality-constrained primal barrier functions arising in PDE constrained optimization. Each of these subproblems is solved by minimizing a sequence of primal-dual augmented Lagrangian functions defined as

\[ M(\mu; \pi; \pi^e, \rho) = f(\mu; x) - c(x)^T \pi + \frac{\rho}{2} \| c(x) \|^2_2 + \frac{\rho}{2} \| c(x) \|^2 + \frac{1}{\rho} \| \pi - \pi^e \|^2_2, \]
proposed by Robinson [59] and Gill and Robinson [34]. A scaled version of this function is necessary in the PDE context and is derived in Chapter 2. In contrast to augmented Lagrangian methods, optimization methods based on the primal-dual augmented Lagrangian minimize with respect to both the primal and dual variables.

Minimizing a single primal-dual augmented Lagrangian function is unconstrained in the primal and dual variables. A truncated Newton trust-region method is used to carry out the approximate unconstrained minimization. The trust-region prevents unproductive, and potentially unbounded search directions. It is shown that a direction of negative curvature is detected when the inertia of a related, regularized KKT matrix has incorrect inertia. A second-order line-search method is used in conjunction with a trust-region algorithm to deal with these problematic search directions.

In PDE-constrained optimization, computing the inertia of a KKT matrix is prohibitively expensive. A benefit of the trust-region method is that incorrect inertia is detected “on-the-fly” using a variant of the preconditioned conjugate-gradient method (PCG) developed independently by Steihaug [61] and Toint [62]. The conjugate-gradient algorithm has a natural, built-in mechanism for detecting indefiniteness and is terminated in this case, yielding a descent pair.

Convergence of CG may be accelerated with a positive definite preconditioner. Traditional preconditioning strategies in PDE-constrained optimization rely on preconditioners for the underlying PDE, which makes them inappropriate for regularized KKT systems. Therefore, a major component of this research is the development of a preconditioning strategy for PDE-constrained optimization that takes advantage of both regularization and sparsity. The method is based on performing an incomplete factorization of a symmetric quasi-definite approximation of the regularized KKT matrix. Incomplete factorizations have played a prominent role in preconditioning positive definite finite-element matrices. It is shown how the incomplete factorization implicitly defines a positive-definite preconditioner required for the trust-region subproblem solver.

1.8 Outline

The rest of this chapter presents results in linear algebra that will be important throughout the thesis. The primal-dual augmented Lagrangian function and its role in interior-point methods is described in Chapter 2. The emphasis in Chapter 3 is methods for computing search directions for a fixed barrier subproblem. Conventional preconditioning
approaches for KKT matrices arising in PDE-constrained optimization, and methods for dealing with regularized KKT matrices are presented in Chapter 4. An incomplete, sparsity exploiting preconditioning strategy is proposed in Chapter 5. Finally, preliminary numerical results are presented in Chapter 6.

1.9 Useful Results

Some important results in linear algebra are reviewed below.

**Theorem 1.9.1** (Sylvester’s law of inertia). Given a symmetric $K$ and a nonsingular matrix $U$ of the same dimension, $\text{In}(K) = \text{In}(UKU^T)$.

**Theorem 1.9.2** (Inertia of a KKT system). Given an $n \times n$ symmetric matrix $H$ and an $m \times n$ matrix $J$, let $r = \text{rank}(J)$ and let $Z$ be a matrix whose columns form a basis for $\text{null}(A)$. If $K$ is defined as

$$K = \begin{pmatrix} H & J^T \\ J & 0 \end{pmatrix}, \quad \text{then} \quad \text{In}(K) = \text{In}(Z^T H Z) + (r, r, m - r).$$

**Theorem 1.9.3** (Inertia of regularized KKT matrix). Given an $n \times n$ symmetric matrix $H$, an $m \times n$ matrix $J$, and an $m \times m$ nonsingular matrix $D$, if $K$ is defined as

$$K = \begin{pmatrix} H & J^T \\ J & -D \end{pmatrix}, \quad \text{then} \quad \text{In}(K) = \text{In}(H + J^T D^{-1} J) + \text{In}(-D).$$

**Theorem 1.9.4** (Debreau’s Lemma). Given an $n \times n$ symmetric $H$ and an $m \times n$ matrix $J$, then $p^T H p > 0$ for all nonzero $p \in \text{null}(J)$ if and only if there is a finite $\bar{\rho} > 0$ such that $H + \rho J^T J$ is positive definite for all $\rho > \bar{\rho}$.

**Theorem 1.9.5** (Symmetric indefinite factorization). Let $K$ be a nonsingular symmetric matrix. Then there exist a permutation matrix $\Pi$, a unit lower triangular matrix $L$, and block diagonal matrix $B$ with $1 \times 1$ and $2 \times 2$ blocks such that

$$\Pi^T K \Pi = L B L^T,$$

with every $2 \times 2$ block of $B$ having one positive and one negative eigenvalue.
Chapter 2

Interior methods

2.1 Overview

In this chapter, we develop an interior-point method for solving the problem

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad c_E(x) = 0, \ c_I(x) \geq 0,
\end{align*}
\]

by finding an approximate local minimizer of the sequence of equality-constrained problems

\[
\begin{align*}
\text{minimize} & \quad f_\mu(x) = f(x) - \mu_k \sum_{i \in I} w_i \ln(c_i(x)) \\
\text{subject to} & \quad c_E(x) = 0
\end{align*}
\]

as \( \mu_k \to 0 \), where the vector \( w \in \mathbb{R}^{m_I} \) has strictly positive entries. As the emphasis is on solving a sequence of equality-constrained subproblems, \( c_E(x) \) and \( J_E(x) \) will be denoted by \( c(x) \) and \( J(x) \), respectively. A KKT point for problem (2.2) with \( \mu = \mu_k \) is denoted by \( v(\mu_k) = (x(\mu_k), \pi(\mu_k)) \). An approximate subproblem solution \( v^*_k = (x^*_k, \pi^*_k) \) is obtained by minimizing a sequence of primal-dual merit functions that are also described in this chapter.

2.2 Background on barrier-SQP methods

A KKT point for

\[
\begin{align*}
\text{minimize} & \quad f_\mu(x) \quad \text{subject to} \quad c(x) = 0.
\end{align*}
\]

satisfies the equation

\[
F_\mu(x, \pi) = \begin{pmatrix}
g_\mu(x) - J(x)^T \pi \\
c(x)
\end{pmatrix} = 0.
\]
For a fixed $\mu$, the barrier-SQP approach for problem (2.1) seeks a KKT point by solving a sequence of symmetrized Newton equations

$$
\begin{pmatrix}
H(\mu, x, \pi) & J(x)^T \\
J(x) & -D
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
-\Delta \pi
\end{pmatrix}
= -\begin{pmatrix}
g(\mu) - J(x)^T \pi \\
c(x)
\end{pmatrix}.
$$

(2.5)

The matrix in equation (2.5) is denoted by $K_{\mu}(x, \pi)$ and in the current context, $D = 0$. The following theorem relates the inertia of $K_{\mu}(x, \pi)$ to the local convexity of problem (2.1).

**Lemma 2.2.1.** Suppose $c_T(x) > 0$ and rank($J(x)$) = $m$. If $K_{\mu}(x, \pi)$ is nonsingular and has more than $m$ negative eigenvalues, then there exists a direction $p \in \text{null}(J(x))$ such that $p^T H(\mu, x, \pi)p < 0$.

*Proof.* Theorem 1.9.2 implies that if In($K_{\mu}(x, \pi)$) $\neq (n, m, 0)$ and rank($J(x)$) = $m$, then there exists a direction $p \in \text{null}(J(x))$ such that $p^T H(\mu, x, \pi)p < 0$. The result now follows from the fact that $H(\mu, x, \pi)$ is a positive-definite perturbation of $H(x, \pi)$. 

As $K_{\mu}(x, \pi)$ is a positive-definite perturbation of $K(x, \pi)$, it is not generally true that

$$\text{In}(K_{\mu}(x, \pi)) = (n, m, 0) \quad \text{implies} \quad \text{In}(K(x, \pi)) = (n, m, 0).$$

However, as $\mu_k \to 0$, if

$$F_{\mu_k}(x^*_k, \pi^*_k) \approx 0 \quad \text{and} \quad \text{In}(K_{\mu_k}(x^*_k, \pi^*_k)) = (n, m, 0),$$

then under certain conditions, \(\{x^*_k, \pi^*_k, \mu_k S_k^{-1}\} \to (x^*, \pi^*, z^*)\), where $S_k = \text{diag}(c_T(x^*_k))$ and $(x^*, \pi^*, z^*)$ satisfies the second-order necessary conditions.

A suitable merit function for problem (2.3) must deal with points $(x, \pi)$ where $K_{\mu}(x, \pi)$ has more than $m$ negative eigenvalues. Assuming $K_{\mu}(x, \pi)$ is nonsingular, the fact that $K_{\mu}(x, \pi)$ has incorrect inertia implies that a direction of negative curvature for the reduced Hessian $H_Z(x, \pi)$ must exist. It would be advantageous to work with a merit function that decreases along such directions. Below is a standard implementation of a primal interior-point method.
Algorithm 2.1 Primal interior-point method

Choose $\mu_0 \gg \mu^* > 0$;

Set $k = 0$;

Choose $v_k^* = (x_k, \pi_k)$;

**do until** $\mu_k \leq \mu^*$

\[
j = 0;
\]

\[
v_j = v_k^*;
\]

**do until** $\|F_{\mu_k}(v_j)\| = O(\mu_k)$

Solve $F'_{\mu_k}(v_j)s_j = -F_{\mu_k}(v_j) + r_j$ with $\|r_j\| \leq \tau_j\|F_{\mu_k}(v_j)\|$;

Find $\alpha_j$ such that $M(v_j + \alpha_js_j) < M(v_j)$;

\[
j = j + 1;
\]

**end do**

$v_k^* = v_j$;

Choose $\mu_{k+1} < \mu_k$;

Set $k = k + 1$;

**end do**

The search directions in the inner loop of Algorithm 2.1 will be computed with an inexact Newton method. Inexact Newton methods are the subject of Chapter 3.

2.3 A primal-dual augmented Lagrangian

A fixed barrier subproblem has equality constraints only, and it seems reasonable to use a merit function associated with the equality-constrained optimization to problem (2.2). Of particular interest here is the *primal-dual augmented Lagrangian* function given by

\[
M(x, \pi; \pi^e, \rho) = f_\mu(x) - c(x)^T\pi^e + \frac{\rho}{2}\|c(x)\|_2^2 + \frac{\rho}{2}\|c(x) + \frac{1}{\rho}(\pi - \pi^e)\|_2^2.
\] (2.6)

This is just the Hestenes-Powell augmented Lagrangian function

\[
\mathcal{L}_\mu(x; \pi^e, \rho) = f_\mu(x) - c(x)^T\pi^e + \frac{\rho}{2}\|c(x)\|_2^2,
\] (2.7)

for a fixed barrier subproblem, with an additional proximity term. As in the case of the augmented Lagrangian function, the modified function may be used to develop methods for constrained optimization based on solving a sequence of unconstrained problems.
2.3.1 Scaling in the PDE context

In the finite-element discretization of PDE-constrained optimization problems, both the vector of equality constraints and the objective function are implicitly scaled by the area of an element. As a result, the use of the standard two-norm of $c(x)$ will lead to a poorly scaled merit function. To remedy this, a scaled primal-dual augmented Lagrangian function is proposed that uses a scaled norm in place of the two-norm. This norm is specified by a positive-definite diagonal scaling matrix $W$, and is given by

$$
\|y\|_h^2 = y^T W^{-1} y, \quad \text{for } y \in \mathbb{R}^m.
$$

(2.8)

Defining $D_\rho = \frac{1}{\rho} W$ gives the scaled primal-dual augmented Lagrangian

$$
M_\mu(x, \pi; \pi^e, \rho) = f_\mu(x) - c(x)^T \pi^e + \frac{\rho}{2} \|c(x)\|_h^2 + \frac{\rho}{2} \|c(x) + D_\rho (\pi - \pi^e)\|_h^2.
$$

(2.9)

The gradient and Hessian of $M_\mu$ are

$$
\nabla M_\mu(x, \pi; \pi^e, \rho) = \begin{pmatrix} g_\mu(x) - J(x)^T (2\tilde{\pi} - \pi) \\ D_\rho (\pi - \pi^e) \end{pmatrix},
$$

(2.10)

and

$$
\nabla^2 M_\mu(x, \pi; \pi^e, \rho) = \begin{pmatrix} H_\mu(x, 2\tilde{\pi} - \pi) + 2J(x)^T D_\rho^{-1} J(x) & J(x)^T \\ J(x) & D_\rho \end{pmatrix},
$$

(2.11)

where $\tilde{\pi} \triangleq \tilde{\pi}(x; \pi^e, \rho) = \pi^e - D_\rho^{-1} c(x)$ is the vector of first-order multiplier estimates. The following theorem justifies the role of the primal-dual augmented Lagrangian as a second-order merit function. For a proof in the unscaled case, see Gill and Robinson [34].

**Theorem 2.3.1.** Let $(x^*, \pi^*)$ satisfy the following optimality conditions associated with minimizing $f_\mu(x)$ subject to the equality constraints $c(x) = 0$:

1. $J(x^*)^T \pi^* = g_\mu(x^*)$,
2. $c(x^*) = 0$, $c_I(x^*) > 0$, and
3. $\text{In}(K_\mu(x^*, \pi^*)) = (n, m, 0)$.

Then there exists a finite $\bar{\rho}$ such that $(x^*, \pi^*)$ is an unconstrained minimizer of $M_\mu(x, \pi; \pi^*, \rho)$ for all $\rho > \bar{\rho}$. 

Proof. First, we show that \((x^*, \pi^*)\) is a stationary point of \(M(x, \pi; \pi^*, \rho)\), for any \(\rho > 0\). To see this, note that \(\bar{\pi} = \pi^e - D^{-1}_\rho c(x^*) = \pi^e\), and therefore
\[
\nabla M_\mu(x^*, \pi^*; \pi^*, \rho) = \begin{pmatrix} g_\mu(x^*) - J(x^*)^T \pi^* \\ D_\rho(\pi^* - \pi^*) \end{pmatrix} = 0. 
\] (2.12)

To complete the proof, it suffices to show that \(\nabla^2 M_\mu(x^*, \pi^*; \pi^*, \rho)\) is positive-definite for sufficiently large \(\rho\). Let \(H_\mu = H_\mu(x^*, \pi^*)\), and \(J = J(x^*)\). The Hessian is given by
\[
\nabla^2 M_\mu = \begin{pmatrix} H_\mu + 2J^T D^{-1}_\rho J & J \\ J & D_\rho \end{pmatrix},
\] (2.13)
with \(\text{In} (\nabla^2 M_\mu) = \text{In}(H_\mu + J^T D^{-1}_\rho J) + (m, 0, 0)\). Let \(Q\) be a nonsingular \(n \times n\) matrix of the form \(Q = (Z \ Y)\), where the columns of \(Z\) form a basis for \(\text{null}(J)\). Sylvester’s law of inertia implies that the inertia of \(H_\mu + J^T D^{-1}_\rho J\) is the same as the inertia of
\[
Q^T (H_\mu + J^T D^{-1}_\rho J) Q = Q^T H_\mu Q + Q^T J^T D^{-1}_\rho J Q
\]
\[
= \begin{pmatrix} Z^T H_\mu Z & Z^T H_\mu Y \\ Y^T H_\mu Z & Y^T H_\mu Y + \rho Y^T J^T G J Y \end{pmatrix}. 
\] (2.14)

Let \(H_{11} = Z^T H_\mu Z\), \(H_{12} = Y^T H_\mu Z\), and \(H_{22} = Y^T H_\mu Y\). The Schur complement from the matrix in (2.14) is
\[
\text{In}(H_\mu + J^T D^{-1}_\rho J) = \text{In}(H_{11}) + \text{In}(H_{22} - H_{21} H_{11}^{-1} H_{21}^T + \rho Y^T J^T G J Y).
\]

Let \(\bar{\lambda}_{\text{min}}\) and \(\hat{\lambda}_{\text{min}}\) denote the smallest eigenvalues of \(H_{22} - H_{21} H_{11}^{-1} H_{21}^T\) and \(Y^T J^T G^{-1} J Y\), respectively, and let \(\bar{\rho} = \min \left\{ 0, -\bar{\lambda}_{\text{min}}/\hat{\lambda}_{\text{min}} \right\}\). Then for any \(\rho > \bar{\rho}\), and for any nonzero vector \(x\), it follows that
\[
x^T (H_{22} - H_{21} H_{11}^{-1} H_{21}^T) x + \rho x^T (Y^T J^T G^{-1} J Y) x
\]
is bounded below by the positive quantity
\[
\bar{\lambda}_{\text{min}} \|x\|_2^2 + \rho \hat{\lambda}_{\text{min}} \|x\|_2^2,
\]
which implies the result. \(\square\)

Therefore, if \(\pi^*\) is known, \(x^*\) can be obtained by performing one unconstrained minimization. In the general case when \(\pi_e \neq \pi^*\), this result also implies that if \(\nabla M_\mu(x, \pi; \pi^e, \rho)\) is zero, then \(\pi\) is likely to be a better estimate of \(\pi^*\) than \(\pi^e\).
2.4 Doubly-augmented systems

The Newton equations associated with minimizing $\mathcal{M}_\mu(x, \pi; \pi^e, \rho)$ constitute a doubly-augmented system, the solution of which is discussed by Forsgren, Gill, and Griffin [18] in the context of preconditioners for KKT matrices arising in primal-dual interior-point methods. Preconditioning will be discussed in Chapters 4 and 5. Consider the parameterized equations $K_\nu s = -b_\nu$ where

$$K_\nu = \begin{pmatrix} H + (1 + \nu)J^TD^{-1}J & \nu J^T \\ \nu J & \nu D \end{pmatrix}, \quad b_\nu = \begin{pmatrix} b_x + (1 + \nu)J^TD^{-1}b_x \\ \nu b_x \end{pmatrix},$$

(2.15)

$H$ is an $n \times n$ symmetric matrix, $D$ is an $m \times m$ positive-definite diagonal matrix, $b_x \in \mathbb{R}^n$, and $b_x \in \mathbb{R}^m$. There is a close relationship between the inertia of these parameterized matrices, as the following theorem illustrates.

**Theorem 2.4.1** (Inertia of parameterized systems). The inertia of $K_\nu$ for $\nu = 1, 0, -1$ satisfies

$$\text{In}(K_\nu) = \begin{cases} \text{In}(H_D) + (m, 0, 0) & \text{if } \nu = 1 \\ \text{In}(H_D) + (0, m, 0) & \text{if } \nu = -1 \\ \text{In}(H_D) + (0, 0, m) & \text{if } \nu = 0 \end{cases},$$

where $H_D = H + J^TD^{-1}J$.

**Proof.** Consider the nonsingular matrix $U_\nu$ defined by,

$$U = \begin{pmatrix} I & -J^TD^{-1} \\ 0 & I \end{pmatrix}$$

and note that

$$UK_\nu U^T = \begin{pmatrix} H + J^TD^{-1}J & 0 \\ 0 & \nu D \end{pmatrix}.$$ 

Sylvester’s law of inertia immediately implies the result.

The matrix $H_D$ of Theorem 2.4.1 is referred to as a condensed system. With this notation, observe that $K_\nu$ can be expressed as

$$K_\nu = \begin{pmatrix} H_D + \nu J^TD^{-1}J & \nu J^T \\ \nu J & \nu D \end{pmatrix}.$$ 

(2.16)
This notation will be used later in the formulation of preconditioning strategies. By applying the technique used in Theorem 2.4.1, it follows that the primal-dual augmented is positive-definite if and only if the matrix

\[
\begin{pmatrix}
H_\mu(x, 2\tilde{\pi} - \pi) & J(x)^T \\
J(x) & -D_\rho
\end{pmatrix}
\]

has inertia \((n, m, 0)\).

## 2.5 Primal-dual augmented Lagrangian methods

Primal-dual augmented Lagrangian methods are proposed by Gill and Robinson in the context of equality-constrained and bound-constrained problems [59, 30, 31, 32, 33, 34, 35]. In these algorithms, the constrained problem is solved by minimizing a sequence of primal-dual augmented Lagrangian functions in the spirit of augmented and bound-constrained Lagrangian methods. This section introduces methods based on the primal-dual augmented Lagrangian and describes how they relate to a fixed barrier subproblem.

