RICE UNIVERSITY
A Robust Trust Region Algorithm for Nonlinear Programming
by
Karen Anne Williamson
A Thesis Submitted
in Partial Fulfillment of the Requirements for the Degree
Doctor of Philosophy

Approved, Thesis Committee:

John E. Dennis, Jr., Chairman
Noah Harding Professor of Mathematical Sciences

J. Boyd Pearson, Jr.
J. S. Abercrombie Professor in the Department of Electrical and Computer Engineering

H. Georg Bock
Professor für Angewandte Mathematik Universität Augsburg, Bavaria, Germany

Dan C. Sorensen
Professor of Mathematical Sciences

Richard A. Tapia
Professor of Mathematical Sciences

Houston, Texas
April, 1990
A Robust Trust Region Algorithm for Nonlinear Programming

Karen Anne Williamson

Abstract

This paper develops and tests a trust region algorithm for the nonlinear equality constrained optimization problem. Our goal is to develop a robust algorithm that can handle lack of second-order sufficiency away from the solution in a natural way. Celis, Dennis and Tapia [1985] give a trust region algorithm for this problem, but in certain situations their trust region subproblem is too difficult to solve. The algorithm given here is based on the restriction of the trust region subproblem given by Celis, Dennis and Tapia [1985] to a relevant two-dimensional subspace. This restriction greatly facilitates the solution of the subproblem. The trust region subproblem that is the focus of this work requires the minimization of a possibly non-convex quadratic subject to two quadratic constraints in two dimensions. The solution of this problem requires the determination of all the global solutions, and the non-global solution, if it exists, to the standard unconstrained trust region subproblem. Algorithms for approximating a single global solution to the unconstrained trust region subproblem have been well-established. Analytical expressions for all of the solutions will be derived for a number of special cases, and necessary and sufficient conditions are given for the existence of a non-global solution for the general case of the two-dimensional unconstrained trust region subproblem. Finally, numerical results are presented for a preliminary implementation of the algorithm, and these results verify that it is indeed robust.
Acknowledgments

I would like to thank the members of my committee for their attention. I owe special thanks to John Dennis whose faith and encouragement helped me through all of those trying moments when I thought I wouldn’t make it.

I would also like to thank the people at Rice with whom I shared the difficult experience of graduate school: Bill LeFebvre, Virginia Torzcon and Cathy Samuelsen, and I would like to thank Dan Davison for his continuing support.

Finally, I would like to thank Duncan and Nyssa for their continual affection, and I would like to thank Mike Alford for his love and encouragement.
## Contents

Abstract ii  
Acknowledgments iii  
List of Tables vi  

1 Introduction 1

2 Some Trust Region Subproblems for Equality Constrained Optimization 4  
2.1 Background 4  
2.2 The New Trust Region Subproblem 7  
2.3 Overview of the Algorithm 7  

3 Solution of the Quadratic Program 9  
3.1 Formulation of the Algorithm 12  
3.2 Statement of the algorithm 15  

4 Calculation of a Trial Step 17  
4.1 Using the Vardi subproblem when ($\nabla h_c h_c = 0$) 17  
4.2 Determining the Required Amount of Linear Feasibility and the Choice of the Two-dimensional Subspace 19  
4.3 Using the Vardi subproblem in two more special cases 21  
4.4 Solution of constrained trust region subproblem 2DCTR 23  
4.5 Conversion of Subproblem 2DCTR to Two dimensions 26  
4.6 Conversion of Problem LF into standard trust region form 27  
4.7 Statement of the Algorithm 29  

5 Solution of the Unconstrained Trust Region Subproblem Restricted to Two Dimensions 33  
5.1 Preliminaries 34
# Tables

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Multiplier Test Results</td>
<td>75</td>
</tr>
<tr>
<td>6.2 Convergence Results</td>
<td>84</td>
</tr>
<tr>
<td>6.3 More Convergence Results</td>
<td>85</td>
</tr>
<tr>
<td>6.4 Local Solutions Found</td>
<td>86</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In this work, we will consider the nonlinear equality constrained optimization problem:

Problem EQ: \( \begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h_i(x) = 0, \quad i = 1, \ldots, m,
\end{align*} \) \hspace{1cm} (1.1)

where \( f \) and \( h_i \) are assumed to be smooth nonlinear functions such that \( f : IR^n \to IR \), \( h_i : IR^n \to IR \) for \( i = 1, \ldots, m \), and \( (m \leq n) \). We will denote by \( h(x) \) the vector \((h_1(x), h_2(x), \ldots, h_m(x))^T\). The Lagrangian function associated with Problem EQ is the function

\[ l(x, \lambda) = f(x) + \lambda^T h(x) \] \hspace{1cm} (1.2)

where \( \lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)^T \) are the Lagrange multipliers. The augmented Lagrangian function associated with Problem EQ is the function

\[ L(x, \lambda, \rho) = f(x) + \lambda^T h(x) + \rho h(x)^T h(x) \] \hspace{1cm} (1.3)

with penalty constant \( \rho \geq 0 \).

We will assume that problem EQ has a solution \( x_* \). The standard assumptions for the analysis of Newton-type methods applied to problem EQ are

1. The functions \( f \) and \( h_i \) have continuous second derivatives in an open neighborhood \( D \) of a local solution \( x_* \) of problem EQ and these second derivatives are Lipschitz continuous at \( x_* \).

2. \( \nabla h(x_*) \) has full rank.

3. \( z^T \nabla^2 l(x_*, \lambda_*) z > 0 \) for all \( z \neq 0 \) satisfying \( \nabla h^T(x_*) z = 0 \).

If assumptions 1 and 2 hold, then necessary conditions for \( x_* \) to be a solution of problem EQ are that there exists \( \lambda_* \in IR^m \) such that \((x_*, \lambda_*)\) is a solution of the
nonlinear system of equations

$$\nabla_x l(x, \lambda) = 0$$
$$h(x) = 0,$$

and

$$z^T \nabla^2_x l(x_*, \lambda_*) z \geq 0 \text{ for all } z \neq 0 \text{ satisfying } \nabla h^T(x_*) z = 0. \quad (1.5)$$

Assumption 3 is the standard second-order sufficiency condition requiring the Hessian of the Lagrangian to be positive definite on the null space of $\nabla h^T$ at the solution.

Since we are interested in iterative methods, we will use $x_c$ to denote the current iterate and the subscript (+) for quantities at the next iteration. Subscripted values of functions represent evaluation at a particular point. For example, $f_c \equiv f(x_c)$, and $l_+ \equiv l(x_+, \lambda_+)$. We use $B(x, \lambda)$ to denote the Hessian of the Lagrangian with respect to $x$, $\nabla^2_x l(x, \lambda)$, or an approximation to it. Finally, all vector norms $\| \cdot \|$ are the 2-norm unless they are specifically labelled otherwise.

One of the most successful methods for solving the equality constrained optimization problem is the successive quadratic programming (SQP) method. At each iteration, the SQP method solves a quadratic program of the form

Problem QP:

$$\begin{align*}
\text{minimize} & \quad \nabla_x l(x_c, \lambda_c)^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \nabla h(x_c)^T s + h(x_c) = 0,
\end{align*} \quad (1.6)$$

for the step $s_c$ and the associated multiplier $\Delta \lambda_c$. Thus, a SQP method solves a sequence of quadratic programming problems of the form (1.6) for steps $s_c$ and multipliers $\Delta \lambda_c$ and then takes $x_+ = x_c + s_c$ and $\lambda_+ = \lambda_c + \Delta \lambda_c$.

To clarify the terminology concerning the multipliers, notice that $\Delta \lambda_c$ is the multiplier step, or change in the multipliers, for the iterative SQP algorithm. Using the form of the quadratic model given in (1.6), $\Delta \lambda_c$ are the multipliers associated with the solution of problem QP, and they are the change in the multipliers for the SQP algorithm.

Of course, the solution to this quadratic program, which we will denote by $s_{QP}$, may fail to exist for several reasons, some more serious than others for standard SQP implementations. Frequently, assumptions 2 and 3 are implicitly assumed to hold not only at the solution, but for all intermediate iterates $(x_c, \lambda_c)$. Since our
goal is to develop a robust nonlinear programming algorithm, we specifically will not assume that $\nabla h(x)$ has full rank or that second-order sufficiency holds, except at the solution. If $\nabla h_c$ does not have full rank, the linearized constraints, $\nabla h_c^T s + h_c = 0$, may be degenerate, or there may not exist a feasible point for problem QP at each intermediate iteration. We will discuss dealing with these situations in Chapter 3.