### 2.5.1 Regularized primal-dual SQP method

Gill and Robinson propose a regularized SQP method for problems of the form

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad c(x) = 0, \quad x \geq 0,
\end{align*}
\]

that uses a “filter-like” technique to monitor the progress being made in satisfying the KKT conditions [35]. A first-order KKT point for problem (2.17), \((x^*, \pi^*, z^*)\) satisfies the conditions

\[
c(x^*) = 0, \quad x^* \geq 0
\]

\[
z^* = g(x^*) - J(x^*)^T \pi^*
\]

\[
x \cdot z = 0.
\]

As in the case of interior methods, the regularized SQP method use major and minor iterations. An approximate KKT point of problem (2.17) is found by solving a sequence of linearly constrained, quadratic subproblems. The objective function for the \(k\)-th quadratic subproblem is a quadratic approximation of the primal-dual augmented Lagrangian function defined at \((x_k, \pi_k)\).
The following two functions measure the residual for the KKT conditions, in a compact form:

\[ \sigma_S(x, \pi) = \eta(x) + \beta \omega(x, \pi) \quad \text{and} \quad \sigma_L(x, \pi) = \beta \eta(x) + \omega(x, \pi), \]  

(2.18)

where \( \eta(x) = \|c(x)\|_\infty \) and \( \omega(x, \pi) = \|\min(x, g(x) - J(x)^T \pi)\|_\infty \). The parameter \( \beta (\beta > 0) \) is fixed and is intended to blend feasibility, stationarity, and complementarity. Let \( \sigma_{S}^{\text{max}} \) and \( \sigma_{L}^{\text{max}} \) denote thresholds that are updated during the solution process. If

\[ \sigma_S(x_j, \pi_j) \leq \gamma \sigma_{S}^{\text{max}} \quad \text{or} \quad \sigma_L(x_j, \pi_j) \leq \gamma \sigma_{L}^{\text{max}} \]  

(2.19)

for \( \gamma \) between 0 and 1, then \((x_j, \pi_j)\) is considered to be “good enough” to warrant an update of the current multiplier estimate. To achieve global convergence, this update is accompanied by a decrease in \( \sigma_{S}^{\text{max}} \) or \( \sigma_{L}^{\text{max}} \), depending on which is satisfied. The multiplier estimate is also updated if \((x_j, \pi_j)\) satisfies

\[ \|\nabla M(x_j, \pi_j; \pi^e_j, \rho_j)\| \leq \tau_j, \quad \text{where} \quad \tau_j > 0. \]  

(2.20)

If (2.20) is satisfied but (2.19) is not, then \( x_j \) is not sufficiently feasible and \( \pi_j \) may not be the best choice for \( \pi^e_{j+1} \). In this case, the first-order multiplier estimates, \( 2\tilde{\pi}_j - \pi_j \) are used for \( \pi^e_{j+1} \). If neither (2.19) nor (2.20) hold, the method resorts to minimizing a primal-dual augmented Lagrangian function with fixed multiplier estimates and penalty parameter. The unconstrained minimization can be globalized with a line-search or trust-region method.

### 2.5.2 Using an approximate Hessian

Rather than minimize \( M_\mu(x, \pi; \pi^e, \rho) \) directly using a line-search or trust-region method, a search direction will be defined from a regularized KKT matrix. Consider the matrix

\[ B_\mu(x, \pi; \rho) = \begin{pmatrix} H_\mu(x, \pi) + 2J(x)^TD_\rho^{-1}J(x) & J(x)^T \\ J(x) & D_\rho \end{pmatrix}, \]  

(2.21)

obtained by replacing \( \tilde{\pi} \) by \( \pi \) in \( \nabla^2 M_\mu(x, \pi; \pi^e, \rho) \) of (2.11). Observe that for any \( x \),

\[ B_\mu(x, \pi; \rho) = \nabla^2 M_\mu(x, \pi; \pi^e, \rho), \]  

(2.22)

which implies that (2.21) may be viewed as an approximate Hessian for \( M_\mu(x, \pi; \pi^e, \rho) \).

The approximate Newton equations \( B_\mu(v)\Delta v = -\nabla M_\mu(v; \pi^e, \rho) \) in block form are

\[ \begin{pmatrix} H_\mu + 2J^TD_\rho^{-1}J & J^T \\ J & D_\rho \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \pi \end{pmatrix} = -\begin{pmatrix} g_\mu - J^T(2\tilde{\pi} - \pi) \\ c + D_\rho(\pi - \pi^e) \end{pmatrix}, \]
where \( H_\mu = H_\mu(x, \pi), J = J(x), g_\mu = g_\mu(x), \) and \( c = c(x) \). Multiplying on the left by
\[
\begin{pmatrix}
I & -2J^T D_\rho^{-1} \\
0 & I
\end{pmatrix},
\]
and symmetrizing yields the equivalent system
\[
\begin{pmatrix}
H_\mu & J^T \\
J & -D_\rho
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
-\Delta \pi
\end{pmatrix} = -
\begin{pmatrix}
g_\mu - J^T \pi \\
c + D_\rho (\pi - \pi^e)
\end{pmatrix}.
\tag{2.23}
\]
This defines a nonsingular system if \( H_\mu + J^T D_\rho^{-1} J \) is nonsingular.

If the matrix from (2.23) is denoted \( K_\mu(x, \pi; \rho) \) then it follows from Sylvester’s Law of Inertia that if \( \text{In}(K_\mu(x, \pi; \rho)) \neq (n, m, 0) \), then \( B_\mu(x, \pi; \rho) \) (and hence \( H_\mu + J^T D_\rho^{-1} J \)) is indefinite. If the reduced-Hessian is indefinite, then no finite \( \rho \) will make the condensed matrix positive-definite. The matrix \( H_\mu + J^T D_\rho^{-1} J \) may also be indefinite if the reduced-Hessian is positive-definite but \( \rho \) is not large enough to make \( B_\mu(x, \pi; \rho) \) positive-definite in the range space of \( J \).

### 2.5.3 The inner-iterations

The \( k \)-th outer-iterate computes an \((x^*_k, \pi^*_k)\) such that \( F_{\mu_k}(x^*_k, \pi^*_k) = O(\mu_k) \) by minimizing a sequence of primal-dual augmented Lagrangians defined by a nondecreasing sequence of penalty parameters \( \{\rho_j\} \) and a sequence of multiplier estimates \( \{\pi^e_j\} \). These sequences may remain constant over a range of inner-iterates, in which case a single primal-dual augmented Lagrangian function is being minimized. As in augmented Lagrangian methods, the sequence of penalty parameters is intended to make the primal-dual augmented Lagrangian locally convex. Increasing the penalty parameter places more emphasis on satisfying the equality constraints. The sequence of multiplier estimates affects the convergence rate of the method. If \( \{\pi^e_j\} \) is held fixed throughout the inner-iterations, then the method resembles a quadratic-penalty method, which does converge under certain circumstances if \( \rho_j \to \infty \). Moreover, if \( \rho = 1/\mu \), then the classical penalty-barrier method is recovered. A topic worth investigating in another paper is the circumstances under which \( \rho_j \) and \( \pi^e_j \) should be updated.

In the barrier-SQP context, the functions associated with the filter, measure constraint infeasibility and stationarity
\[
\sigma^h_S(x, \pi) = \eta(x) + \beta \omega(x, \pi) \quad \text{and} \quad \sigma^m_S(x, \pi) = \beta \eta(x) + \omega(x, \pi), \tag{2.24}
\]
where $\eta(x) = \|c(x)\|_{\infty}$ and $\omega_\mu(x, \pi) = \|g_\mu(x) - J(x)^T\pi\|_{\infty}$. Because the emphasis is on the inner-iterations, the subscript $k$ will be ignored. The $j$-th search direction is derived from the doubly-augmented system

$$B_\mu(v_j; \rho_j) \Delta v_j = -\nabla M_\mu(v_j; \pi_j^e, \rho_j),$$

or equivalently, the regularized KKT system,

$$
\begin{pmatrix}
    H_\mu(x_j, \pi_j) & J(x_j)^T \\
    J(x_j) & -D_j
\end{pmatrix}
\begin{pmatrix}
    \Delta x_j \\
    -\Delta \pi_j
\end{pmatrix}
= -
\begin{pmatrix}
    g_\mu(x_j) - J(x_j)^T(2\pi_j - \pi_j) \\
    c(x_j) - D_j(\pi_j - \pi_j^e)
\end{pmatrix},
$$

where $D_j = D_\rho_j$.

**The treatment of indefiniteness**

Unlike SQP methods, the regularized KKT matrix $K_\mu(x, \pi; \rho)$ uses exact second-derivatives, and therefore the use of directions of negative curvature is critical for convergence to second-order points. Recall that a nonsingular regularized KKT matrix $K_\mu(x, \pi; \rho)$ has more than $m$ negative eigenvalues if and only if $B_\mu(x, \pi; \pi_j^e, \rho)$ is not positive-definite. This may occur because the KKT matrix $K_\mu(x, \pi)$ has more than $m$ negative eigenvalues, or because $\rho$ is too small. In the large-scale setting, computing the inertia of $K_\mu(x, \pi)$ is prohibitively expensive. Therefore, if a direction $d_j$ can be found such that $d_j^T B_\mu(x_j, \pi_j; \pi_j^e, \rho_j) d_j < 0$, a second-order line-search based on the quadratic function

$$Q_{\overline{j}}(s) = \nabla M_\mu(v; \pi_j^e, \rho_j)^T s + \frac{1}{2} \min \left\{ 0, s^T B_\mu(v; \pi_j^e, \rho_j) s \right\},$$

(2.25)

can be used to move away from $(x_j, \pi_j)$. In order to allow for the possibility that $\rho_j$ is not large enough, the detection of indefiniteness should be accompanied by an increase in the penalty parameter. The decision on whether or not $\pi_j^e$ should be updated can be based on the filter merit functions described previously.

**Formal statement of the algorithm**

The following algorithm uses a trust-region strategy to minimize a sequence of primal-dual augmented Lagrangians.
Algorithm 2.2 Primal-dual augmented Lagrangian method

Input $v_0 = (x_0, \pi_0)$;
Set $\pi_0^e = \pi_0$; indef = false;
Initialize $\rho_0 > 0$; $\tau_{\text{stop}} > 0$; $\gamma \in (0, 1)$;

for $k = 0, 1, 2, \ldots$ do

Compute $s_k \approx \text{argmin} \{Q_k(s) : \|s\|_k \leq \delta\}$;
if negative curvature detected then indef = true;
Find $\alpha_k$ with a line-search;
$v_{k+1} = v_k + \alpha_k s_k$;
Update trust-region radius to get $\delta_{k+1}$;

if $\sigma^S(v_{k+1}) \leq \gamma \sigma_{\text{max}}^S$ then
$\sigma_{\text{max}}^S = \gamma \sigma_{\text{max}}^S$; $\rho_{k+1} = \rho_k$; $\pi_{k+1}^e = \pi_k^e$; $\tau_{k+1} = \tau_k$;

else if $\sigma^L(v_{k+1}) \leq \gamma \sigma_{\text{max}}^L$ then
$\sigma_{\text{max}}^L = \gamma \sigma_{\text{max}}^L$; $\rho_{k+1} = \rho_k$; $\pi_{k+1}^e = \pi_k^e$; $\tau_{k+1} = \tau_k$;

else if $\|M_{\mu}(v_{k+1}; \pi_k^e, \rho_k)\| \leq \tau_k$ or indef then
Choose $\pi_{k+1}^e$, $\pi_{k+1}$, $\tau_{k+1} < \tau_k$, and $\rho_{k+1} > \rho_k$;

end if
if $\max \left\{ \|c(x_k)\|, \|g_{\mu}(x_k) - J(x_k)^T \pi_k\| \right\} \leq \mu \tau_{\text{stop}}$ then exit;

end do
Chapter 3

Inexact Newton directions

3.1 Overview

The $j$-th iteration of a line-search or trust-region method for minimizing the primal-dual augmented Lagrangian requires the approximate solution of the approximate Newton equations

$$B_\mu(v_j; \rho_j) \Delta v_j = -\nabla \mathcal{M}_\mu(v_j; \pi^e_j, \rho_j),$$

with fixed barrier parameter $\mu$, and approximate Hessian,

$$B_\mu(v_j; \rho_j) = \begin{pmatrix} H_\mu(v_j) + 2J(x)^T D^{-1}_j J(x) & J(x)^T \\ J(x) & D_j \end{pmatrix}. \quad (3.2)$$

Many inexact Newton line-search methods compute $s_j = \Delta v_j$ by using a positive-definite matrix $\bar{B}_\mu(v_j; \rho_j)$ in (3.1) and apply an iterative method to the modified equations. If $\bar{B}_\mu(v_j; \rho_j)$ comes from an implicit modification of $B_\mu(v_j; \rho_j)$, then a second-order line-search method is possible.

The focus of this chapter is on obtaining $s_j$ from the trust-region subproblem

$$\min_s Q_j(s) \text{ subject to } \|N_j s\|_2 \leq \delta_j,$$

where $N_j^T N_j$ is positive-definite and

$$Q_j(s) = \nabla \mathcal{M}_\mu(v_j; \pi^e_j, \rho_j)^T s + \frac{1}{2} s^T B_\mu(v_j; \rho_j)s. \quad (3.4)$$

Because these methods must be used in the large-scale setting, direct methods, such as the More-Sorensen algorithm, are too expensive and iterative approaches are the only realistic option. The emphasis is on solving a specific instance of (3.3) and therefore the subscript $j$
will be ignored unless specified otherwise. Moreover, the approximate Hessian and gradient of the current primal-dual augmented Lagrangian are denoted by $B$ and $g$. The generic trust-region problem takes the form,

$$\min_{\bar{s}} \ Q(s) \ \text{subject to} \ \|Ns\|_2 \leq \delta. \tag{3.5}$$

The base point for the quadratic model is denoted $v = (x, \pi)$ and a global minimizer of (3.5) is $s^*$. Any computed solution will be denoted by $\bar{s}$.

### 3.2 Powell’s dogleg method

Powell’s dog-leg method [58] computes an approximate solution to (3.5) for positive-definite $B$, by searching along a path consisting of two linear segments. The segments connect the base point $v$, the Cauchy point $v^C$, and the Newton point $v^N = v + s^N$ where $s^N$ is the solution to the Newton equations $Bs^N = -g$. The Cauchy point takes the form $v^C = v + s^C$, where $s^C$ is the Cauchy step, which solves the one-dimensional minimization problem

$$\min_{\bar{s}} \ Q(s) \ \text{subject to} \ \|Ns\|_2 \leq \delta, \ s = -\alpha g. \tag{3.6}$$

If $\|Ns^N\|_2 \leq \delta$, then $\bar{s} = s^N$ and the new point is $v + s^N$. On the other hand, if $\|Ns^N\|_2 > \delta$, then the new point is of the form $v(\alpha) = v + s(\alpha)$ where $s(\alpha)$ is parameterized so that,

$$v(\alpha)|_{\alpha=0} = v, \quad v(\alpha)|_{\alpha=1/2} = v^C, \quad \text{and} \quad v(\alpha)|_{\alpha=1} = v^N.$$

Because $B$ is positive definite, it can be shown that $Q(s(\alpha))$ is strictly decreasing on $[0, 1]$. Therefore, when $\|Nv^N\|_2 > \delta$, the computed trust-region step is $\bar{s} = s(\bar{\alpha})$ with $\|Ns(\bar{\alpha})\|_2 = \delta$.

If $B$ is not positive-definite, then the $s^N$ may be an ascent direction for $Q(s)$ and hence $s(\alpha)$ may increase on $[1, 2]$. As a result, the classical dog-leg method is not appropriate for general matrices $B$.

### 3.3 Minimizing quadratic functions

A generalization of Powell’s dog-leg method to general quadratic functions $Q(s)$ has been proposed independently by Steihaug [61] and Toint [62]. The Steihaug-Toint method uses the conjugate-gradient method to generate a polygonal path on which $Q(s)$ strictly decreases. Like Powell’s dog-leg method, $\bar{s}$ is chosen to lie on this polygonal path.
The Steihaug-Toint CG method is discussed later in this chapter. First, it is necessary to introduce the conjugate-gradient method.

### 3.3.1 Conjugate-gradient method

The conjugate-gradient method is an iterative algorithm for solving positive-definite systems, or alternatively, for minimizing convex quadratic functions. Because the latter interpretation is more relevant to the trust-region subproblem, it will be considered first. The method generates a finite sequence of iterates \( \{s_k\} \) that satisfy

\[
\begin{align*}
  s_0 &= 0 \quad \text{and} \quad s_k = \arg\min_{s \in S_k} Q(s) \quad \text{for} \quad k \geq 1,
\end{align*}
\]

where \( S_k = \text{span}\{p_0, p_1, \ldots, p_k\} \) is a basis for the \( k \)-th Krylov subspace \( \mathcal{K}(B, p_0, k) \), and defined as

\[
\mathcal{K}(B, p_0, k) = \text{span}\left\{p_0, Bp_0, \ldots, B^{k-1}p_0\right\}.
\] (3.7)

The vectors \( p_i \) form a \( B \)-conjugate basis, i.e., \( p_i^T B p_j = 0 \) for all \( i \neq j \).

The sequence of minimizers \( \{s_k\} \) defines a polygonal path \( s(\alpha) \), parameterized so that \( s(1) = s^N \). Moreover, \( Q(s(\alpha)) \) is strictly decreasing on \([0, 1]\) and it can be shown that \( \|s(\alpha)\|_2 \) is strictly increasing on \([0, 1]\). An implementation of the conjugate-gradient method is given as Algorithm 3.1 below.

**Algorithm 3.1 Conjugate-gradient method**

Choose \( \eta \in (0, 1) \);

\[
\begin{align*}
  s_0 &= 0; \quad r_0 = g; \quad z_0 = r_0; \quad p_0 = -z_0; \\
  k &= 0;
\end{align*}
\]

while \( \|r_k\| \geq \eta\|r_0\| \) do

\[
\begin{align*}
  \alpha_k &= r_k^T z_k / p_k^T B p_k; \\
  s_{k+1} &= s_k + \alpha_k p_k; \\
  r_{k+1} &= r_k + \alpha_k B p_k; \\
  z_{k+1} &= r_{k+1}; \\
  \beta_k &= r_{k+1}^T z_{k+1} / r_k^T z_k; \\
  p_{k+1} &= -z_{k+1} + \beta_k p_k; \\
  k &= k + 1;
\end{align*}
\]

end do

In exact arithmetic, the sequence of residuals \( \{r_k\} \) consists of mutually orthogonal
vectors with $l$-th element
\[ \nabla Q(s_l) = g + Bs_l. \]
Moreover, $r_{k+1}^T p_j = 0$ for all $j = 0:k$. In theory, the conjugate-gradient algorithm will return the global minimizer of a convex quadratic of $n + m$ variables in at most (and possible much fewer than) $n + m$ iterations. For large problems, requiring $n + m$ iterations may be inefficient. Another difficulty is that each iteration undergoes rounding errors. The termination criterion in algorithm (3.1) is usually based on the Dembo, Eisenstat, Steihaug conditions for inexact Newton methods discussed in Chapter 1. The convergence rate of the conjugate gradient method is sensitive to the condition number of the matrix $B$, as the following theorem shows.

**Theorem 3.3.1.** If $B$ is symmetric positive-definite and the conjugate gradient is carried out on $Q(s) = g^T s + \frac{1}{2} s^T B s$, then
\[ \|s^* - s_k\|_B \leq 2\|s^* - s_0\|_B \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k, \]
where $\|s\|_B^2 = s^T B s$ for an arbitrary vector $s$ and $\kappa$ is the two-norm condition number of $B$.

### 3.3.2 The preconditioned conjugate-gradient method

The convergence rate of the conjugate-gradient method can be accelerated with a positive-definite preconditioner $\tilde{B} \approx B$ with Cholesky factorization $\tilde{B} = R^T R$, where $R$ is upper triangular. Consider the transformed quadratic function in the transformed variables $\tilde{s} = Rs$, i.e.,
\[ \tilde{Q}(\tilde{s}) = \tilde{g}^T \tilde{s} + \frac{1}{2} \tilde{s}^T \tilde{B} \tilde{s}, \quad \text{where} \quad \tilde{B} = R^{-T} B R^{-1}, \quad \tilde{g} = R^{-T} g. \quad (3.8) \]
If Algorithm 3.1 is applied to $\tilde{Q}(\tilde{s})$, the computations can be expressed in terms of the untransformed quantities and arranged so that only solves with $\tilde{B}$ are required. The resulting algorithm is called the *preconditioned conjugate-gradient* (PCG) method and is given in Algorithm 3.2 below.
Algorithm 3.2 Preconditioned conjugate-gradient method

Choose \( \eta \in (0, 1) \);
\( s_0 = 0; \; r_0 = g; \) Solve \( \bar{B}z_0 = r_0; \; p_0 = -z_0; \)
\( k = 0; \)

while \( \| r_k \| \geq \eta \| r_0 \| \) do
\( \alpha_k = r_k^T z_k / p_k^T B p_k ; \)
\( s_{k+1} = s_k + \alpha_k p_k ; \)
\( r_{k+1} = r_k + \alpha_k B p_k ; \)
Solve \( \bar{B}z_{k+1} = r_{k+1} ; \)
\( \beta_k = r_{k+1}^T z_{k+1} / r_k^T z_k ; \)
\( p_{k+1} = -z_{k+1} + \beta_k p_k ; \)
\( k = k + 1 ; \)
end do

The sequence \( \{ s_k \} \) produced from the preconditioned conjugate-gradient method increases in the elliptic norm \( \| \cdot \|_{\bar{B}} \), and not necessarily in \( \| \cdot \|_2 \). To see this recall that applying the conjugate-gradient algorithm to the transformed quadratic results in a sequence of iterates \( \{ \hat{s}_k \} \) that increases in two-norm. The fact that \( \hat{s}_k = Rs_k \) implies,
\[
\| \hat{s}_k \|_2^2 = s_k^T \bar{B} s_k = \| s_k \|_{\bar{B}}^2 .
\]

3.3.3 The Lanczos conjugate-gradient method

The conjugate-gradient method can also be derived from the Lanczos process for transforming a symmetric matrix in tridiagonal form. Rather than using a minimization paradigm, the Lanczos-CG method builds the solution to \( Bs = -g \) by projecting onto a sequence of expanding Krylov subspaces spanned by the Lanczos vectors.

Consider the \( k \)-th Krylov subspace,
\[
\mathcal{K}(B, g, k) = \text{span} \left\{ g, Bg, \ldots, B^{k-1} g \right\} .
\]

The minimization approach for the conjugate-gradient method constructs a \( B \)-conjugate basis for this space, whereas the Lanczos-CG method builds an orthonormal basis of Lanczos vectors, \( \{ q_0, q_1, \ldots, q_{k+1} \} \), positive scalars \( \gamma_0, \gamma_1, \ldots, \gamma_k \), and nonzero scalars \( \beta_1, \beta_2, \ldots, \beta_k \),
such that

$$T_k = \begin{pmatrix} 
\gamma_0 & \beta_1 \\
\beta_1 & \gamma_1 & \beta_2 \\
& \ddots & \ddots \\
& & \ddots & \beta_{k-1} \\
& & & \beta_k \gamma_{k-1} 
\end{pmatrix}, \quad \text{and} \quad BQ_k = Q_k T_k + \beta_k q_{k+1} e_{k+1}^T,$$

(3.10)

where $Q_k = (q_0 \ q_1 \ \cdots \ q_k)$. Projecting $Bs = -g$ onto $\mathcal{K}(B, g, k)$ yields the linear system

$$Q_k^T BQ_k w_k = \beta_0 e_1, \quad \text{where} \quad \beta_0 = \|g\|_2.$$

(3.11)

As $Q_k^T BQ_k$ is a positive-definite tridiagonal matrix, $w_k$ may be computed by finding the $LDL^T$ factorization of $T_k$ and performing forward and backward substitution. Given $w_k$, and assuming $s_0 = 0$, the next iterate is $s_{k+1} = Q_k w_k$, where $w_k$ satisfies $T_k w_k = \beta_0 e_1$. The $(k+1)$-st residual is,

$$r_{k+1} = B s_{k+1} + g$$

$$= B Q_k w_k - \beta_0 q_0$$

$$= (Q_k T_k + \beta_{k+1} q_{k+1} e_{k+1}^T) w_k - \beta_0 q_0$$

$$= Q_k T_k w_k + \beta_{k+1} q_{k+1} \alpha_k - \beta_0 q_0$$

$$= \beta_{k+1} q_{k+1} \alpha_k,$$

(3.12)

where $\alpha_k$ is the $(k+1)$-st entry of $w_k$, and hence $\|r_{k+1}\|_2 = \beta_{k+1} |\alpha_k|$. This provides a convenient termination test for the Lanczos-CG method.
Algorithm 3.3 Lanczos-CG method

Choose $\tau^* > 0$

$t_0 = -g; \quad \beta_0 = -\|d_0\|_2; \quad \alpha_{-1}; \quad \tau = -\beta_0; \quad k = -1;$

while $\tau > \tau^*$ do

$q_{k+1} = t_{k+1}/\beta_{k+1}; \quad k = k + 1;$

$\gamma_k = q_k^T B q_k;$

if $k = 0$ then

$s_k = 0;$

$l_k = 0; \quad p_k = q_k; \quad t_{k+1} = B q_k - \gamma k q_k;$

else

$l_k = \beta_k/d_{k-1}; \quad p_k = q_k - l_k p_{k-1}; \quad t_{k+1} = B q_k - \gamma_k q_k - \beta_k q_{k-1};$

end if

$d_k = \gamma_k - \beta_k l_k; \quad \alpha_k = -\beta_k \alpha_{k-1}/d_k;$

$s_{k+1} = s_k + \alpha_k p_k;$

$\beta_{k+1} = -\|t_{k+1}\|_2; \quad \tau = -\beta_{k+1} \alpha_k;$

end do

Observe that $p_k \in \text{span} \{q_i\}_{i=1}^k$ and therefore so is $s_k$. Because the Lanczos vectors are mutually orthogonal, $\|s_j\|_2 \leq \|s_{j+1}\|_2$ for $j = 1:k - 1$. It is also possible to derive a preconditioned version of the Lanczos-CG algorithm based on applying the standard algorithm to $B \hat{s} = -\hat{g}$ and rearranging the computation so that only solves with the positive-definite preconditioner are required.

The Lanczos reduction to tridiagonal form is the basis of many iterative solvers for symmetric systems, including MINRES and SYMMLQ. Rather than compute the $LDL^T$ of $T_k$, these algorithms compute an $LBL^T$ factorization that remains stable for symmetric indefinite systems.

3.4 The Steihaug-Toint CG method

The Steihaug-Toint method generates a sequence of approximate minimizers for problem (3.5) using the conjugate-gradient method. Here we emphasize the use of Algorithm 3.1 rather than Algorithm 3.3, which uses the Lanczos-CG method.

Unlike the standard conjugate-gradient method, a mechanism must be in place to
deal with abnormal termination. If the computed iterates exceed the trust-region boundary, or indefiniteness is detected, the Steihaug-Toint method terminates with a point on the trust-region boundary.

If the trust-region constraint is violated at $s_{j+1}$ and $\|s_j\| < \delta$, then the Steihaug-Toint method terminates with the unique point on the segment between $s_j$ and $s_{j+1} = s_j + p_j$. This point is found by computing $\bar{\alpha}$ such that

$$\|N(s_j + \bar{\alpha}p_j)\|_2 = \delta.$$ 

Squaring this equation and using the quadratic formula shows that $\bar{\alpha}$ satisfies

$$\bar{\alpha} = \frac{-s_j^T N^TNp_j + \sqrt{s_j^T N^TNp_j - \|Np_j\|^2_2(\|Ns_j\|^2_2 - \delta^2)}}{\|Np_j\|^2_2}. \quad (3.13)$$

In this case, the Steihaug-Toint CG method terminates with $\bar{s} = s_j + \bar{\alpha}p_j$ on the trust-region boundary. The algorithm is terminated in this way if $p_j^T Bp_j < 0$.

Because the Steihaug-Toint method is CG with additional termination criteria, it is possible to define a preconditioned version. The following pseudocode is based on the algorithm in [9]. The unpreconditioned version is recovered by choosing $B = N^TN$. 
Algorithm 3.4 Preconditioned Steihaug-Toint CG method

Choose $\eta \in (0, 1)$;
$s_0 = 0; r_0 = g; \text{Solve } \bar{B}z_0 = r_0; p_0 = -z_0$;
$k = 0$;

while $\|r_k\| \geq \eta\|r_0\|$ do
  $\kappa_k = p^T_k B p_k$;
  if $\kappa_k < 0$ then
    Find $\bar{\alpha}$ such that $\|s_k + \bar{\alpha}p_k\|_{\bar{B}} = \delta$;
    return $\bar{s} = s_k + \bar{\alpha}p_k$;
  end if
  $\alpha_k = r^T_k z_k / p^T_k B p_k$;
  if $\|s_k + \alpha_k p_k\| \geq \delta$ then
    Find $\bar{\alpha}$ such that $\|s_k + \bar{\alpha}p_k\|_{\bar{B}} = \delta$;
    return $\bar{s} = s_k + \bar{\alpha}p_k$;
  end if
  $s_{k+1} = s_k + \alpha_k p_k$;
  $r_{k+1} = r_k + \alpha_k B p_k$;
  Solve $\bar{B}z_{k+1} = r_{k+1}$;
  $\beta_k = r^T_{k+1} z_{k+1} / r^T_k z_k$;
  $p_{k+1} = -z_{k+1} + \beta_k p_k$;
  $k = k + 1$;
end do

return $\bar{s} = s_k$;

Despite its simplicity, a trust-region algorithm that uses the Steihaug-Toint point as a search direction converges to stationary points under mild conditions. It suffices to show that $Q(\bar{s}) \leq Q(s^C)$. The following preliminary lemma will be helpful.

**Lemma 3.4.1.** Suppose $s_j$ and $p_j$ are generated from the conjugate-gradient method. Then for any $\alpha > 0$, it holds that

$$Q(s_j + \alpha p_j) = Q(s_j) + Q(\alpha p_j).$$

**Proof.** Expanding $Q(s_j + \alpha p_j)$ as a Taylor polynomial yields,

$$Q(s_j + \alpha p_j) = Q(s_j) + \alpha \nabla Q(s_j)^T p_j + \frac{\alpha^2}{2} p_j^T B p_j,$$
and therefore it suffices to show that
\[ \nabla Q(s_j)^T p_j = g^T p_j. \]

For this, note that \( \nabla Q(s_j) = g + B s_j \) and \( s_j \in \text{span} \{p_0, \ldots, p_{j-1}\} \). Therefore,
\[ p_j^T B s_j = \sum_{i=0}^{j-1} \alpha_i p_j^T B p_i = 0, \]
implying that \( p_j^T \nabla Q(s_j) = p_j^T g + p_j^T B s_j = p_j^T g. \) By remarks above, this proves the lemma.

Lemma 3.4.1 is particularly important when \( p_j \) is a direction of negative curvature for \( B \). If \( p_j^T B p_j < 0 \), then, assuming without loss of generality, that \( g^T p_j < 0 \), it holds that
\[ Q(\alpha p_j) = \alpha g^T p_j + \frac{\alpha^2}{2} p_j^T B p_j < 0. \]

**Theorem 3.4.1.** The Steihaug-Toint CG method terminates at a point \( \bar{s} \) such that
\[ Q(\bar{s}) \leq Q(s^C). \]

**Proof.** If \( \bar{s} \) is the Cauchy point, then the result holds trivially. Therefore, assume that \( \bar{s} = s_j + \bar{\alpha} p_j \) for \( \bar{\alpha} > 0 \). If \( \bar{\alpha} = \alpha_j \), then \( \bar{s} \) is the vector \( s_{j+1} \) from the conjugate-gradient method, and \( s^C = s_1 \). This implies
\[ Q(\bar{s}) = Q(s_{j+1}) < Q(s_1) = Q(s^C), \]
which proves the result in this case. Next, suppose \( p_j^T B p_j > 0 \) but \( \|N(s_j + \alpha_j p_j)\|_2 > \delta \). Therefore, \( Q(s) \) is convex on \( S_{k+1} \) and strictly decreases on the segment connecting \( s_j \) and \( s_j + \alpha_j p_j \). If \( \bar{\alpha} > 0 \) is chosen so that \( \|\bar{s}\|_2 = \delta \), then
\[ Q(\bar{s}) < Q(s_j) < Q(s^C), \]
thereby proving the result in the case that the trust-region constraint is violated. Finally, suppose \( p_j^T B p_j < 0 \) in which case \( Q(\bar{\alpha} p_j) < 0 \). Lemma (3.4.1) implies that
\[ Q(\bar{s}) = Q(s_j) + Q(\bar{\alpha} p_j) < Q(s_j) < Q(s^C). \]
To compute $\alpha$ such that $\|N(s_j + \alpha p_j)\|_2 = \delta$ requires matrix-vector products with $N^T N$. In this case, the formula for $\alpha$ becomes, the formula for $\bar{\alpha}$ becomes,

$$\bar{\alpha} = -s_j^T \bar{B}p_j + \sqrt{s_j^T \bar{B}p_j - \|p_j\|_B^2 (\|s_j\|_B^2 - \delta^2)} / \|p_j\|_B^2.
$$ (3.14)

Many preconditioners are not easily represented as explicit matrices, making this formula impractical. Fortunately, each term can be recovered from the CG iterations. To begin, note that

$$\|p_k\|_B^2 = z_k^2 - 2\beta_k^{-1} z_k^T \bar{B}p_{k-1} + \beta_k^{-2}\|p_{k-1}\|_B^2 = z_k^T r_k - 2\beta_k^{-1} r_k^T p_{k-1} + \beta_k^{-2}\|p_{k-1}\|_B^2 = z_k^T r_k + \beta_k^{-2}\|p_{k-1}\|_B^2
$$ (3.15)

To compute $p_k^T \bar{B}s_k$, we use the formula for $s_k$ and take an inner-product with $\bar{B}p_k$, yielding,

$$p_k^T \bar{B}s_k = p_k^T \bar{B}s_{k-1} + \alpha_{k-1} p_k^T \bar{B}p_{k-1}.
$$ (3.16)

Replacing $p_k$ with $-z_k + \beta_{k-1} p_{k-1}$ and using the fact that

$$\bar{B}z_k = r_k \text{ and } s_{k-1} \in \text{span}\{p_0, \ldots, p_{k-1}\},
$$ (3.17)

transforms (3.16) into

$$p_k^T \bar{B}s_k = -r_k^T s_{k-1} + \beta_{k-1} p_{k-1}^T \bar{B}s_{k-1} + \alpha_{k-1} (r_k^T p_{k-1} + \beta_{k-1}\|p_{k-1}\|_B^2).
$$ (3.18)

This simplifies to

$$p_k^T \bar{B}s_k = \beta_{k-1} (p_{k-1}^T \bar{B}s_{k-1} + \alpha_{k-1}\|p_{k-1}\|_B^2).
$$ (3.19)

Finally, observe that $\|s_k\|_B^2$ is equal to

$$\|s_k\|_B^2 = \|s_{k-1}\|_B^2 + 2\alpha_{k-1} (p_{k-1}^T \bar{B}s_{k-1}) + \|p_{k-1}\|_B^2.
$$ (3.20)

To summarize, the step-length $\bar{\alpha}$ is available at little cost if the following formulas are used and updated during the CG iterations:

$$\|s_k\|_B^2 = \|s_{k-1}\|_B^2 + 2\alpha_{k-1} (p_{k-1}^T \bar{B}s_{k-1}) + \|p_{k-1}\|_B^2,
$$

$$\|p_k\|_B^2 = z_k^T r_k + \beta_k^{-2}\|p_{k-1}\|_B^2,
$$

$$s_k^T \bar{B}p_k = \beta_{k-1} (s_{k-1}^T \bar{B}p_{k-1} + \alpha_{k-1}\|p_{k-1}\|_B^2).
$$ (3.21)
Complications with the preconditioned Steihaug-Toint CG method

If \( \{ s_j \} \) is a sequence of PCG iterates generated with a nontrivial preconditioner and \( \|s_l\|_2 > \delta \), it may still be the case that \( \|s_l\|_2 < \delta \) for \( l > \hat{l} \). Therefore, it may be possible to obtain a better solution to the trust-region subproblem. The complication comes from embedding such a subproblem solver into a trust-region algorithm. The preconditioned Steihaug-Toint method implicitly scales the trust-region radius and as a result, the length of the solution of one subproblem may have little relation to the solution of another.

There are a few options to help deal with these theoretical issues. Although it is possible that the PCG iterates may reenter the two-norm trust-region, this is rarely observed in practice. One option then is to use the two-norm trust-region and hope for the best. A more theoretically sound option is to do away with the preconditioned Steihaug-Toint method altogether and rely on the standard PCG method instead. The output from PCG can serve as an “accelerator” direction for a two-dimensional trust-region subproblem,

\[
\minimize_{s \in \mathbb{R}^{n+m}} Q(s) \quad \text{subject to} \quad s \in \text{span} \left\{ s^C, \bar{s} \right\}, \quad \|s\|_2 \leq \delta.
\]

This two-dimensional problem can be solved with a direct method (such as the Moré-Sorensen algorithm) and its output used for the trust-region step. The PCG method should be terminated if negative curvature is detected, in which case, \( \bar{s} \) can be taken as a linear combination of the last PCG iterate and the direction of negative curvature, as in the Steihaug-Toint method.

3.5 Other research

While the Steihaug-Toint CG method can form the basis for a globally convergent trust-region algorithm, its lack of accuracy in the constrained case may prevent convergence to second-order points. Many methods have been proposed that yield a more accurate solution on the trust-region boundary. A common theme among many of these algorithms involves using the Lanczos process to estimate the eigenpair of \( B \) corresponding to the most negative eigenvalue. Examples include the Generalized Lanczos Trust-Region (GLTR) method of Gould, Luicidi, Roma, and Toint [38], and the sequential subspace minimization methods of Erway and Gill [12, 13, 14] and Hager and Park [39, 40]. Directions of negative curvature can be used to develop second-order trust-region methods, or can be used in conjunction with a second-order line-search method.
Chapter 4

Preconditioners for the doubly augmented system

4.1 Overview

This chapter focuses on preconditioning strategies for truncated Newton methods, which approximately solve systems of the form

\[ B_\mu(v; \rho) \Delta v = -\nabla \mathcal{M}_\mu(v; \pi^e, \rho), \]

where \( v = (x, \pi) \), or equivalently,

\[
\begin{pmatrix}
H + 2J^T D^{-1} J & J^T \\
J & D
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
\Delta \pi
\end{pmatrix} = -\begin{pmatrix}
g - J^T (2\pi - \tilde{\pi}) \\
c + D(\pi - \pi^e)
\end{pmatrix},
\]

where \( H, J, D, g \) and \( c \) denote \( H_\mu(x, \pi), J(x), D_\rho, g_\mu(x), \) and \( c(x) \) respectively. The penalty parameter, barrier parameter, multiplier estimate, and \( (x, \pi) \) are assumed to be fixed. As in the previous chapter, \( B_\mu(v; \rho) \) is denoted by \( B \) and \( \bar{B} \) represents the preconditioner. Building a practical, positive definite preconditioner for these systems is a nontrivial, but necessary task. The method is based on the concept of constraint preconditioning.

The quality of \( \bar{B} \) as a preconditioner for \( B \) is determined by how close the eigenvalues of \( \bar{B}^{-1}B \) are to one. These eigenvalues are also the generalized eigenvalues of the generalized eigenvalue problem, \( \bar{B}x = \lambda Bx \). Analyzing the quality of a preconditioner requires the computation of bounds on these eigenvalues.
4.2 Constraint preconditioners for KKT systems

Preconditioners for the doubly augmented system will be based on constraint preconditioning discussed by Lukšan and Vlček [47] for solving systems with symmetric KKT matrices of the form,

$$K = \begin{pmatrix} H & J^T \\ J & -D \end{pmatrix},$$

(4.2)

for positive semidefinite matrix $D$. The emphasis here is on the approximation properties of the preconditioner, and not on the iterative method for which it is used. A constraint preconditioner for $K$ has the form,

$$\bar{K} = \begin{pmatrix} \bar{H} & J^T \\ J & -D \end{pmatrix},$$

(4.3)

for some matrix $\bar{H}$ that approximates $H$. In optimization, $J$ represents the linearized constraint Jacobians, which is where the “constraint” in “constraint preconditioning” comes from. Constraint preconditioners have very attractive theoretical properties. They are analyzed by Keller, Gould, and Wathen [45] and the following theorem summarizes the key results.

**Theorem 4.2.1.** Consider the $(n+m) \times (n+m)$ matrix $K$ in (4.2) with $D = 0$, rank($J$) = $m$, and null($J$) spanned by the columns of an $n \times (n-m)$ matrix $Z$. If $\bar{K}$ is the constraint preconditioner in (4.3), then the set of eigenvalues for $\bar{K}^{-1}K$ consist of $2m$ unit eigenvalues and $n-m$ eigenvalues that are the same as the generalized eigenvalues for $Z^T\bar{H}Zv = \lambda Z^T\bar{H}Zv$.

**Proof.** See Theorem 2.1 in [45].

Therefore, the burden in choosing a good constraint preconditioner lies in choosing a good approximation to $H$. Matrices of this form arise in interior-point methods. Constraint preconditioners for KKT systems arising in interior-point methods are proposed by Bergamaschi, Gondzio, and Zilli [5]. Their results may be summarized as follows.