The more fundamental difficulty in the definition of the SQP step is that second-order sufficiency need not hold at any intermediate iteration. By this we mean that $B(x, \lambda)$ need not be positive definite on the null space of $\nabla h^T$. If there is a direction of negative curvature inside the null space of $\nabla h^T$, then the quadratic model of the Lagrangian is unbounded below on the set of feasible points. This situation can also happen if there is a direction of zero curvature inside the null space of $\nabla h^T$. Near a solution to problem EQ (1.1), this difficulty should not arise because of the standard assumptions, and, locally, the SQP method performs very well. Away from the solution, however, any acceptable algorithm must be prepared to choose a step based on a globalization strategy, particularly when second-order sufficiency does not hold. The issue of a satisfactory globalization strategy still remains open. A number of line search techniques have been proposed, but none of them have proven to be entirely successful. Our approach will use a trust region strategy to handle non-positive curvature in a natural way.
Chapter 2

Some Trust Region Subproblems for Equality Constrained Optimization

Trust region algorithms have been very successful in the solution of nonlinear equations and unconstrained minimization problems where they deal very naturally with negative or zero curvature in the objective function. In this chapter, we will first describe some trust region subproblems that have been proposed to extend the trust region concept to equality constrained optimization. Then we will present the trust region subproblem that will be the focus of this work.

2.1 Background

Consider the essence of trust region algorithms for unconstrained optimization. They are centered around Newton's method, a method with fast local convergence properties. At each iteration, we build a quadratic model $q_c(s)$ of the objective function around the current point $x_c$ and calculate the Newton step for this model. The trust region serves to restrict the step to a region of the form $\|s\| \leq \Delta_c$ in which we trust the model. If the Newton step is inside the trust region, then we will take it as our trial step $s_c$. Otherwise, we choose the trial step to be the solution of the unconstrained trust region subproblem

$$\begin{align*}
\text{minimize} & \quad q_c(s) \\
\text{subject to} & \quad \|s\| \leq \Delta_c.
\end{align*}$$

Once we have a trial step, we must decide if $x_+ = x_c + s_c$ is a better approximation to the solution $x_*$ than the current point. If it is, then we accept the step and start the next iteration with the new iterate $x_+$. If the step is not acceptable, then we reduce the radius of the trust region and calculate a new trial step in this smaller region.

Trust region algorithms for problem EQ contain all of the basic ingredients of unconstrained trust region algorithms. They are based on the SQP method which has fast local convergence properties. The SQP step will play the role that the Newton
step plays in unconstrained optimization. If the SQP step does not exist or if it is too long, then the trial step will be the solution to a constrained trust region subproblem. A variety of constrained trust region subproblems have been proposed.

The most straightforward way to extend the trust region idea to SQP is to simply add a trust region constraint, \( \|s\| \leq \Delta \), to the SQP subproblem. This leads to a subproblem of the form

\[
\begin{align*}
\text{minimize} & \quad \nabla x_l c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \nabla h_c^T s + h_c = 0 \\
& \quad \|s\| \leq \Delta_c.
\end{align*}
\]

If the current iterate is a nonlinearly feasible point, i.e., \( h(x_c) = 0 \), then this subproblem can be solved as a lower dimensional unconstrained trust region subproblem. However, if \( x_c \) is not a feasible point, then this subproblem may not have a solution. The difficulty is that the feasible set may be empty, for the linearized constraints \( \nabla h_c^T s + h_c = 0 \) may not intersect the trust region.

Vardi [1980, 1985] studies a trust region subproblem of the form

Vardi Subproblem: \hspace{1cm} (2.1)

\[
\begin{align*}
\text{minimize} & \quad \nabla x_l c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \nabla h_c^T s + h_c = \Theta_c \\
& \quad \|s\| \leq \Delta_c,
\end{align*}
\]

where \( \Theta_c \in \mathbb{R}^m \) is a multiple of \( h_c \) chosen to ensure that the feasible set is non-empty. The Vardi subproblem can be transformed into an unconstrained trust region subproblem of a lower dimension, and then existing algorithms from unconstrained trust region methods can be used to obtain the solution at each iteration. This method handles lack of second-order sufficiency away from the solution with no difficulty, but there remains the problem of the specific choice of \( \Theta_c \).