**Theorem 4.2.2.** Suppose $J$ has full row rank. Then at least $m$ eigenvalues of $\bar{K}^{-1}K$ are one. For each of the remaining $n$ eigenvalues, there exist vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ such that

$$\lambda_i = \frac{x^THx + y^TDy}{x^THx + y^TDy} \geq 0.$$
Moreover,
\[
\min \left\{ \lambda_\ell (\bar{H}^{-1}H), 1 \right\} \leq \lambda_\ell (\bar{K}^{-1}K) \leq \lambda_u (\bar{K}^{-1}K) \leq \max \left\{ \lambda_u (\bar{H}^{-1}H), 1 \right\}.
\]

**Remark 4.2.1.** It turns out that the full-rank assumption on \( J \) is not required to prove the theorem. The proof of this more general result is provided below.

**Proof.** Consider the nonsingular \((n + m) \times (n + m)\) block matrix,
\[
U = \begin{pmatrix} I & 0 \\ D^{-1}J & I \end{pmatrix},
\]
and let \( \bar{K}_U = U^T \bar{K} U \) and \( K = U^T K U \). Then the eigenvalues of \( \bar{K}^{-1}K \) are the same as the eigenvalues of \( \bar{K}_U^{-1}K_U \). Furthermore,
\[
K_U = \begin{pmatrix} H + J^T D^{-1}J & 0 \\ 0 & -D \end{pmatrix} \quad \text{and} \quad \bar{K}_U = \begin{pmatrix} \bar{H} + J^T D^{-1}J & 0 \\ 0 & \bar{I} \end{pmatrix},
\]
which implies
\[
\bar{K}_U^{-1}K_U = \begin{pmatrix} (\bar{H} + J^T D^{-1}J)^{-1}(K + J^T D^{-1}J) & 0 \\ 0 & \bar{I} \end{pmatrix}.
\]
Therefore, \( \lambda(\bar{K}^{-1}K) \) includes at least \( m \) unit eigenvalues. The remaining \( n \) eigenvalues satisfy the generalized eigenvalue problem,
\[
(H + J^T D^{-1}J)x = \lambda(\bar{H} + J^T D^{-1}J)x,
\]
and hence satisfy,
\[
\lambda = \frac{x^T (H + J^T D^{-1}J)x}{x^T (H + J^T D^{-1}J)x}.
\]
As \( D \) is positive-definite by assumption, there exists \( y \in \mathbb{R}^m \) such that \( Dy = Jx \). Replacing \( Jx \) by \( Dy \) shows
\[
\lambda = \frac{x^T Hx + y^T Dy}{x^T Hx + y^T Dy}
\]
as asserted. For the eigenvalue bounds, first note that the inequalities hold trivially for the unit eigenvalues. For the non-unit eigenvalues, let
\[
\lambda_{\min} \triangleq \lambda_{\min}(\bar{H}^{-1}H), \quad \lambda_{\max} \triangleq \lambda_{\max}(\bar{H}^{-1}H), \quad \text{and} \quad \delta = y^T Dy.
\]
Then
\[
\lambda_{\min} x^T \bar{H}x + \delta \leq x^T Hx + \delta \leq \lambda_{\max} x^T \bar{H}x + \delta,
\]
and therefore
\[
\min \{\lambda_{\text{min}}, 1\} (x^T \tilde{H} x + \delta) \leq x^T H x + \delta \leq \max \{\lambda_{\text{max}}, 1\} (x^T \tilde{H} x + \delta).
\]
Dividing both sides by the positive quantity \(x^T \tilde{H} x + \delta\) yields the eigenvalue bounds.

Theorem 4.2.2 immediately implies that the quality of the constraint preconditioner for regularized KKT systems is determined by how well \(\tilde{H}\) approximates \(H\).

### 4.3 Block factorization preconditioners

Traditionally, preconditioners for KKT matrices arising in PDE-constrained optimization are based on approximate block factorizations of unregularized KKT equations as in equation (4.2). The structure of \(K\) is particularly important, because it provides a way of utilizing sophisticated PDE preconditioners. Both PLTMG and LNKS use block factorization preconditioning strategies that can be expressed by,

\[
\tilde{K} = \begin{pmatrix}
\tilde{H} & \tilde{J}^T \\
\tilde{J} & 0
\end{pmatrix},
\]

where \(\tilde{H} \approx H\) and \(\tilde{J} \approx J\). To begin, consider \(K\) in partitioned form where the state and control portions of \(H\) and \(J\) are explicit, i.e.,

\[
K = \begin{pmatrix}
H_{yy} & H_{yu} & J_y^T \\
H_{uy} & H_{uu} & J_u^T \\
J_y & J_u & 0
\end{pmatrix},
\]

Rearranging yields the matrix,

\[
\tilde{K} = \begin{pmatrix}
H_{yy} & J_y^T & H_{yu} \\
J_y & 0 & J_u \\
H_{uy} & J_u^T & H_{uu}
\end{pmatrix},
\]

that satisfies the key property that

\[
\tilde{K}/K_{yy} = Z^T H Z, \quad \text{where} \quad K_{yy} = \begin{pmatrix}
H_{yy} & J_y^T \\
J_y & 0
\end{pmatrix},
\]

and the columns of \(Z\) form a basis for \(\text{null}(J)\), with the particular representation,

\[
Z = \begin{pmatrix}
-J_y^{-1} J_u \\
I
\end{pmatrix}.
\]
PLTMG and LNKS solve the Newton equations approximately using nonsymmetric Lanczos techniques. The preconditioned equations take the form,

$$
\begin{pmatrix}
\bar{H} & \bar{J}^T \\
\bar{J} & 0
\end{pmatrix}
\begin{pmatrix}
z_x \\
z_\pi
\end{pmatrix}
= 
\begin{pmatrix}
r_x \\
r_\pi
\end{pmatrix}.
$$

(4.10)

In the descriptions that follow, let $z_x$ and $r_x$ be partitioned into states and controls, giving $z_x = (z_y, z_u)$ and $r_x = (r_y, r_u)$.

### 4.3.1 Other approaches

PLTMG uses a block-symmetric Gauss-Seidel scheme for $\bar{K}$. Consider the linear system,

$$
\begin{pmatrix}
H_{yy} & J_y^T & H_{yu} \\
J_y & 0 & J_u \\
H_{uy} & J_u^T & H_{uu}
\end{pmatrix}
\begin{pmatrix}
z_y \\
z_\pi \\
z_u
\end{pmatrix}
= 
\begin{pmatrix}
r_y \\
r_\pi \\
r_u
\end{pmatrix}.
$$

(4.11)

The symmetric Gauss-Seidel scheme proceeds by solving the following systems

$$
\begin{pmatrix}
H_{yy} & J_y^T \\
J_y & 0
\end{pmatrix}
\begin{pmatrix}
w_y \\
w_\pi
\end{pmatrix}
= 
\begin{pmatrix}
r_y \\
r_\pi
\end{pmatrix},
$$

(4.12)

$$
H_{uu}z_u = r_u - H_{uy}w_y - J_u^T w_\pi,
$$

(4.13)

$$
\begin{pmatrix}
H_{yy} & J_y^T \\
J_y & 0
\end{pmatrix}
\begin{pmatrix}
z_y \\
z_\pi
\end{pmatrix}
= 
\begin{pmatrix}
r_y \\
r_\pi
\end{pmatrix} - 
\begin{pmatrix}
H_{yu} \\
J_u
\end{pmatrix} z_u,
$$

(4.14)

(4.15)

where $w_y \in \mathbb{R}^{n_y}$ and $w_\pi \in \mathbb{R}^{n_\pi}$ are auxiliary vectors. Taking note of the fact that solving with $K_{yy}$ can be performed by solving with $J_y$, the equations can be written as

$$
J_yw_y = r_\pi,
$$

$$
J_y^T w_\pi = r_y - H_{yy}w_y,
$$

$$
H_{uu}z_u = r_u - H_{uy}w_y - J_u^T w_\pi,
$$

(4.16)

$$
J_yz_y = r_\pi - J_u z_u,
$$

$$
J_y^T z_\pi = r_y - H_{yu}z_u - H_{yy}z_y.
$$

Therefore, the bottleneck is performing solves with $H_{uu}$ and $J_y$. The $n_u \times n_u$ matrix $H_{uu}$ is positive definite and diagonally dominant, and hence it can be approximated in many ways. Furthermore, $J_y$ and $J_y^T$ are PDE stiffness matrices for which many general purpose, or
problem specific, preconditioners exist. Ignoring the details for a moment, let \( \bar{J}_y \) and \( \bar{H}_{uu} \) denote these preconditioners. Replacing \( J_y \) and \( H_{uu} \) with their preconditioners in (4.16) is equivalent to solving the following system exactly,

\[
\begin{pmatrix}
H_{yy} & H_{yu} \\
H_{uy} & \bar{H}_{uu} + K_{uy} K_{yy}^{-1} K_{yu} \\
\bar{J}_y & J_u \\
0 & 0
\end{pmatrix}
\begin{pmatrix}
\bar{J}_y^T \\
0 \\
J_u^T
\end{pmatrix}
\begin{pmatrix}
\bar{J}_y \\
0
\end{pmatrix}
= \begin{pmatrix} r_y \\ r_u \end{pmatrix},
\]

where \( K_{yy} = (H_{uy} J_u^T) \). The following theorem proves this explicitly.

**Theorem 4.3.1.** Solving \( \bar{K} z = r \) with equation (4.17) is equivalent to the symmetric Gauss-Seidel scheme with \( H_{uu} \) and \( J_y \) replaced by \( \bar{H}_{uu} \) and \( \bar{J}_y \).

**Proof.** Rearranging (4.17) yields

\[
\begin{pmatrix}
\bar{K}_{yy} & K_{yu} \\
K_{uy} & \bar{H}_{uu} + K_{uy} K_{yy}^{-1} K_{yu}
\end{pmatrix}
\begin{pmatrix}
\bar{K}_{yy} & K_{yu} \\
K_{uy} & \bar{H}_{uu} + K_{uy} K_{yy}^{-1} K_{yu}
\end{pmatrix}
\begin{pmatrix}
z_y \\
z_u
\end{pmatrix}
= \begin{pmatrix} r_y \\ r_u \\ 0 \\ 0 \end{pmatrix},
\]

where \( K_{yy} = (H_{yy} J_y^T) \).

Multiplying on the left by the block matrix

\[
\begin{pmatrix} I & 0 \\ -K_{uy} \bar{K}_{yy}^{-1} & I \end{pmatrix}
\]

and letting \( w = (w_y, w_\pi) \), the system

\[
\bar{K}_{yy} \begin{pmatrix} w_y \\ w_\pi \end{pmatrix} = \begin{pmatrix} z_y \\ z_u \end{pmatrix},
\]

gives the equivalent upper-triangular block system,

\[
\begin{pmatrix} \bar{K}_{yy} & K_{yu} \\ 0 & \bar{H}_{uu} \end{pmatrix}
\begin{pmatrix}
z_y \\
z_\pi \\
z_u
\end{pmatrix}
= \begin{pmatrix} z_y \\ z_\pi \\
z_u \\
r_u - K_{uy} w \end{pmatrix}.
\]

The steps for obtaining \( z \) are precisely those of (4.12) with \( \bar{K}_{yy} \) and \( \bar{H}_{uu} \) replacing \( K_{yy} \) and \( H_{uu} \), thus yielding the result.

Of course the method above only works if \( J_y \) is nonsingular, and hence \( J \) must have full row-rank. In addition, the preconditioner \( \bar{J}_y \) must be nonsingular. In PLTMG \( \bar{J}_y \) comes from an algebraic multigrid method [3, 4].
Next, we show that the preconditioning approach in PLTMG is a special case of those developed for LNKS. Both of these methods turn out to be full-space, reduced-space hybrids in which a positive-definite preconditioner for the reduced Hessian is maintained.

In the case of PLTMG, this reduced space preconditioner is

$$H_Z = H_{uu}.$$  \hspace{1cm} (4.20)

**The LNKS approach**

The LNKS preconditioners are based on approximate block factorizations. Using the representation of $H_Z$ as a Schur complement, $K$ can be decomposed as

$$
\begin{pmatrix}
H_{yy} & H_{yu} & J_{y}^T \\
H_{uy} & H_{uu} & J_{u}^T \\
J_{y} & J_{u} & 0
\end{pmatrix} =
\begin{pmatrix}
H_{yy} J_y^{-1} & 0 & I \\
H_{uy} J_y^{-1} & I & J_u^T J_y^{-1} \\
0 & 0 & H_Z
\end{pmatrix}
\begin{pmatrix}
J_y & J_u & 0 \\
0 & H_{uu} - H_{yy} J_y^{-1} J_u & J_y^T \\
0 & 0 & H_{uu}
\end{pmatrix}. \hspace{1cm} (4.21)
$$

Replacing $J_y$ by a suitable preconditioner $\tilde{J}_y$ and $H_Z$ with a positive-definite preconditioner $\bar{H}_Z$ leads to a matrix $\tilde{K}$ with the same structure as the matrix $\bar{K}$ from PLTMG, with the more generic reduced space preconditioner.

### 4.4 Solving the doubly augmented system inexactly

Consider the parameterized linear equations, that define Newton’s method for minimizing the primal-dual augmented Lagrangian, i.e.,

$$K_\nu \begin{pmatrix}
\Delta x \\
\Delta \pi
\end{pmatrix} = -b_\nu,$$

where

$$K_\nu = \begin{pmatrix}
H + (1 + \nu) J^T D^{-1} J & \nu J^T \\
\nu J & \nu D
\end{pmatrix}, \text{ and } b_\nu = \begin{pmatrix}
b_x + (1 + \nu) J^T D^{-1} b_\pi \\
\nu b_\pi
\end{pmatrix}.$$  \hspace{1cm} (4.23)

For $\nu = 1$, Forsgren, Gill, and Griffin [18] consider iterative methods for solving (4.22) that utilize positive definite preconditioners. As mentioned above, $K_\nu$ may not be positive-definite and so they propose using MINRES and SYMMLQ [53], which are suitable for solving general symmetric systems. As these iterative methods are only guaranteed to converge with positive definite preconditioners, $\bar{H}$ is chosen so that $\bar{H} + J^T D^{-1} J$ is positive-definite. This defines a positive definite preconditioner,

$$\tilde{K}_\nu = \begin{pmatrix}
\bar{H} + (1 + \nu) J^T D^{-1} J & \nu J^T \\
\nu J & \nu D
\end{pmatrix}. \hspace{1cm} (4.24)$$
The following theorem is a direct consequence of this and generalizes Theorem 4.2.2.

**Theorem 4.4.1.** At least \( m \) of the eigenvalues of \( \bar{K}^{-1}_{\nu}K_{\nu} \) are one and the remaining eigenvalues satisfy

\[
\lambda(\bar{K}^{-1}_{\nu}K_{\nu}) = \frac{x^T(H + J^TD^{-1}J)x}{x^T(H + J^TD^{-1}J)x} \geq 0
\]

for some \( x \in \mathbb{R}^n \). Moreover,

\[
\min \left\{ \lambda_\ell(\bar{H}^{-1}H), 1 \right\} \leq \lambda(\bar{K}^{-1}_{\nu}K_{\nu})) \leq \max \left\{ \lambda_u(\bar{H}^{-1}H), 1 \right\}.
\]

**Proof.** This is identical to the proof of Theorem 4.2.2. \(\Box\)

During the course of the iterations, linear systems of the form \( \bar{K}_\nu z = r \) must be solved. This system is replaced by the equivalent system,

\[
\begin{pmatrix}
H & J^T \\
J & -D
\end{pmatrix}
\begin{pmatrix}
z_x \\
-z_\pi
\end{pmatrix} =
\begin{pmatrix}
r_x - J^TD^{-1}r_\pi \\
r_\pi
\end{pmatrix},
\] (4.25)

which can be solved with a direct method. In the large-scale setting, this requires solving a very large symmetric indefinite system which may be too expensive. In this case, such an approach is prohibitively expensive, creating the need for alternatives.

One alternative is based on the fact that \( K_{\nu} \) can be expressed as

\[
K_{\nu} = \begin{pmatrix}
H_D + \nu J^TD^{-1}J & \nu J^T \\
\nu J & \nu D
\end{pmatrix}, \text{ where } H_D = H + J^TD^{-1}J.
\] (4.26)

Consider approximating \( H_D \) with a positive-definite matrix of the form

\[
P_G = P + A^TG^{-1}A.
\] (4.27)

In the case of \( \nu = 1 \), this will yield a positive-definite preconditioner,

\[
\tilde{K}_{\nu} = \begin{pmatrix}
P_G + \nu J^TD^{-1}J & \nu J^T \\
\nu J & \nu D
\end{pmatrix}.
\] (4.28)

**Theorem 4.4.2.** For \( \nu = 1 \), the matrix \( \tilde{K}_{\nu} \) in (4.28) is positive-definite if \( P_G \) is positive-definite.

**Proof.** If \( U \) is the nonsingular matrix,

\[
U = \begin{pmatrix}
I & -J^TD^{-1} \\
0 & I
\end{pmatrix}
\]
then
\[ U\bar{K}_\nu U^T = \begin{pmatrix} P_G & 0 \\ 0 & \nu D \end{pmatrix}, \]
and the result follows from Sylvester’s Law of Inertia.

The proof of the previous result immediately implies eigenvalue bounds for \( \bar{K}_\nu^{-1}K_\nu \).

**Corollary 4.4.1.** At least \( m \) eigenvalues of \( \bar{K}_\nu^{-1}K_\nu \) are one and the remaining \( n \) eigenvalues satisfy
\[
\lambda(\bar{K}_\nu^{-1}K_\nu) = \frac{x^T H_D x}{x^T P_G x}
\]
for some \( x \in \mathbb{R}^n \).

Therefore, the quality of \( K_\nu \) as a preconditioner consists entirely of how well \( P_G \) approximates \( H_D \). The matrix \( U \) used in the proof of Theorem 4.4.2 may be used to solve the preconditioner equations. In particular, multiplying \( \bar{K}_\nu z = r \) on the left by \( U \) yields
\[
\begin{pmatrix} P_G & 0 \\ \nu J & \nu D \end{pmatrix} \begin{pmatrix} z_x \\ z_\pi \end{pmatrix} = \begin{pmatrix} r_x - J^T D^{-1} r_\pi \\ r_\pi \end{pmatrix}.
\]

To solve this block lower-triangular system, \( P_G z_\pi = r_x - J^T D^{-1} r_\pi \) is solved for \( z_\pi \) and \( z_x \) is computed using \( z_\pi = D^{-1}(r_\pi - J z_x) \). It follows that if \( P_G \) can be solved efficiently, then \( \bar{K}_\nu \) will also be easy to apply.

### 4.5 Preconditioners for condensed matrix

With the preconditioning strategy specified, two questions remain: the choice of \( P_G \), and the method used to solve with it. Recall that \( P_G \) has the condensed structure,
\[
P_G = P + A^T G^{-1} A,
\]
where \( P \) is an \( n \times n \) symmetric matrix, \( A \) is a \( m \times n \) matrix, and \( G \) is an \( m \times m \) symmetric positive-definite matrix. The purpose of the next chapter will be to implicitly and dynamically build \( P_G \) with incomplete factorizations. For the remainder of this chapter, it is assumed that \( P_G \) is formed. The next two sections will focus on methods for solving with \( P_G \).
4.5.1 A direct approach

The direct approach for solving

\[ PGz_x = r_x - J^T D^{-1} r_{\pi} \] (4.31)

is based on a method often seen in projected conjugate-gradient methods [37]. Solving equation (4.31) is also called the normal equations approach. The augmented system approach solves a larger, yet more sparse system and is discussed in the following theorem.

**Theorem 4.5.1.** Let \( P_G \) denote the matrix \( P + A^T G^{-1} A \) where \( P \in \mathbb{R}^{n \times n} \), \( G \in \mathbb{R}^{m \times m} \), and \( A \in \mathbb{R}^{m \times n} \). Then the solution to the normal equations (4.31) also satisfies the regularized KKT system,

\[
\begin{pmatrix}
P & A^T \\
A & -G
\end{pmatrix}
\begin{pmatrix}
z_x \\
w_{\pi}
\end{pmatrix} = \begin{pmatrix}
r_x - J^T D^{-1} r_{\pi} \\
0
\end{pmatrix},
\] (4.32)

where \( w_{\pi} \in \mathbb{R}^m \) is an auxiliary vector.

**Proof.** As \( G \) is positive-definite, there exists a \( w_{\pi} \in \mathbb{R}^m \) such that \( Gw_{\pi} = Az_x \), where \( z_x \) satisfies (4.31). It follows that

\[ PGz_{\pi} = (P + A^T G^{-1} A)z_x = Pz_x + A^T w_{\pi}. \]

Using this with the choice for \( w_{\pi} \) above yields equation (4.32). \( \square \)

The following algorithm applies the preconditioner.

**Algorithm 4.1** Solve \( K_{\nu} z = r \) for \( \nu = 1 \)

Choose \( P_G = P + A^T G^{-1} A \);

Solve KKT system,

\[
\begin{pmatrix}
P & A^T \\
A & -G
\end{pmatrix}
\begin{pmatrix}
z_x \\
w_{\pi}
\end{pmatrix} = \begin{pmatrix}
r_x - J^T D^{-1} r_{\pi} \\
0
\end{pmatrix};
\]

Compute \( z_{\pi} = D^{-1}(r_{\pi} - Jz_x) \);

Set \( z = (z_x, z_{\pi}) \);

**Notation.** Let the regularized KKT matrix in algorithm (4.1) be denoted by

\[ P = \begin{pmatrix} P & A^T \\ A & -G \end{pmatrix}, \]
where the matrix is partitioned with respect to the state and control variables. Then

\[ P = \begin{pmatrix} P_{yy} & P_{yu} \\ P_{uy} & P_{uu} \end{pmatrix}, \quad A = \begin{pmatrix} A_y \\ A_u \end{pmatrix}, \quad P_{yy} = \begin{pmatrix} P_{yy} & A_y^T \\ A_y & -G \end{pmatrix}, \]

and \( P_{uy} = (P_{uy} A_u^T). \)

### 4.5.2 A symmetric block Gauss-Seidel approach

A related strategy is based on the symmetric Gauss-Seidel scheme used in PLTMG.

Consider the expanded system from algorithm (4.1),

\[
\begin{pmatrix} P_{yy} & P_{yu} & A_y^T \\ P_{uy} & P_{uu} & A_u^T \\ A_y & A_u & -G \end{pmatrix}
\begin{pmatrix} z_y \\ z_u \\ w_x \end{pmatrix} = \begin{pmatrix} r_y - J_y^T D^{-1} r_x \\ r_u - J_u^T D^{-1} r_x \\ 0 \end{pmatrix}
\]

(4.33)

where \( z_x = (z_y, z_u) \) and \( r_x = (r_y, r_u) \). As in the PLTMG preconditioning strategy, the application of a symmetric block Gauss-Seidel scheme with respect to the reordered system:

\[
\begin{pmatrix} P_{yy} & A_y^T & P_{yu} \\ A_y & -G & A_u \\ P_{uy} & A_u^T & P_{uu} \end{pmatrix}
\begin{pmatrix} z_y \\ z_u \\ w_x \end{pmatrix} = \begin{pmatrix} r_y - J_y^T D^{-1} r_x \\ r_u - J_u^T D^{-1} r_x \\ 0 \end{pmatrix}
\]

(4.34)

is equivalent to a direct method with matrix

\[
\bar{P} = \begin{pmatrix} P_{yy} & P_{yu} & A_y^T \\ P_{uy} & P_{uu} & P_{uy} P_{yy}^{-1} P_{yu} \\ A_y & A_u & -G \end{pmatrix}.
\]

(4.35)

To show that \( \bar{P} \) can be used within algorithm (4.1), it must be shown that \( \bar{P} + A^T G^{-1} A \) is positive-definite, where

\[
\tilde{P} = \begin{pmatrix} P_{yy} & P_{yu} \\ P_{uy} & P_{uu} + P_{uy} P_{yy}^{-1} P_{yu} \\ A_y & A_u \end{pmatrix}.
\]

(4.36)

The following simple result establishes this fact.

**Theorem 4.5.2.** Let \( P \) be a symmetric \( n \times n \) matrix, \( G \) be an \( m \times m \) symmetric positive definite matrix, and \( A \) an \( m \times n \) matrix. Then

\[
P = \begin{pmatrix} P & A^T \\ A & -G \end{pmatrix}
\]

satisfies \( \text{In}(P) = (n, m, 0) \) if and only if \( P + A^T G^{-1} A \) is positive-definite.
**Proof.** Let $U$ denote the matrix

$$U = \begin{pmatrix} I & A^T G^{-1} \\ 0 & I \end{pmatrix},$$

and observe that

$$U P U^T = \begin{pmatrix} P + A^T G^{-1} A & 0 \\ 0 & -G \end{pmatrix}.$$ 

Sylvester’s Law of Inertia implies that

$$\text{In}(P) = \text{In}(P + A^T G^{-1} A) + \text{In}(-G),$$

and since $G$ is positive definite, $\text{In}(P) = (n, m, 0)$ if and only if

$$\text{In}(P + A^T G^{-1} A) = (n, m, 0) - (0, m, 0) = (n, 0, 0),$$

or equivalently when $P + A^T G^{-1} A$ is positive definite. □

This result implies that if $\text{In}(\bar{P}) = (n, m, 0)$, then $\bar{P}$ can be used in place of $P$ in the algorithm.

**Theorem 4.5.3.** If $P_{uu}$ is positive-definite, then the matrix $\bar{P}$ of (4.35) has inertia $(n, m, 0)$.

**Proof.** Because $P + A^T G^{-1} A$ is positive definite by assumption then so is $P_{yy} + \bar{A}_y^T G^{-1} A_y$. Sylvester’s Law of Inertia therefore implies that,

$$\text{In}(P_{yy}) = \text{In}(P_{yy} + A_y^T G^{-1} A_y) + \text{In}(-G),$$

$$= (n_y, 0, 0) + (0, m, 0),$$

$$= (n_y, m, 0).$$

As the Schur complement of $P_{yy}$ in

$$\begin{pmatrix} P_{yy} & P_{yu} \\ P_{uy} & P_{uu} + P_{uy} P_{yy}^{-1} P_{yu} \end{pmatrix},$$

is the positive-definite matrix $P_{uu}$, another application of Sylvester’s law of inertia implies

$$\text{In}(\bar{P}) = \text{In}(P_{yy}) + \text{In}(P_{uu}) = (n_y, m, 0) + (n_u, 0, 0) = (n, m, 0)$$

□

Although the previous theorem places a restriction on $P_{uu}$, it is not an unreasonable one. In many cases, $H_{uu}$ is positive definite and hence can be used for $P_{uu}$. 
4.5.3 An iterative approach

As $P_G$ is symmetric positive definite, it is possible to apply the preconditioned conjugate-gradient method to the condensed preconditioner. In this setting, the preconditioned conjugate-gradient algorithm has outer and inner iterations, where the inner iterations correspond to the application of the preconditioner.

In order for the inner-iterations to be a viable option, it must be the case that $\tilde{z}^T r > 0$, where $\tilde{z} \approx z$ denotes the approximate solution of $\tilde{K}_\mu z = r$. To show that this is the case, suppose that $k$ iterations of Lanczos-CG are applied to

$$P_G z_x = r_x - J^TD^{-1}r_\pi.$$ 

Recall that the $n \times (k+1)$ matrix of Lanczos vectors is denoted by $Q_k$. After $k$ iterations, the solution to the normal equations projected onto $Q_k$ will be denoted by $w_x$ and solves,

$$Q_k^T P_G Q_k w_x = Q_k^T (r_x - J^TD^{-1}r_\pi). \quad (4.37)$$

Letting $\tilde{z}_x = V_k w_x$, it follows that

$$\bar{z}_\pi = D^{-1}(r_\pi - J\tilde{z}_x). \quad (4.38)$$

It is shown below that this method for finding $\tilde{z}$ may be interpreted as a projection method.

**Theorem 4.5.4.** The vector $\tilde{z} = (\tilde{z}_x, \bar{z}_\pi) \approx z$ computed using CG on $\tilde{K}_\mu z = r$ may be written as

$$\tilde{z} = S_k (S_k^T \tilde{K}_\mu S_k)^{-1} S_k^T r, \quad \text{where } S_k = \begin{pmatrix} Q_k & 0 \\ 0 & I \end{pmatrix}.$$ 

**Proof.** First note that because $Q_k$ has linearly independent columns, $S_k$ does as well. Observe that $(w_x, \tilde{z}_\pi)$ solves the system,

$$\begin{pmatrix} Q_k^T P_G Q_k & 0 \\ JQ_k & D \end{pmatrix} \begin{pmatrix} w_x \\ \tilde{z}_\pi \end{pmatrix} = \begin{pmatrix} Q_k^T(r_x - J^TD^{-1}r_\pi) \\ r_\pi \end{pmatrix}.$$ 

Multiplying on the left by

$$\begin{pmatrix} I & (JQ_k)^TD^{-1} \\ 0 & I \end{pmatrix}$$

yields the system,

$$\begin{pmatrix} Q_k^T P_G Q_k + (JQ_k)^TD^{-1}(JQ_k)^T \\ JQ_k \end{pmatrix} \begin{pmatrix} w_x \\ \bar{z}_\pi \end{pmatrix} = \begin{pmatrix} Q_k^T r_x \\ r_\pi \end{pmatrix}. $$
or equivalently,
\[ \begin{bmatrix} \tilde{w}_x \\ \tilde{z}_\pi \end{bmatrix} = S_k^T r. \]

As a result,
\[ \tilde{z} = \begin{bmatrix} Q_k w_x \\ \tilde{z}_\pi \end{bmatrix} = S_k \begin{bmatrix} w_x \\ \tilde{z}_\pi \end{bmatrix} = S_k (S_k^T K_\nu S_k)^{-1} S_k^T r, \]
as required.

\[ \begin{bmatrix} \tilde{w}_x \\ \tilde{z}_\pi \end{bmatrix} = S_k^T \bar{K}_\nu S_k \begin{bmatrix} \tilde{w}_x \\ \tilde{z}_\pi \end{bmatrix} \]

The expression for \( \tilde{z} \) implies that inner-iterations do indeed yield a viable direction.

**Theorem 4.5.5.** If \( r \neq 0 \), then any sequence of Lanczos-CG iterations for the normal equations will yield a vector \( \tilde{z} \) such that \( \tilde{z}^T r > 0 \).

**Proof.** From the previous theorem
\[ \tilde{z} = S_k (S_k^T \bar{K}_\nu S_k)^{-1} S_k^T r, \] (4.39)
and therefore
\[ \tilde{z}^T r = r^T S_k (S_k^T \bar{K}_\nu S_k)^{-1} S_k^T r \geq 0. \] (4.40)
As \( (S_k^T \bar{K}_\nu S_k)^{-1} \) is positive-definite, \( \tilde{z}^T r = 0 \) only if \( r \in \text{null}(S_k^T) \). If this were the case then
\[ \begin{pmatrix} Q_k^T & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} r_x \\ r_\pi \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \]
which implies that \( r_\pi = 0 \) and \( Q_k^T r_x = 0 \). Recall that \( Q_k \) is generated from Lanczos-CG on the condensed system, and therefore the first Lanczos vector is a scalar multiple of \( r_x - J^T D^{-1} r_\pi = r_x \). As a result, \( r \notin \text{null}(S_k^T) \) and hence, \( \tilde{z}^T r > 0 \).

The linear operator
\[ S_k (S_k^T \bar{K}_\nu S_k)^{-1} S_k^T \] (4.41)
from Theorem 4.5.4 turns out to be the Moore-Penrose generalized inverse of
\[ \bar{K}_\nu^{S_k} = \left( S_k S_k^T \right)^{-1} \bar{K}_\nu \left( S_k S_k^T \right). \]
As \( S_k \) has orthonormal columns and therefore this is the orthogonal projection of \( \bar{K}_\nu \) onto \( \text{span}(S_k) = \text{span}(Q_k) \oplus \mathbb{R}^m \).

**Theorem 4.5.6.** The Moore-Penrose generalized inverse of \( \bar{K}_\nu^{S_k} \) is
\[ \left( \bar{K}_\nu^{S_k} \right)^\dagger = S_k (S_k^T \bar{K}_\nu S_k)^{-1} S_k^T. \]
Proof. The uniqueness of the generalized inverse implies that it is sufficient to prove the following properties:

1. $\bar{K}^S_k (\bar{K}^S_k)^\dagger \bar{K}^S_k = \bar{K}^S_k$

2. $(\bar{K}^S_k)^\dagger \bar{K}^S_k (\bar{K}^S_k)^\dagger = (\bar{K}^S_k)^\dagger$

3. $(\bar{K}^S_k (\bar{K}^S_k)^\dagger)^T = \bar{K}^S_k (\bar{K}^S_k)^\dagger$

4. $(\bar{K}^S_k)^\dagger \bar{K}^S_k)^T = (\bar{K}^S_k)^\dagger \bar{K}^S_k$

Claim 1:

$\bar{K}^S_k (\bar{K}^S_k)^\dagger \bar{K}^S_k = (S_k S^T_k \bar{K}^S_k) (S^T_k S_k S^T_k) (S_k S^T_k \bar{K}^S_k)^{-1} S^T_k (S_k S^T_k \bar{K}^S_k) S_k S^T_k S_k$

$= S_k (S^T_k \bar{K}^S_k) (S^T_k \bar{K}^S_k) S_k S^T_k S_k$

$= \bar{K}^S_k$.

Claim 2:

$(\bar{K}^S_k)^\dagger \bar{K}^S_k (\bar{K}^S_k)^\dagger = (S_k S^T_k \bar{K}^S_k) (S^T_k S_k S^T_k) (S_k S^T_k \bar{K}^S_k)^{-1} S^T_k (S_k S^T_k \bar{K}^S_k) S_k S^T_k S_k$

$= S_k (S^T_k \bar{K}^S_k) (S^T_k S_k S^T_k) (S_k S^T_k \bar{K}^S_k) S_k S^T_k S_k$

$= (\bar{K}^S_k)^\dagger$.

Claim 3:

$(\bar{K}^S_k (\bar{K}^S_k)^\dagger)^T = ((S_k S^T_k \bar{K}^S_k) (S^T_k S_k S^T_k) (S_k S^T_k \bar{K}^S_k)^{-1} S^T_k) (S_k S^T_k \bar{K}^S_k)) T$

$= (S_k S^T_k) T$

$= S_k S^T_k$

$= \bar{K}^S_k (\bar{K}^S_k)^\dagger$.

Claim 4:

$((\bar{K}^S_k)^\dagger \bar{K}^S_k)^T = ((S_k (S^T_k \bar{K}^S_k)^{-1} S^T_k) (S_k S^T_k \bar{K}^S_k S^T_k)) T$

$= (S_k S^T_k) T$

$= S_k S^T_k$

$= (\bar{K}^S_k)^\dagger \bar{K}^S_k$. 

□
Chapter 5

Modified and incomplete factorizations

5.1 Overview

The previous chapter focused on the formulation of positive-definite constraint preconditioners for doubly-augmented systems. The main idea was to express $K_\nu$ as

$$
K_\nu = \begin{pmatrix}
H_D + \nu J^T D^{-1} J & \nu J^T \\
\nu J & \nu D
\end{pmatrix},
$$

(5.1)

and consider positive-definite preconditioners for the condensed matrix $H_D$. Previous results demonstrate that if positive-definite preconditioners for $H_D$ maintain a condensed matrix

$$
P_G = P + A^T G^{-1} A,
$$

(5.2)

then the bottleneck associated with solving

$$
\bar{K}_\nu = \begin{pmatrix}
P_G + \nu J^T D^{-1} J & \nu J^T \\
\nu J & \nu D
\end{pmatrix}
$$

(5.3)

occurs when solving the regularized KKT system

$$
P = \begin{pmatrix}
P & A^T \\
A & -G
\end{pmatrix}.
$$

(5.4)

This chapter will develop a means of constructing $P$ implicitly by applying incomplete factorization techniques to

$$
K = \begin{pmatrix}
H & J^T \\
J & -D
\end{pmatrix}.
$$

(5.5)
In the PDE-constrained context, these methods exploit sparsity.

5.2 Inertia-controlling factorizations

Inertia-controlling factorizations are intended to compute search directions in linearly-constrained and nonlinearly-constrained optimization. They are proposed by Forsgren and Murray for large-scale equality-constrained optimization [19], and are generalized by Gill and Forsgren [17], and Forsgren [16]. They are two-phase symmetric indefinite factorizations that relegate directions of negative curvature to the second phase. The first phase uses a special sequence of *inertia-controlling* pivots. This is in contrast to the traditional Bunch-Kaufman [7] and Bunch-Parlett [8] strategies that choose pivots solely on the basis of stability. The first phase ends when all rows from the $D$ block of $K$ have been eliminated. The pivots in the second phase can be chosen to satisfy any criteria.

A phase-one pivot is labeled by position, size, and inertia. The $1 \times 1$ pivots carry the label $H^+, H^0, H^-$ if they lie in the $H$-part of $K$. If a $1 \times 1$ pivot lies in the $D$-part of $K$ then it carries the label $D^+, D^0, D^-$. The sign of the superscript indicates the sign of the pivot. The superscript for $2 \times 2$ pivots indicates the number and type of eigenvalues. The first phase of an inertia-controlling factorization only allows $H^+, D^-, HH^{++}, DD^{--},$ and $HD^{++}$ pivots.

To show how inertia controlling methods work, consider $K$ partitioned in the form

$$
K = \begin{pmatrix}
H_{bb} & H_{bn} & J_b^T \\
H_{nb} & H_{nn} & J_n^T \\
J_b & J_n & -D
\end{pmatrix},
$$

where “$b$” and “$n$” imply a partitioning into “basic” and “nonbasic” variables. The first phase is complete when all $D$ rows of $K$ have been eliminated. Assume that the phase-one pivots come from the $H_{bb}, J_b,$ and $-D$ blocks of $K$. Let $\Pi_b$ be the permutation matrix defined by

$$
\Pi_b^T K \Pi_b = \begin{pmatrix} K_{bb} & K_{nb} \\
K_{bn} & H_{nn} \end{pmatrix},
$$

(5.6)

where

$$
K_{bb} = \begin{pmatrix} H_{bb} & J_b^T \\
J_b & -D \end{pmatrix} \quad \text{and} \quad K_{nn} = \begin{pmatrix} H_{nn} & J_n^T \end{pmatrix}.
$$

(5.7)

Using the restricted pivot choice, and ignoring the pivot order gives $K_{bb} = L_{11} B_{11} L_{11}^T$, with $\text{In}(K_{bb}) = (n_b, m, 0)$. This implies that $\text{In}(K) = (n, m, 0)$ if and only if the Schur
complement matrix, $H_{nn} - K_{bn} K_{bb}^{-1} K_{bn}^T$ is positive-definite.

The second phase computes an $L B L^T$ factorization of the Schur complement with the only restriction on pivot choice being that $\|L\|$ is sufficiently small. Ignoring the pivot order, the phase-two factorization ends with

$$H_{nn} - K_{nb} K_{bb}^{-1} K_{bn} = L_{22} B_{22} L_{22}^T,$$

yielding the inertia-controlling factorization,

$$\Pi_b^T K \Pi_b = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} B_{11} & 0 \\ 0 & B_{22} \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & L_{22}^T \end{pmatrix}.$$ (5.8)

### 5.2.1 A special ordering

A “tiling” is an ordering that enables a sparsity exploiting inertia-controlling factorization. The technique is proposed by Gill, Murray, Saunders, and Wright to solve KKT systems arising in sparse quadratic programming [29]. Their quadratic programming algorithm requires the use of direct methods to factorize KKT matrices with a zero $(2, 2)$ block. Symmetric indefinite solvers favor $1 \times 1$ pivots, and are not typically able to exploit the structure of KKT matrices.

A solution is to distribute the $D$-block of zeros over a set of preselected $2 \times 2$ pivots, known as *tiles* and let the solver find a favorable ordering with respect to the induced tile structure. A symbolic factorization is applied to the tiled matrix

$$\Pi_t^T K \Pi_t = \begin{pmatrix} T & F^T \\ F & E \end{pmatrix},$$ (5.10)

where $\Pi_t$ is a fixed permutation matrix and $T$ is a block matrix consisting entirely of $2 \times 2$ tiles, i.e.,

$$T = \begin{pmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{m1} & \cdots & t_{mm} \end{pmatrix}.$$  

with $t_{ij} = \begin{pmatrix} h & \tilde{j} \\ j & d \end{pmatrix}$.

The $h$, $j$, $\tilde{j}$ and $d$ represent elements from the $H$, $J$, and $D$ parts of $K$. Therefore each tile has a KKT-like structure, and $T$ contains all $D$ rows of $K$. This last point implies that if $\Pi_t$ is selected so that $\text{In}(T) = (m, m, 0)$, then tiling gives an inertia-controlling factorization. The following theorem describes a condition that $\Pi_t$ must satisfy.
Theorem 5.2.1. Let $J$ be an $m \times n$ matrix with full row-rank, $H$ be a $n \times n$ symmetric matrix, and $D = 0$. Suppose $K$ is defined by

$$K = \begin{pmatrix} H & J^T \\ J & -D \end{pmatrix}$$

satisfies $\text{In}(K) = (n, m, 0)$.

Finding a tiling with nonsingular $2m \times 2m$ matrix $T$ is equivalent to finding a linearly independent subset from the columns of $J$.

Proof. First, suppose first that a linearly independent subset of $m$ columns from $J$ has been identified. Define the matrix formed from these columns as $J_b$. The remaining $n - m$ columns of $J$ will be labeled $J_n$. There exists a permutation $\Pi_b$ such that

$$\Pi_b^T K \Pi_b = \begin{pmatrix} H_{bb} & J_b^T & H_{bn} \\ J_b & 0 & J_n \\ H_{nb} & J_n^T & H_{nn} \end{pmatrix}.$$ 

Since $J_b$ is nonsingular, the matrix $K_{bb}$ defined as

$$K_{bb} = \begin{pmatrix} H_{bb} & J_b^T \\ J_b & 0 \end{pmatrix}$$

has $\text{In}(K_{bb}) = (m, m, 0)$, and is therefore nonsingular. Let $\bar{\Pi}_t$ denote the permutation matrix that tiles $K_{bb}$, and extend it to define permutation on the entire matrix. Then

$$\Pi_t = \begin{pmatrix} \bar{\Pi}_t & 0 \\ 0 & I \end{pmatrix},$$

which implies $\Pi_t^T \Pi_b^T K \Pi_b \Pi_t = \begin{pmatrix} T & F^T \\ F & H_{nn} \end{pmatrix},$

where $F = \begin{pmatrix} H_{nb} & J_n^T \end{pmatrix} \bar{\Pi}_t$.

For the other direction, let $T$ be a nonsingular $2m \times 2m$ tiled matrix. This implies that there exists $\bar{\Pi}_t$ such that

$$\bar{\Pi}_t^T T \bar{\Pi}_t = \begin{pmatrix} H_{bb} & J_b^T \\ J_b & 0 \end{pmatrix},$$

where $H_{bb}$ and $J_b$ are submatrices of $H$ and $J$. Since $T$ is nonsingular, so is $J_b$, giving the result. 

This result implies that if nonsingular tilings exist, they yield inertia-controlling factorizations with $m$ $HD$ phase-one pivots. To date, there is no rule that tiles a general KKT matrix and yields an inertia-controlling factorization.
5.3 Symmetric quasi-definite matrices

Symmetric quasi-definite matrices will play a particularly prominent role in what follows.

**Definition 5.3.1.** A square symmetric matrix $\mathcal{P}$ is symmetric quasi-definite (SQD) if there exists a permutation matrix $\Pi$ such that

$$\Pi^T \mathcal{P} \Pi = \begin{pmatrix} P & A^T \\ A & -G \end{pmatrix},$$

where $P$ and $G$ are symmetric positive-definite.

SQD matrices can be thought of the symmetric indefinite analog of positive-definite matrices. This analogy comes from the following theorem of Vanderbei [65, 64], which describes the factorization properties of these matrices.

**Theorem 5.3.1 (Vanderbei).** Let $\mathcal{P}$ be an $n+m$ dimensional SQD matrix. For any permutation matrix $\Pi$, there exists a lower-triangular matrix $L$ and a diagonal matrix $B$ such that

$$\Pi^T \mathcal{P} \Pi = LBL^T.$$

**Stability of $LBL^T$ factorization for SQD matrices**

Strong-factorizability of SQD matrices is used in the nonlinear software package LOQO. Although fill-reducing ordering strategies are used, they do not take numerical stability into account. As a result, the $LBL^T$ factorization may lead to large $\|L\|$. The stability of $LBL^T$ factorizations for SQD matrices is explored by Gill, Saunders, and Shinner [36]. Their analysis exploits the equivalence of the $LBL^T$ factorization of SQD matrices and the $LDU$ factorization of a related unsymmetric positive-definite matrix.

Gill et al. derived conditions under which permutations for stability (but not necessarily sparsity) are necessary. This required developing a “certificate of stability” for a general SQD matrix that can be computed without factorizing the matrix. One of these certificates is

$$\theta(\mathcal{P}) = \left( \frac{\|A\|_2}{\max(\|P\|_2,\|G\|_2)} \right)^2 \max(\kappa_2(P),\kappa_2(G)).$$

The following result relates $\theta(\mathcal{P})$ and the need for pivoting.

**Theorem 5.3.2.** If $\mathcal{P}$ is SQD, the factorization $\Pi^T \mathcal{P} \Pi = LBL^T$ is stable for every permutation $\Pi$ if $\theta(\mathcal{P})$ is not too large.
In general, $\theta(P)$ is large if $\|A\|_2 \gg \max(\|P\|_2, \|G\|_2)$ or $\text{diag}(P, G)$ is very ill-conditioned. It is important to note that the previous theorem only applies to scalar factorizations. Because the KKT matrices in many problems, including those encountered in PDE-constrained optimization, are highly structured, it seems reasonable that alternative factorizations for SQD matrices may exist. If for instance, block $LBL^T$ factorizations are used, this stability analysis, described above, does not apply.

5.4 Tiling $K$

When the PDE is discretized with the finite element method, the KKT matrices arising in the optimization problem are large and sparse. Recall that the state and control variables naturally partition $K$ into a $3 \times 3$ block matrix

$$K = \begin{pmatrix} H & J^T \\ J & -D \end{pmatrix} = \begin{pmatrix} H_{yy} & H_{yu} & J_y^T \\ H_{uy} & H_{uu} & J_u^T \\ J_y & J_u & -D \end{pmatrix},$$

where $H_{yy}$ is $n_y \times n_y$, $J_y$ is $m \times n_y$, $J_u$ is $m \times n_u$, $H_{uu}$ is $n_u \times n_u$, $H_{yu}$ is $n_y \times n_u$, and $D$ is $m \times m$. Since the sparsity pattern of $H_{yy}$, $J_y$ are determined by the finite-element mesh, the $(m + n_y)$-dimensional matrix

$$K_{yy} = \begin{pmatrix} H_{yy} & J_y^T \\ J_y & -D \end{pmatrix}$$

(5.11)

can be tiled in a very well-defined way. Indeed, using the fact that $n_y = m$, the $2 \times 2$ tiling of $K_{yy}$ is

$$T_{yy} = \tilde{\Pi}_t^T K_{yy} \tilde{\Pi}_t = \begin{pmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{m1} & \cdots & t_{mm} \end{pmatrix}, \quad \text{with} \quad t_{ij} = \begin{pmatrix} [H_{yy}]_{ij} & [J_y]_{ij} \\ [J_y]_{ij} & -[D]_{ij} \end{pmatrix}. \quad (5.12)$$

As before, this tile ordering can be extended to the reordered version of $K$,

$$\begin{pmatrix} K_{yy} & K_{yu} \\ K_{uy} & H_{uu} \end{pmatrix}, \quad \text{where} \quad K_{uy} = \begin{pmatrix} H_{yy} & J_y^T \\ J_y & -D \end{pmatrix}$$

(5.13)

by considering

$$\begin{pmatrix} \tilde{\Pi}_t^T & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} K_{yy} & K_{yu} \\ K_{uy} & H_{uu} \end{pmatrix} \begin{pmatrix} \tilde{\Pi}_t & 0 \\ 0 & I \end{pmatrix} = \begin{pmatrix} T_{yy} & T_{yu} \\ T_{uy} & H_{uu} \end{pmatrix}$$

(5.14)
with $T_{yy} = K_{yy} \Pi_t$. From remarks above, the sparsity pattern of $T_{yy}$ with respect to the $2 \times 2$ tiles is the same as the matrix graph determined by the finite-element mesh. Depending on the problem being considered, the $(2, 2)$ block $H_{uu}$ and the columns of $K_{yu}$ may be dense or sparse. The possible cases are considered below.

### 5.4.1 Tiling with $3 \times 3$ matrices

In optimal control problems, the control variable is a function, and if a finite-element basis is used for the discretization of the control space, the matrices $H_{uu}$, $J_u$, and $H_{yu}$ are sparse. In PLTMG, the same finite-element space is used for both the state and control variables. Under this assumption, the sparsity patterns of $H_{uu}$, $J_u$, $H_{yu}$ are determined by the finite-element mesh topology, with possible contributions from boundary conditions. Therefore, the sparsity pattern of $H_{uu}$ with respect to its scalar entries is the same as the sparsity pattern of $T_{yy}$ with respect to its $2 \times 2$ tiles. These facts imply that the tiling of $K_{yy}$ may be extended to a tiling of $K$. To do this, the $2 \times 2$ tiles are expanded to include the appropriate entries from $H_{uu}$, $T_{yu}$, and $T_{uy}$. This expansion gives $3 \times 3$ tiles of the form

$$t_{ij} = \begin{pmatrix}
[H_y]_{ij} & [J_y]_{ji} & [H_{yu}]_{ij} \\
[J_y]_{ij} & -[D]_{ij} & [J_u]_{ij} \\
[H_{yu}]_{ji} & [J_u]_{ji} & [H_{uu}]_{ij}
\end{pmatrix}.$$  \hfill (5.15)

The reordered KKT matrix can be tiled as,

$$\Pi_t^T \begin{pmatrix}
T_{yy} & T_{yu} \\
T_{uy} & H_{uu}
\end{pmatrix} \Pi_t = \begin{pmatrix}
t_{11} & \cdots & t_{1m} \\
\vdots & \ddots & \vdots \\
t_{m1} & \cdots & t_{mm}
\end{pmatrix},$$  \hfill (5.16)

with $t_{ij}$ defined by (5.15). By reordering all tiles, a $3 \times 3$ tiling of $K$ can be derived. Reusing the notation already introduced, there is a permutation $\Pi_t$ such that

$$\Pi_t^T K \Pi_t = \begin{pmatrix}
t_{11} & \cdots & t_{1m} \\
\vdots & \ddots & \vdots \\
t_{m1} & \cdots & t_{mm}
\end{pmatrix},$$  \hfill (5.17)

where in this case,

$$t_{ij} = \begin{pmatrix}
[H_y]_{ij} & [H_{yu}]_{ij} & [J_y]_{ji} \\
[H_{yu}]_{ji} & [H_{uu}]_{ij} & [J_u]_{ji} \\
[J_y]_{ij} & [J_u]_{ij} & -[D]_{ij}
\end{pmatrix}.$$  \hfill (5.18)
5.4.2 Generalized tiling for $n_u < n_y$

The requirement that $n_y = n_u$ imposes structural limitations on the problem. In general, there will be far fewer control variables than state variables. For example, in parameter estimation problems, $n_u$ may be as small as one. In fact, PLTMG includes a parameter estimation option that uses one control variable. It may also be the case that the control variable models some subdomain or portion of the boundary. In these cases, the $3 \times 3$ tilings as described above cannot be used. By artificially enlarging the number of control variables from $n_u$ to $n_y$, $3 \times 3$ tilings can be used.

Consider the enlarged control vector $x^e_u \in \mathbb{R}^{n_y}$, defined by

$$x^e_u = (x_u, x^a_u), \quad \text{where } x^a_u \in \mathbb{R}^{n_y - n_u} \text{ and } x^a_u = 0,$$

with the enlarged search direction being

$$s^e_u = (s_u, s^a_u), \quad \text{where } s^a_u \in \mathbb{R}^{n_y - n_u}.$$

The key property is that $(s_y, s_{\pi}, s_u)$ satisfies the system

$$
\begin{pmatrix}
H_{yy} & J_y^T & H_{yu} \\
J_y & -D & J_u \\
H_{uy} & J_u^T & H_{uu}
\end{pmatrix}
\begin{pmatrix}
s_y \\
s_{\pi} \\
s_u
\end{pmatrix}
= 
\begin{pmatrix}
b_y \\
b_{\pi} \\
b_u
\end{pmatrix},
$$

if and only if $(s_y, s_{\pi}, s^e_u)$ satisfies the expanded system

$$
\begin{pmatrix}
H_{yy} & J_y^T & H_{yu}^e \\
J_y & -D & J_u^e \\
H_{uy}^e & (J_u^e)^T & H_{uu}^e
\end{pmatrix}
\begin{pmatrix}
s_y \\
s_{\pi} \\
s^e_u
\end{pmatrix}
= 
\begin{pmatrix}
b_y \\
b_{\pi} \\
b^e_u
\end{pmatrix},
$$

where $b^e_u = (b_u, 0) \in \mathbb{R}^{n_y}$ is expanded by a vector of zeros as before. Also, $H_{yu}$ and $J_u^e$ are both $n_y \times n_y$ matrices of the form

$$H_{yu}^e = \begin{pmatrix} H_{yu} & 0 \end{pmatrix} \quad \text{and} \quad J_u^e = \begin{pmatrix} J_u & 0 \end{pmatrix},$$

where the “0” here denotes $n_y \times (n_y - n_u)$ matrix of all zeros. Therefore,

$$H_{yu}^e s^e_u = H_{yu} s_u \quad \text{and} \quad J_u^e s^e_u = J_u s_u.$$

Finally $H_{uu}^e$ will be of the form

$$H_{uu}^e = \begin{pmatrix} H_{uu} & 0 \\ 0 & \alpha I \end{pmatrix} \quad \text{where } \alpha > 0.$$
and $I$ is the $(n_y - n_u)$ identity matrix. The enlarged system in block $4 \times 4$ form is therefore,

$$
\begin{pmatrix}
H_{yy} & J_y^T & H_{yu} & 0 \\
J_y & -D & J_u & 0 \\
H_{uy} & J_u^T & H_{uu} & 0 \\
0 & 0 & 0 & \alpha I
\end{pmatrix}
\begin{pmatrix}
s_y \\
s_x \\
s_u \\
0
\end{pmatrix}
=
\begin{pmatrix}
b_y \\
b_x \\
b_u \\
0
\end{pmatrix}.
$$

(5.21)

Clearly, this system is nonsingular and $s_u^0 = 0$. The following result shows that the number of positive eigenvalues of the expanded KKT matrix

$$
K^e = \begin{pmatrix}
H^e & (J^e)^T \\
J_e & -D
\end{pmatrix}
$$

(5.22)

increases by $(n_y - n_u)$, where

$$
H^e = \begin{pmatrix}
H_{yy} & H_{yu} & 0 \\
H_{uy} & H_{uu} & 0 \\
0 & 0 & \alpha I
\end{pmatrix}, \text{ and } J^e = \begin{pmatrix}
J_y & J_u & 0
\end{pmatrix}.
$$

(5.23)

**Theorem 5.4.1.** If $K^e$ is the enlarged KKT matrix in (5.22), then

$$
\text{In}(K^e) = \text{In}(K) + (n_y - n_u, 0, 0).
$$

**Proof.** Sylvester’s law of inertia implies that

$$
\text{In}(K^e) = \text{In}(H^e + (J^e)^TD^{-1}J^e) + (0, m, 0).
$$

By construction,

$$
H^e + (J^e)^TD^{-1}J^e = \begin{pmatrix}
H + J^TD^{-1}J & 0 \\
0 & \alpha I
\end{pmatrix},
$$

and since $\alpha > 0$,

$$
\text{In}(H^e + (J^e)^TD^{-1}J^e) = \text{In}(H + J^TD^{-1}J) + (n_y - n_u, 0, 0).
$$

Putting everything together gives,

$$
\text{In}(K^e) = \text{In}(H^e + (J^e)^TD^{-1}J^e) + (0, m, 0) \\
= \text{In}(H + J^TD^{-1}J) + (n_y - n_u, 0, 0) + (0, m, 0) \\
= \text{In}(K) + (n_y - n_u, 0, 0).
$$
5.5 Block factorizations for \( 3 \times 3 \) tilings

The focus of this section is the block factorization of regularized KKT matrices with \( 3 \times 3 \) blocks. If \( K \) comes from a problem for which \( n_u < n_y \), \( K \) represents the enlarged KKT matrix \( K^e \).

Unless a dynamic pivoting strategy is used, it is not generally the case that an \( LBL^T \) factorization for a symmetric matrix exists. In the block setting, pivot breakdown occurs when a diagonal block pivot is singular. In PDE-constrained optimization, an \( LBL^T \) factorization may exist, independent of numerical pivoting.

When block factorizations do not succeed, it is possible to modify the pivots in a numerically stable way. The following theorem provides motivation for the strategy and is based on a result due to Forsgren [16] in the context of inertia-controlling factorizations. It can be used to show that the \( 3 \times 3 \) tile factorization will fail if one of the diagonal tile pivots has a singular \( H \)-block.

**Theorem 5.5.1.** Let \( K \) be the \((n + m)\) dimensional regularized KKT matrix partitioned into the form

\[
K = \begin{pmatrix}
H_{11} & H_{12} & J_{11}^T & J_{12}^T \\
J_{11} & J_{12} & -D_{11} & -D_{12} \\
J_{21} & J_{22} & -D_{21} & -D_{22}
\end{pmatrix},
\]

and assume that \( H_{11} \) is an \( n_1 \times n_1 \) positive-definite matrix. Denote the dimensions of \( H_{22} \), \( D_{11} \), and \( D_{22} \) by \( n_2 \), \( m_1 \), and \( m_2 \) respectively. Then the \( D \)-part of the Schur-complement \( \hat{K}/K_{11} \), where

\[
\hat{K} = \begin{pmatrix}
K_{11} & K_{12} \\
K_{21} & K_{22}
\end{pmatrix}
\]

and \( K_{11} = \begin{pmatrix} H_{11} & J_{11} \\ J_{11}^T & -D_{11} \end{pmatrix} \)

is negative-definite. Furthermore, if \( K \) is SQD, then so is the Schur complement.

**Proof.** Let the Schur complement in question be denoted by

\[
S = \begin{pmatrix}
S_{11} & S_{12} \\
S_{21} & S_{22}
\end{pmatrix}, \quad (5.24)
\]

where \( S_{11} \) comes from the \( H \)-block of \( K \), and \( S_{22} \) comes from the \( D \) block of \( K \). Observe that \( S_{22} \) is also equal to the Schur-complement \( K_{11} \) in \( \hat{K}_D \), where

\[
\hat{K}_D = \begin{pmatrix}
H_{11} & J_{11}^T & J_{12}^T \\
J_{11} & -D_{11} & -D_{12} \\
J_{21} & -D_{21} & -D_{22}
\end{pmatrix}.
\]
This matrix has the same inertia as the SQD matrix
\[
\begin{pmatrix}
H_{11} & J_{11}^T & J_{12}^T \\
J_{11} & -D_{11} & -D_{12} \\
J_{12} & -D_{21} & -D_{22}
\end{pmatrix}.
\]
Therefore, the inertia of the $D$-block of $S$ is
\[
\text{In}(S_{22}) = \text{In}(\tilde{K}_D) - \text{In}(K_{11})
= (n_1, m, 0) - (n_1, m_1, 0)
= (0, m_2, 0).
\]
Hence, the $D$-block of $S$ is negative-definite.

To show that $S$ is SQD when $K$ is SQD, it suffices to show that the $S_{11}$ is positive definite. As in the previous case, $S_{11}$ is equal to the Schur complement of $K_{11}$ in $\tilde{K}_H$, where
\[
\tilde{K}_H = \begin{pmatrix}
H_{11} & J_{11}^T & H_{12} \\
J_{11} & -D_{11} & J_{12} \\
H_{21} & J_{12}^T & H_{22}
\end{pmatrix}.
\]
This matrix has the same inertia as the SQD matrix
\[
\begin{pmatrix}
H_{11} & H_{12} & J_{11}^T \\
H_{21} & H_{22} & J_{12}^T \\
J_{11} & J_{12} & -D_{11}
\end{pmatrix},
\]
which implies that
\[
\text{In}(S_{11}) = \text{In}(\tilde{K}_H) - \text{In}(K_{11})
= (n, m_1, 0) - (n_1, m_1, 0)
= (n_2, 0, 0).
\]
Hence, $S_{11}$ is positive-definite, as required.

In the theorem above, $K_{11}$ is an SQD block-pivot. If $K$ is SQD, then so is $S$ and an SQD pivot can be selected from it as well. Continuing in this way yields the inertia controlling factorization,
\[
\Pi_t^T K \Pi_t = \begin{pmatrix}
L_{11} & 0 \\
L_{21} & L_{22}
\end{pmatrix} \begin{pmatrix}
B_{11} & 0 \\
0 & B_{22}
\end{pmatrix} \begin{pmatrix}
L_{11}^T & L_{12}^T \\
L_{21}^T & L_{22}^T
\end{pmatrix},
\]
where $B_{11} = \text{diag}(B_1, \ldots, B_m)$ consists entirely of $HD^+$ pivots and the block diagonal matrix $B_{22} = \text{diag}(B_{m+1}, \ldots, B_s)$ consists of $1 \times 1$ or $2 \times 2$ pivots determined by a standard symmetric indefinite factorization.
### 5.5.1 Constructing SQD preconditioners

This section considers methods for approximating $K$ with an SQD matrix

$$\tilde{K} = \begin{pmatrix} \tilde{H} & J^T \\ J & -D \end{pmatrix},$$

(5.25)

where $\tilde{H} \approx H$ is positive-definite. The selection of $\tilde{H}$ can be performed either *statically* by exploiting the structure in $H$, or *dynamically* by applying a modified factorization to $K$.

#### Static techniques

A static technique for guaranteeing a positive-definite preconditioner $\tilde{K}_\nu$ finds an SQD preconditioner for $K$. As $D$ is positive-definite by assumption, replacing $H$ with a positive-definite matrix $\tilde{H}$ will yield an SQD preconditioner,

$$\tilde{K} = \begin{pmatrix} \tilde{H} & J^T \\ J & -D \end{pmatrix}.$$  

(5.26)

Because the goal is to apply a factorization procedure to $\tilde{K}$, it is crucial that $\tilde{H}$ be sparse. Depending on the problem, choosing a good approximation to $H$ may or may not be easy. The most straightforward method drops $H_{yu}$ from $H$ and considers,

$$\tilde{H} = \begin{pmatrix} \tilde{H}_{yy} & 0 \\ 0 & \tilde{H}_{uu} \end{pmatrix}$$

(5.27)

where $\tilde{H}_{yy} \approx H_{yy}$ and $\tilde{H}_{uu} \approx H_{uu}$ are positive-definite. For example, $\tilde{H}_{yy}$ and $\tilde{H}_{uu}$ may be defined as

$$\tilde{H}_{yy} = \text{diag}(H_{yy}) + R_{yy} \quad \text{and} \quad \tilde{H}_{uu} = \text{diag}(H_{uu}) + R_{uu},$$

(5.28)

where $R_{yy}$ and $R_{uu}$ are positive-definite diagonal matrices of size $n_y$ and $n_u$. It is worth pointing out that approximating $H$ in this way ignores any nonlinear coupling between $x_y$ and $x_u$ in $c(x)$, and hence may not be a very good approximation to $H$.

### 5.6 Incomplete factorizations

Direct factorizations of large, sparse matrices arising from finite-element discretizations are not recommended. For fine spatial discretizations, the factorizations discussed so far are impractical. The purpose of this section is to apply the partial elimination techniques for symmetric positive-definite matrices proposed by Tuff and Jennings [63], and Jennings and Malik [43] to the SQD case.
5.6.1 Partial elimination for positive-definite matrices

Partial elimination strategies simulate a full factorization of a positive-definite matrix, while dropping elements that are assumed to have little effect on the final factors. Therefore, the method can reduce storage costs. An important component is a diagonal-compensation strategy that incorporates the dropped elements into the diagonal of the matrix. This ensures that the matrix remains positive-definite, while not affecting the factors “too much”, when elements are dropped.

To give the method some context, let $P$ denote an $n \times n$ symmetric positive-definite matrix partitioned in the form

$$P = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}, \quad \text{with } P_{11} \in \mathbb{R}^{k \times k},$$

and consider the partial Cholesky factorization,

$$P = \begin{pmatrix} L_{11} & 0 \\ L_{21} & I \end{pmatrix} \begin{pmatrix} B_{11} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & I \end{pmatrix}, \quad (5.29)$$

where $S = P_{22} - L_{21}B_{11}L_{21}^T$ is the Schur complement, and $B_{11}$ consists of $k$, $1 \times 1$ positive pivots. Step $k + 1$ of the procedure starts by computing the first column of $S$ as

$$Se_1 = P_{22}e_1 - L_{21}B_{11}L_{21}^Te_1 \in \mathbb{R}^{n-k}, \quad (5.30)$$

and ends with a matrix $\bar{S} \approx S$ such that $\bar{S}$ is symmetric positive definite and $\bar{S}e_1$ has fewer nonzeros than $Se_1$. The first column of $\bar{S}$ is determined by the following threshold condition

$$\bar{s}_{j1} = \begin{cases} s_{j1} & \text{if } |s_{j1}| \geq \tau \sqrt{s_{11}s_{jj}} \\ 0 & \text{otherwise.} \end{cases} \quad (5.31)$$

Because $S$ is symmetric, the same rule applies to the first row of $S$ and $\bar{S}$. Therefore, only sufficiently large elements are retained in $\bar{S}e_1$. If the remaining rows and columns of $S$ are kept for $\bar{S}$, then $\bar{S}$ may be indefinite. The diagonal modification strategy that keeps $\bar{S}$ positive-definite depends on a family of strictly positive scalar parameters $\{\xi_{ij}\}$ and can be summarized with the following pseudo-code:
Algorithm 5.1 Dropping and diagonal modification strategy

Let \( \bar{S} = S \);

do \ j = 1 : n - k 

\[
\text{if } \frac{|s_{j1}|}{\sqrt{s_{11} s_{jj}}} < \tau \text{ then } \\
\quad \text{Choose } \xi_{j1} > 0; \\
\quad \bar{s}_{j1} = 0; \quad \bar{s}_{ij} = 0; \\
\quad \bar{s}_{11} = s_{11} + \xi_{j1}|s_{1j}|; \quad \bar{s}_{jj} = s_{jj} + \frac{|s_{j1}|}{\xi_{j1}}; \\
\text{end if }
\]

end do

The following lemma justifies why Algorithm 5.6.1 defines a positive-definite \( \bar{S} \).

Lemma 5.6.1. Assume \( k < l \) and let \( \xi > 0 \). If \( S \) be an \( n \times n \) symmetric positive-definite matrix and \( \bar{S} \) is defined according to the rules

\[
\bar{s}_{ij} = \begin{cases} 
0 & \text{if } (i, j) = (k, l) \\
 s_{ii} + \xi |s_{kl}| & \text{if } i = k \\
 s_{ii} + |s_{lk}|/\xi & \text{if } i = l \\
 s_{ij} & \text{otherwise}
\end{cases} \quad (5.32)
\]

then \( \bar{S} \) is symmetric positive-definite.

Proof. Let \( \mathcal{E}_{kl} \) be the \( n \times n \) matrix of zeros with the \( k \)-th and \( l \)-th diagonals replaced with \( \xi |s_{kl}| \) and \( |s_{lk}|/\xi \), and the \( (k, l) \)-th and \( (l, k) \)-th entries replaced with \( -s_{lk} \) and \( -s_{kl} \). With this notation, \( \bar{S} = S + \mathcal{E}_{kl} \). Consider the \( 2 \times 2 \) positive-semidefinite matrix

\[
\mathcal{E}_{kl}^e = \begin{pmatrix} 
\xi |s_{kl}| & -s_{kl} \\
-s_{kl} & |s_{kl}|/\xi 
\end{pmatrix}. \quad (5.33)
\]

As \( \mathcal{E}_{kl} \) is a symmetric permutation of the \( k \times k \) symmetric positive-semidefinite matrix

\[
\begin{pmatrix} 
\mathcal{E}_{kl}^e & 0 \\
0 & 0 
\end{pmatrix}, \quad (5.34)
\]

the matrix \( \mathcal{E}_{kl} \) must also be semidefinite, thereby implying the positive-definiteness of \( \bar{S} \).

The previous theorem is somewhat general in the sense that \( \xi \) is left unspecified. As \( \bar{P} \) should be as close to \( P \) as possible, it is important not to modify \( P \) too much. It turns out that this is the case if (5.31) is used as the drop test.
Lemma 5.6.2. Assume $k < l$ and let $S$ be an $n \times n$ symmetric positive-definite matrix. Suppose $s_{kl}$ and $s_{lk}$ are dropped according to (5.31). Moreover, assume that the $k$-th and $l$-th diagonals of $S$ are modified as in Algorithm 5.6.1. If $\xi$ is of the form
\[
\xi = \alpha \sqrt{\frac{s_{kk}}{s_{ll}}} \quad \text{with} \quad \alpha > 0,
\]
then,
\[
\frac{|\bar{s}_{kk} - s_{kk}|}{s_{kk}} \leq \alpha \tau \quad \text{and} \quad \frac{|\bar{s}_{ll} - s_{ll}|}{s_{ll}} \leq \alpha \tau.
\]

Proof. The update formulae are:
\[
\bar{s}_{kk} = s_{kk} + \alpha |s_{kl}| \sqrt{s_{kk}/s_{ll}} \quad \text{and} \quad \bar{s}_{ll} = s_{ll} + \alpha |s_{kl}| \sqrt{s_{ll}/s_{kk}}
\]
Letting $\Delta s_{kk} = \bar{s}_{kk} - s_{kk}$ and $\Delta s_{ll} = \bar{s}_{ll} - s_{ll}$, and using the fact that $|s_{kl}| < \tau \sqrt{s_{kk}s_{ll}}$ implies that
\[
|\Delta s_{kk}| = \alpha |s_{kl}| \sqrt{s_{kk}/s_{ll}} \leq \alpha \tau \sqrt{s_{kk}s_{ll}} \sqrt{s_{kk}/s_{ll}} = \alpha \tau s_{kk},
\]
giving the first bound. The second bound follows in a similar way,
\[
|\Delta s_{ll}| = \alpha |s_{kl}| \sqrt{s_{ll}/s_{kk}} \leq \alpha \tau \sqrt{s_{kk}s_{ll}} \sqrt{s_{ll}/s_{kk}} = \alpha \tau s_{ll},
\]
as required. \qed

The end result of the partial elimination strategy is the computation of the Cholesky factorization of a perturbed, positive-definite matrix of the form
\[
P + \mathcal{E} = LBL^T,
\]
where $\mathcal{E}$ is a sum of the compensation matrices $\mathcal{E}_{ij}$ considered in Lemma 5.6.1.

5.7 Incomplete factorizations for KKT matrices

In this section, an analogue of partial elimination for block matrices will be developed. Let $T$ denote a tiled matrix derived from a $2 \times 2$ or $3 \times 3$ tiling of $K$, i.e.,
\[
\Pi^T_t K \Pi_t = T.
\]
In order to treat $3 \times 3$ and $2 \times 2$ tilings simultaneously, label the $(i,j)$-th tile $t_{ij}$ in the following way
\[
t_{ij} = \begin{pmatrix} [t_{ij}]_H & [t_{ij}]_J^T \\ [t_{ij}]_J & [t_{ij}]_D \end{pmatrix},
\]
where the subscripts $H$, $D$, $J$ and $J^T$ denote the element labels in a given tile.

### 5.7.1 Tile factorizations

The partial elimination strategy in the tile setting involves dropping tiles, not just scalar entries. As in the positive-definite setting, suppose that $k$-steps of a partial block factorization of $T$ have been computed, giving
\[
T = \begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & I \end{pmatrix} \begin{pmatrix} B_{11} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & I \end{pmatrix},
\]
where, in this case, $B_{11}$ consists entirely of $k$, $2 \times 2$ tile pivots or $k$, $3 \times 3$ tile pivots. Unlike the scalar case, $\tilde{S}$, the modified Schur complement $S$, must be kept SQD. It should also be the case that the dropping and diagonal compensation strategy prevent $S$ from being modified too much. A natural extension of (5.31) to the tile setting is given by
\[
\tilde{s}_{j1} = \begin{cases} s_{j1} & \text{if } \|s_{j1}\|_\infty \geq \tau \sqrt{\|s_{11}\|_\infty^{-1} \|s_{jj}\|_\infty} \\ 0 & \text{otherwise.} \end{cases}
\]

Because dropping tiles may prevent $\tilde{S}$ from being SQD, a tile diagonal compensation strategy must be proposed. The most obvious way to do this is to incorporate the $H$-blocks and $D$-blocks of dropped tiles into the $H$ and $D$ blocks of specific diagonal tiles with compensation matrices. In the tile context, the elimination matrices will be quasi-semidefinite rather than semidefinite. The $J$ and $J^T$ blocks of dropped tiles can be dropped completely without any compensation.

Consider the $2 \times 2$ or $3 \times 3$ matrix
\[
\Phi = \phi \begin{pmatrix} E & 0 \\ 0 & -1 \end{pmatrix}, \quad \text{where } \phi = \sqrt{\|s_{11}\|_\infty / \|s_{jj}\|_\infty}.
\]
If $T$ is a $2 \times 2$ tiling then $E = 1$. If $T$ is a $3 \times 3$ tiling, $E$ is the $2 \times 2$ matrix of all ones. Also, consider the matrix
\[
\Theta = \phi^{-1} \begin{pmatrix} E & 0 \\ 0 & -1 \end{pmatrix}.
\]
Because $E$ is positive semidefinite, both $\Phi$ and $\Theta$ are quasi-semidefinite. In the $3 \times 3$ tile case

$$\text{diag}(|s|\Phi) = \phi \begin{pmatrix} |s_{11}| + |s_{12}| \\ |s_{21}| + |s_{21}| \\ -|s_{33}| \end{pmatrix},$$

and in the $2 \times 2$ tile case

$$\text{diag}(|s|\Phi) = \phi \begin{pmatrix} |s_{11}| \\ -|s_{22}| \end{pmatrix},$$

which implies that $\text{diag}(\text{diag}(|s|\Phi))$ is SQD and $\|\text{diag}(|s|\Phi)\|_\infty \leq \phi\|s\|_\infty$. The diagonal compensation strategy can now be described as follows.

**Lemma 5.7.1.** Assume that $k < l$. Define $\phi > 0$. If $S$ is a block $n \times n$ matrix of tiles that is also SQD, define $\tilde{S}$ according to the rule

$$\tilde{s}_{ij} = \begin{cases} 0 & \text{if } (i, j) = (k, l) \\ s_{ii} + \text{diag}(\text{diag}(|s_{kl}|\Phi)) & \text{if } i = k \\ s_{ii} + \text{diag}(\text{diag}(|s_{lk}|\Theta)) & \text{if } i = l \\ s_{ij} & \text{otherwise}. \end{cases}$$

Then the matrix $\tilde{S}$ is SQD.

**Proof.** Let $\mathcal{E}_{kl}$ be the block $n \times n$ matrix of zeros with $k$-th and $l$-th diagonal blocks replaced by $\text{diag}(\text{diag}(|s_{kl}|\Phi))$ and $\text{diag}(\text{diag}(|s_{kl}|\Theta))$, and $(k, l)$-th and $(l, k)$-th blocks replaced by $-s_{kl}$ and $-s_{lk}$. With this notation, $\tilde{S} = S + \mathcal{E}_{kl}$. Consider the quasi-semidefinite matrix,

$$\mathcal{E}_{kl}^c = \begin{pmatrix} \text{diag}(\text{diag}(|s_{kl}|\Phi)) & -s_{kl} \\ -s_{lk} & \text{diag}(\text{diag}(|s_{lk}|\Theta)) \end{pmatrix}.\quad (5.47)$$

Because $\mathcal{E}_{kl}$ is a symmetric permutation of the block $n \times n$ symmetric quasi-semidefinite matrix

$$\begin{pmatrix} \mathcal{E}_{kl}^c & 0 \\ 0 & 0 \end{pmatrix},\quad (5.48)$$

the matrix $\mathcal{E}_{kl}$ is also quasi-semidefinite, thereby implying that $\tilde{S}$ is SQD. □

The next theorem shows that this diagonal compensation strategy does not perturb diagonal tiles “too much.”
Theorem 5.7.1. Assume \( k < l \), and let \( S \) be a block \( n \times n \) matrix of \( 2 \times 2 \) or \( 3 \times 3 \) tiles that is also SQD. Suppose \( s_{kl} \) and \( s_{lk} \) are dropped, and that the diagonal compensation strategy of Lemma 5.7.1 is applied. Then

\[
\frac{\|\bar{s}_{kk} - s_{kk}\|_\infty}{\|s_{kk}\|_\infty} \leq \tau \quad \text{and} \quad \frac{\|\bar{s}_{ll} - s_{ll}\|_\infty}{\|s_{ll}\|_\infty} \leq 3\tau.
\] (5.49)

Proof. According to the update formulae

\[
\bar{s}_{kk} = s_{kk} + \text{diag}(\text{diag}(|s_{kl}|\Phi)) \quad \text{and} \quad \bar{s}_{ll} = s_{ll} + \text{diag}(\text{diag}(|s_{lk}|\Theta)).
\] (5.50)

If \( \Delta s_{kk} = \bar{s}_{kk} - s_{kk} \) and \( \Delta s_{ll} = \bar{s}_{ll} - s_{ll} \), it follows from previous remarks that

\[
\|\Delta s_{kk}\|_\infty \leq \phi\|s_{kl}\|_\infty \quad \text{and} \quad \|\Delta s_{ll}\|_\infty \leq \phi^{-1}\|s_{lk}\|_\infty.
\] (5.51)

Since \( s_{kl} \) is dropped from the factorization, it holds that

\[
\|s_{kl}\|_\infty < \tau \sqrt{\|s_{ll}\|_\infty / \|s_{kk}\|_\infty} \leq \tau \sqrt{\|s_{ll}\|_\infty \|s_{kk}\|_\infty},
\]

and therefore

\[
\|\Delta s_{kk}\|_\infty \leq \phi\|s_{kl}\|_\infty \leq \tau \sqrt{\|s_{ll}\|_\infty / \|s_{kk}\|_\infty} \sqrt{\|s_{ll}\|_\infty \|s_{kk}\|_\infty} \leq \tau\|s_{kk}\|_\infty,
\]

giving the first result. To see the second result, we use the fact that

\[
\|s_{lk}\|_\infty = \|s_{kl}\|_1 \leq 3\|s_{kl}\|_\infty,
\]

to yield

\[
\|\Delta s_{ll}\|_\infty \leq 3\phi^{-1}\|s_{kl}\|_\infty \leq 3\tau \sqrt{\|s_{ll}\|_\infty / \|s_{kk}\|_\infty} \sqrt{\|s_{ll}\|_\infty \|s_{kk}\|_\infty} \leq 3\tau\|s_{ll}\|_\infty,
\]

as required. \( \square \)
Chapter 6

Numerical results

6.1 Overview

In this chapter, we discuss implementation details and preliminary numerical results. An interior-point method that includes the incomplete block preconditioner, has been incorporated into the finite-element package PLTMG [2]. The problems considered take the form:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad c(x) = 0, \quad b_\ell \leq x_u \leq b_u.
\end{align*}
\]

(6.1)

As in Chapter 1, \( x_y \) is the vector of discrete state variables and \( x_u \) is the vector of control variables. The objective function \( f \) is always quadratic in the discrete states and controls, i.e.,

\[
f(x) = \frac{1}{2}(x_y - x_t)^T M_y(x_y - x_t) + \frac{\eta}{2} x_u^T M_u x_u,
\]

(6.2)

which is the discretized version of the tracking-type functional

\[
f(y, u) = \frac{1}{2} \| y - y_t \|_{L^2(\Omega)}^2 + \frac{\eta}{2} \| u \|_{L^2(\Omega)}^2,
\]

(6.3)

where \( \Omega \subset \mathbb{R}^2 \) is the domain of interest.

6.2 Model problems

The problems are characterized by the expression for \( x_t \) and the corresponding state equation. In the first set of examples it is assumed that \( x_t \) is the finite element discretization of the function \( y_t(\zeta) = \sin(2\pi\zeta_1)\sin(2\pi\zeta_2) \), \( \eta = 10^{-4} \), with

\[
\begin{align*}
b_\ell &= -100e \in \mathbb{R}^m \quad \text{and} \quad b_u = 100e \in \mathbb{R}^m.
\end{align*}
\]
The domain of interest, \( \Omega \), is the unit circle in \( \mathbb{R}^2 \).

Problem 1:
\[
\Delta y = -u \quad \text{in} \quad \Omega, \quad y = 0 \quad \text{on} \quad \partial \Omega.
\]

Problem 2:
\[
\Delta y = -u \quad \text{in} \quad \Omega, \quad y = y_t \quad \text{on} \quad \Gamma_d, \quad \partial_n y = \partial_n y_t \quad \text{on} \quad \Gamma_n.
\]

Problem 3:
\[
\nabla \cdot (e^y \nabla y) = -u \quad \text{in} \quad \Omega, \quad y = 0 \quad \text{on} \quad \partial \Omega.
\]

For Problem 2 above, \( \Gamma_d \) is the curve on the unit circle between 0 and \( \pi/4 \) radians, and \( \Gamma_n \) is the remainder of \( \partial \Omega \). The constraints for Problems 1 and 2 are linear, whereas Problem 3 has nonlinear constraints.

### 6.3 The preconditioner

The incomplete block preconditioner discussed in Chapter 5 implicitly defines a positive-definite preconditioner for the Steihaug-Toint CG method. For a given \( \mu, \rho \), and \( v = (x, \pi) \), the incomplete factorization of

\[
\tilde{K}_\mu(x, \pi; \rho) = \begin{pmatrix}
\tilde{H}_\mu(x, \pi) & J(x)^T \\
J(x) & -D_\rho
\end{pmatrix},
\]

is executed, where \( \tilde{H}_\mu(x, \pi) \) is a positive-definite approximation of \( H_\mu(x, \pi) \). This can be done in a number of ways and the current method replaces the matrix

\[
H_\mu(x, \pi) = \begin{pmatrix}
H_{yy} & H_{yu} \\
H_{uy} & H_{uu}
\end{pmatrix}
\]

by \( \tilde{H}_\mu(x, \pi) = \begin{pmatrix}
\tilde{H}_{yy} & 0 \\
0 & \tilde{H}_{uu}
\end{pmatrix} \),

where \( \tilde{H}_{yy} \) and \( \tilde{H}_{uu} \) are positive-definite diagonal matrices defined component-wise by

\[
[\tilde{H}_{yy}]_{ii} = \max \{ \tau, [H_{yy}]_{ii} \} \quad \text{and} \quad [\tilde{H}_{uu}]_{ii} = \max \{ \tau, [H_{uu}]_{ii} \}, \quad \text{(6.4)}
\]

with \( \tau (\tau > 0) \) a user-defined threshold parameter. A large value for \( \tau \) leads to a more stable factorization with fewer nonzeros. However, it may also produce an inaccurate preconditioner. More analysis and numerical experiments are needed to determine an estimate of the optimal value for \( \tau \).
6.4 Testing the preconditioner

To access the quality of the preconditioner, the preconditioned conjugate-gradient algorithm is used to find an approximate solution of the system $K_\nu x = b$. The vector $b$ is obtained by multiplying a doubly-augmented system by the vector of ones. This allows the error between the exact solution and computed solution to be evaluated. The matrix $K_\nu$ is a snapshot of a doubly-augmented system associated with Problems 1, 2, and 3 above. The following tests use a barrier parameter of $\mu = 10^{-3}$, a penalty parameter $\rho = 10^3$. The unit circle has been discretized with piecewise linear finite elements, resulting in 1089 grid points. The doubly-augmented systems are not formed explicitly, but are defined implicitly by making calls to the finite-element package PLTMG with a vector of state variables, control variables, Lagrange multipliers, and multiplier estimates. The state variables, Lagrange multipliers, and multiplier estimates are all randomly generated while the control variable is fixed to lie halfway between $b_{\ell}$ and $b_u$.

The quantity “nnz($L$)/nnz” in the tables below is the ratio of nonzero tiles in the lower triangular portion of the incomplete factors verse the nonzero tiles in the original regularized KKT matrix and “rel. resid.” is the quantity $\|K_\nu s - b\|_h/\|b\|_h$, where $s$ is the PCG output and $\|\cdot\|_h$ is a weighted two norm. The first line in each table is the result of applying the standard CG method without a preconditioner. As Problem 3 involves a nonlinear constraint, the PCG method may fail due to detection of negative curvature. This is indicated in the table by an “n” next to the number of iterations.
Table 6.1: Problem 1 preconditioner results

<table>
<thead>
<tr>
<th>dtol</th>
<th>τ</th>
<th>nnz(L)/nnz</th>
<th>cg. time</th>
<th>cg. its</th>
<th>rel. resid.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>2.256e+0</td>
<td>86</td>
<td>1.36e-8</td>
</tr>
<tr>
<td>0.0</td>
<td>2.22e-16</td>
<td>5.045</td>
<td>2.361e-1</td>
<td>5</td>
<td>5.83e-11</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>2.22e-16</td>
<td>4.744</td>
<td>7.752e-1</td>
<td>29</td>
<td>9.04e-8</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>2.22e-16</td>
<td>4.661</td>
<td>8.077e+0</td>
<td>186</td>
<td>2.06e-6</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>2.22e-16</td>
<td>4.550</td>
<td>1.279e+1</td>
<td>300</td>
<td>3.92e-4</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>2.22e-16</td>
<td>4.383</td>
<td>1.237e+1</td>
<td>300</td>
<td>4.56e-4</td>
</tr>
<tr>
<td>0.0</td>
<td>1.00e-1</td>
<td>5.045</td>
<td>2.803e-1</td>
<td>6</td>
<td>1.55e-7</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>1.00e-1</td>
<td>4.564</td>
<td>5.185e-1</td>
<td>12</td>
<td>2.91e-8</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>1.00e-1</td>
<td>4.122</td>
<td>1.949e+0</td>
<td>50</td>
<td>4.48e-7</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>1.00e-1</td>
<td>3.128</td>
<td>8.336e+0</td>
<td>274</td>
<td>5.78e-7</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>1.00e-1</td>
<td>1.658</td>
<td>4.385e+0</td>
<td>246</td>
<td>1.03e-6</td>
</tr>
<tr>
<td>0.0</td>
<td>1.00e+0</td>
<td>5.045</td>
<td>2.349e-1</td>
<td>5</td>
<td>1.29e-6</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>1.00e+0</td>
<td>4.542</td>
<td>2.558e-1</td>
<td>6</td>
<td>3.90e-6</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>1.00e+0</td>
<td>4.093</td>
<td>5.811e-1</td>
<td>15</td>
<td>3.52e-7</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>1.00e+0</td>
<td>3.108</td>
<td>1.177e+0</td>
<td>39</td>
<td>4.00e-7</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>1.00e+0</td>
<td>1.209</td>
<td>5.584e-1</td>
<td>40</td>
<td>4.86e-7</td>
</tr>
</tbody>
</table>

Table 6.2: Problem 2 preconditioner results

<table>
<thead>
<tr>
<th>dtol</th>
<th>τ</th>
<th>nnz(L)/nnz</th>
<th>cg. time</th>
<th>cg. its</th>
<th>rel. resid.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>7.881</td>
<td>300</td>
<td>3.03e-3</td>
</tr>
<tr>
<td>0.0</td>
<td>2.22e-16</td>
<td>5.045</td>
<td>3.267e-1</td>
<td>7</td>
<td>2.50e-11</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>2.22e-16</td>
<td>5.013</td>
<td>9.717e-1</td>
<td>21</td>
<td>3.54e-8</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>2.22e-16</td>
<td>4.993</td>
<td>6.264e+0</td>
<td>135</td>
<td>1.63e-7</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>2.22e-16</td>
<td>4.957</td>
<td>1.379e+1</td>
<td>300</td>
<td>1.76e-5</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>2.22e-16</td>
<td>4.840</td>
<td>1.346e+1</td>
<td>300</td>
<td>2.31e-4</td>
</tr>
<tr>
<td>0.0</td>
<td>1.00e-1</td>
<td>5.045</td>
<td>3.722e-1</td>
<td>8</td>
<td>6.50e-8</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>1.00e-1</td>
<td>4.940</td>
<td>5.032e-1</td>
<td>11</td>
<td>4.18e-8</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>1.00e-1</td>
<td>4.669</td>
<td>2.065e+0</td>
<td>48</td>
<td>1.57e-7</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>1.00e-1</td>
<td>3.729</td>
<td>1.060e+1</td>
<td>300</td>
<td>6.88e-6</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>1.00e-1</td>
<td>1.830</td>
<td>5.721e+0</td>
<td>300</td>
<td>1.64e-5</td>
</tr>
<tr>
<td>0.0</td>
<td>1.00e+0</td>
<td>5.045</td>
<td>2.787e-1</td>
<td>6</td>
<td>7.69e-8</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>1.00e+0</td>
<td>4.933</td>
<td>2.754e-1</td>
<td>6</td>
<td>3.50e-6</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>1.00e+0</td>
<td>4.698</td>
<td>3.924e-1</td>
<td>9</td>
<td>5.98e-7</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>1.00e+0</td>
<td>3.751</td>
<td>1.935e+0</td>
<td>55</td>
<td>3.59e-7</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>1.00e+0</td>
<td>1.308</td>
<td>5.870e-1</td>
<td>40</td>
<td>3.72e-7</td>
</tr>
</tbody>
</table>
Table 6.3: Problem 3 preconditioner results

<table>
<thead>
<tr>
<th>dtol</th>
<th>( \tau )</th>
<th>( \text{nnz}(L)/\text{nnz} )</th>
<th>cg. time</th>
<th>cg. its</th>
<th>rel. resid.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>7.861e+0</td>
<td>300</td>
<td>1.96e-4</td>
</tr>
<tr>
<td>0.0</td>
<td>2.22e-16</td>
<td>5.045</td>
<td>1.411e-1</td>
<td>3.3</td>
<td>1.82e-7</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>2.22e-16</td>
<td>4.501</td>
<td>2.658e-1</td>
<td>64.3</td>
<td>5.95e-3</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>2.22e-16</td>
<td>4.292</td>
<td>7.195e+0</td>
<td>179.3</td>
<td>1.57e-2</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>2.22e-16</td>
<td>3.992</td>
<td>9.392e+0</td>
<td>249.3</td>
<td>6.45e-3</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>2.22e-16</td>
<td>3.564</td>
<td>1.025e+1</td>
<td>300.3</td>
<td>5.63e-4</td>
</tr>
<tr>
<td>0.0</td>
<td>1.00e-1</td>
<td>5.045</td>
<td>1.426e-1</td>
<td>3.0</td>
<td>3.70e-8</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>1.00e-1</td>
<td>4.633</td>
<td>1.815e-1</td>
<td>4.0</td>
<td>3.28e-7</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>1.00e-1</td>
<td>4.365</td>
<td>3.857e-1</td>
<td>9.0</td>
<td>4.05e-5</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>1.00e-1</td>
<td>4.105</td>
<td>7.664e-1</td>
<td>19.0</td>
<td>1.82e-4</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>1.00e-1</td>
<td>2.756</td>
<td>6.021e+0</td>
<td>213.0</td>
<td>4.15e-2</td>
</tr>
<tr>
<td>0.0</td>
<td>1.00e+0</td>
<td>5.045</td>
<td>1.475e-1</td>
<td>3.0</td>
<td>1.50e-8</td>
</tr>
<tr>
<td>1.0e-4</td>
<td>1.00e+0</td>
<td>4.581</td>
<td>2.660e-1</td>
<td>6.0</td>
<td>2.05e-6</td>
</tr>
<tr>
<td>1.0e-3</td>
<td>1.00e+0</td>
<td>4.292</td>
<td>2.939e-1</td>
<td>7.0</td>
<td>1.30e-7</td>
</tr>
<tr>
<td>1.0e-2</td>
<td>1.00e+0</td>
<td>3.780</td>
<td>1.298e+0</td>
<td>35.0</td>
<td>1.76e-4</td>
</tr>
<tr>
<td>1.0e-1</td>
<td>1.00e+0</td>
<td>1.401</td>
<td>6.380e-1</td>
<td>39.0</td>
<td>2.48e-7</td>
</tr>
</tbody>
</table>

The relatively small values for \( \text{nnz}(L)/\text{nnz} \), time, and number of PCG iterations appear to indicate that an effective preconditioning strategy can be built with a large drop tolerance and threshold value.

6.5 Testing the optimization algorithm

The main emphasis of this thesis has been the development of a viable preconditioning strategy for linear systems arising in PDE constrained optimization. Preliminary results are shown that illustrate the performance of the preconditioner when it is used within the primal-dual augmented Lagrangian method discussed in Chapter 2. The results are also compared with the output from PLTMG. There are many parameters that control the performance for both the preconditioner, and the primal-dual augmented Lagrangian method. For this reason, these results are in no sense complete. Testing and optimizing the algorithms is a work in progress.

6.5.1 The optimization algorithm in PLTMG

The PLTMG software package is designed to solve elliptic PDEs over two-dimensional domains. It uses a primal interior-point algorithm for solving optimal control problems. By
default, the optimal control solver assumes that the finite-element spaces for the state and control are the same, and that all components of \( b_\ell \) and \( b_u \) are finite. For a fixed barrier parameter \( \mu \), PLTMG seeks a saddle-point for the barrier Lagrangian function,

\[
L_\mu(x, \pi) = f_\mu(x) - c(x)^T \pi
\]
discussed in Chapter 2. A truncated Newton line-search method is used to find a saddle-point. The solution of the unregularized KKT matrices is computed approximately using the composite step biconjugate-gradient method of Bank and Chan [1].

For a fixed iterate \( v_j = (x_j, \pi_j) \), and search direction \( s_j = \Delta v_j \), the line-search method uses polynomial interpolation to find an \( \alpha_j \) satisfying the Armijo condition

\[
\|F_\mu(v_j + \alpha_j s_j)\|_h^2 \leq (1 - \eta_s \alpha_j)\|F_\mu(v_j)\|_h^2,
\]
subject to the restriction that \( \alpha_j \leq \alpha_{j}^{\text{max}} \), where \( \alpha_{j}^{\text{max}} \) satisfies the fraction-to-the-boundary rule,

\[
\alpha_{j}^{\text{max}} = \max\{\alpha \in (0, 1] : c_\ell(x_j + \alpha s_j) \geq (1 - \tau)c_\ell(x_j)\},
\]
for a non-negative parameter \( \tau \) close to one.

### 6.6 The proposed optimization algorithm

As described in Chapter 1, a primal-dual augmented Lagrangian method is used. For a fixed barrier parameter \( \mu \), penalty parameter \( \rho \) and multiplier estimate \( \pi^e \), we apply Gertz’s “biased” Wolfe line-search algorithm [22] to minimize \( \mathcal{M}_\mu(v; \pi^e, \rho) \). The preconditioned Steihaug-Toint method is used to solve the \( j \)-th trust-region subproblem

\[
\min_{s \in \mathbb{R}^{n+m}} Q_j(s) \quad \text{subject to} \quad \|s\|_{\bar{B}_j} \leq \delta_j
\]
with positive-definite preconditioner \( \bar{B}_j \), where \( Q_j(s) \) denotes the quadratic model

\[
Q_j(s) = \nabla \mathcal{M}_\mu(v_j; \pi^e_j, \rho_j)^T s + \frac{1}{2} s^T B_\mu(v_j; \pi^e_j, \rho_j) s.
\]
The preconditioned Steihaug-Toint method terminates successfully with a search direction \( s_j \) inside the trust-region if

\[
\|g^\mu_j + B^\mu_j s_j\|_{\bar{B}^{-1}_j} \leq \tau_1 \|g_j\|_{\bar{B}^{-1}_j}, \quad \text{where} \quad \tau_1 = \min\left\{10^{-2}, \|g_j\|_{\bar{B}^{-1}_j}^{0.1}\right\},
\]
with respect to the preconditioned norm. In the termination test, \( g^\mu_j = \nabla \mathcal{M}_\mu(v_j; \pi^e_j, \rho_j) \), and \( B^\mu_j = B_\mu(v_j; \pi^e_j, \rho_j) \).
The trust-region step from Steihaug-Toint is used as a search direction in a second-order line-search based on the Wolfe conditions. Let

\[ M_j(\alpha) \triangleq M_\mu(v_j + \alpha s_j; \pi^e_j, \rho_j) \quad \text{and} \quad q_j(\alpha) = Q_j^{-1}(\alpha s_j). \]

With this notation, an acceptable step-length is found when,

\[ M_j(\alpha_j) - M_j(0) \leq q_j(\alpha_j) \quad \text{and} \quad |M_j(\alpha_j)| \leq \eta_w |q_j'(\alpha_j)|. \tag{6.6} \]

The Wolfe line-search from the SQP software package SNOPT [26] has been modified to take advantage of second derivatives. In the numerical tests, \( \eta_w = 0.9 \) and \( \eta_s = 10^{-4} \). Not only does the Wolfe line-search ensure that sufficient progress is made at each iteration of the trust-region algorithm, but the step-length is used to update the trust-region radius. The following trust-region prevents a decrease in the trust-region radius whenever possible.

**Algorithm 6.1** Biased Wolfe line-search

Specify constants \( 0 < \eta_1 < 1 < \gamma_g \);

j = 0;

Choose \( \delta_0 \) and let \( v_0 = (x_0, \pi_0) \);

while not converged do

Compute \( s_j \) as an approximate solution of \( \min \left\{ Q_j(s) : \|s\|_{\bar{B}_j} \leq \delta_j \right\} \);

Find \( \alpha_j \) such that satisfying the Wolfe conditions (6.6);

\[ v_{j+1} = v_j + \alpha_j s_j; \]
\[ \rho_j = (M_\mu(v_{j+1}; \pi^e, \rho) - M_\mu(v_j; \pi^e, \rho))/Q(s_j) \]

if \( \rho_j \geq \eta_1 \) then

\[ \delta_{j+1} = \gamma_g \delta_j; \]

else

\[ \delta_{j+1} = \min \left\{ \alpha_j \|s_j\|_{\bar{B}_j}, \alpha_j \delta_j \right\}; \]

end if

\[ j = j + 1; \]

end do

The numerical tests use \( \gamma_g = \frac{3}{2} \) and \( \eta_1 = \frac{1}{4} \).

### 6.6.1 Optimization results

To compare the primal-dual augmented Lagrangian method with the approach in PLTMG results are collected for Problems 1 and 2. The following table reports the results as determined by PLTMG.
Table 6.4: PLTMG results

<table>
<thead>
<tr>
<th>Problem</th>
<th>$f(x^*)$</th>
<th>$|c(x^*)|_\infty$</th>
<th>its.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.877e-1</td>
<td>2.998e-16</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>3.338e-1</td>
<td>7.511e-10</td>
<td>6</td>
</tr>
</tbody>
</table>

The tables shown below compare the results for Problems 1 and 2, over a sequence of drop tolerances, and \textit{a priori} thresholds. The “l” indicates a failed line-search. When this occurs, the optimization algorithm is terminated. The minimization is terminated at 50 iterations.

Table 6.5: Optimization results for Problem 1

<table>
<thead>
<tr>
<th>dtol</th>
<th>$\tau$</th>
<th>$f(x^*)$</th>
<th>$|c(x^*)|_h$</th>
<th>its.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00e+0</td>
<td>1.00e-8</td>
<td>3.878e-1</td>
<td>8.454e-6</td>
<td>1</td>
</tr>
<tr>
<td>1.00e-6</td>
<td>1.00e-8</td>
<td>3.878e-1</td>
<td>8.454e-6</td>
<td>2</td>
</tr>
<tr>
<td>1.00e-6</td>
<td>1.00e+0</td>
<td>3.878e-1</td>
<td>8.674e-6</td>
<td>2</td>
</tr>
<tr>
<td>1.00e-3</td>
<td>1.00e-8</td>
<td>3.878e-1</td>
<td>8.444e-6</td>
<td>2</td>
</tr>
<tr>
<td>1.00e-3</td>
<td>1.00e+0</td>
<td>3.878e-1</td>
<td>8.454e-6</td>
<td>2</td>
</tr>
<tr>
<td>1.00e-1</td>
<td>1.00e-8</td>
<td>3.874e-1</td>
<td>1.412e-4</td>
<td>50</td>
</tr>
<tr>
<td>1.00e-1</td>
<td>1.00e+0</td>
<td>3.878e-1</td>
<td>1.704e-4</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 6.6: Optimization results for Problem 2

<table>
<thead>
<tr>
<th>dtol</th>
<th>$\tau$</th>
<th>$f(x^*)$</th>
<th>$|c(x^*)|_\infty$</th>
<th>its.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00e+0</td>
<td>1.00e-8</td>
<td>3.338e-1</td>
<td>1.035e-5</td>
<td>1</td>
</tr>
<tr>
<td>1.00e-6</td>
<td>1.00e-8</td>
<td>3.338e-1</td>
<td>1.035e-5</td>
<td>1</td>
</tr>
<tr>
<td>1.00e-6</td>
<td>1.00e+0</td>
<td>7.402e-1</td>
<td>2.497e-3</td>
<td>50</td>
</tr>
<tr>
<td>1.00e-3</td>
<td>1.00e-8</td>
<td>3.338e-1</td>
<td>1.017e-4</td>
<td>2</td>
</tr>
<tr>
<td>1.00e-3</td>
<td>1.00e+0</td>
<td>3.477e+0</td>
<td>3.477e-1</td>
<td>8$^l$</td>
</tr>
<tr>
<td>1.00e-1</td>
<td>1.00e-8</td>
<td>7.588e-1</td>
<td>7.426e+1</td>
<td>50</td>
</tr>
<tr>
<td>1.00e-1</td>
<td>1.00e+0</td>
<td>8.455e-1</td>
<td>6.877e-1</td>
<td>28$^l$</td>
</tr>
</tbody>
</table>
Figure 6.1: State variable predicted by PLTMG for Problem 1

Figure 6.2: Control variable predicted by PLTMG for Problem 1
Figure 6.3: Lagrange multiplier predicted by PLTMG for Problem 1

Figure 6.4: State variable predicted by PLTMG for Problem 2
Figure 6.5: Control variable predicted by PLTMG for Problem 2

Figure 6.6: Lagrange multiplier predicted by PLTMG for Problem 2
Figure 6.7: State variable predicted by PLTMG for Problem 3

Figure 6.8: Control variable predicted by PLTMG for Problem 3
Figure 6.9: Lagrange multiplier predicted by PLTMG for Problem 3

6.7 Conclusion

As discussed above, the algorithms presented in this thesis are preliminary, and future versions will be optimized for performance. We would like to implement a preconditioning strategy that does not assume an *a priori* threshold, thereby reducing the number of parameters. Many of the most effective preconditioning strategies for PDE stiffness matrices are based on *multilevel strategies*. It is currently unknown whether multilevel techniques can be used in the tiled setting and this question will be investigated further. There are many ways to refine the primal-dual augmented Lagrangian method presented in this thesis. Eventually, it is desirable to be able to determine when and how the penalty parameter, barrier parameter, and multiplier estimates should be updated.
Bibliography