Celis, Dennis, and Tapia [1985] avoid this difficulty by considering a subproblem of the form

CDT Subproblem: \hspace{1cm} (2.2)

\[
\begin{align*}
\text{minimize} & \quad \nabla x_l c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|\nabla h_c^T s + h_c\| \leq \theta_c \\
& \quad \|s\| \leq \Delta_c,
\end{align*}
\]
where $\theta_c \in \mathbb{R}$ is chosen to be $\|\nabla h_c^T s + h_c\|$ for some $s$ inside the trust region. In this way, the feasible set in the CDT subproblem is guaranteed to be non-empty. Celis, Dennis and Tapia chose $\theta_c$ to be $\|\nabla h_c^T s_{CP} + h_c\|$ where $s_{CP} = \alpha_c \nabla h_c h_c$ is the step to the Cauchy point for the constraints, i.e., the minimizer inside the trust region $\{s : \|s\| \leq \Delta_c\}$ of $\|\nabla h_c^T s + h_c\|$ along the direction of its negative gradient. This is enough to ensure that nonlinear feasibility will be attained in the limit, but it allows flexibility for the subproblem to progress towards optimality. Furthermore, El-Alem [1988] gives a global convergence proof for a variant of the algorithm given by Celis, Dennis and Tapia [1985] which uses a different strategy for updating the penalty constant.

Powell and Yuan [1986] also consider a subproblem of the same form as the CDT subproblem with a different choice of $\theta_c$. They chose it to be any number that satisfies

$$\theta_c = \min\{\|\nabla h_c^T s + h_c\| : \|s\| \leq \sigma \Delta_c\} \quad (2.3)$$

for some $0 < \sigma \leq 1$. This choice of $\theta_c$ is computationally more expensive than the choice based on the Cauchy point for the constraints, but it will provide faster convergence to nonlinear feasibility. However, getting nonlinearly feasible too early can cause an expensive trip around a curved boundary of the feasible region, and numerical experimentation supports this notion, (Dennis, El-Alem and Tapia, [1989]). A conceptual advantage of $\theta_c$ given by (2.3) is that the SQP step would be chosen automatically whenever it is inside the trust region. Instead, Celis, Dennis and Tapia [1985] compute the SQP step and take it as $s_c$ whenever it is inside the trust region. Notice that when the SQP step is inside the trust region, it is not necessarily the solution to the CDT subproblem. The solution to the CDT subproblem could give a smaller value of the quadratic model than the SQP step but have a larger residual of the linearized constraints.

Celis, Dennis and Tapia [1985] give an algorithm for solving the CDT subproblem. Using this subproblem, they developed an algorithm for solving the equality constrained optimization problem which compared favorably with two existing SQP implementations. However, at the time of the original development of the CDT algorithm, a characterization of the solutions to the CDT subproblem was not known. In essence, this is the problem of minimizing a non-convex quadratic subject to two quadratic constraints. When both of the quadratic constraints are binding, the CDT subproblem is too difficult and expensive to solve.
2.2 The New Trust Region Subproblem

Motivated by the work of Byrd, Schnabel and Shultz [1986] on trust region methods for unconstrained optimization, Dennis, Martinez, Tapia, and Williamson [1990] have proposed a more convenient trust region problem by restricting the CDT subproblem to a relevant two-dimensional subspace. This gives a subproblem of the form

\[
\begin{aligned}
\text{2DCTR Subproblem:} \\
\text{minimize} & \quad \nabla_x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|\nabla h_c^T s + h_c\|_2 \leq \theta_c \\
& \quad \|s\|_2 \leq \Delta_c \\
& \quad s \in \text{span}\{v_1, v_2\}.
\end{aligned}
\]

We will refer to this subproblem as the 2DCTR (2-Dimensional Constrained Trust Region) subproblem. For the required amount of linear feasibility, we choose \(\theta_c\) in a manner similar to Celis, Dennis and Tapia [1985]. We use a dogleg strategy as in unconstrained trust region algorithms to determine the step \(\hat{s}\) which will give us \(\theta_c = \|\nabla h_c^T \hat{s} + h_c\|\). We use the Cauchy point for the constraints and a most linearly feasible point \(s_{LF}\) for the dogleg. More details about the dogleg and calculating the required linear feasibility can be found in Chapter 4.

The 2DCTR subproblem also requires the choice of the relevant two-dimensional subspace. We will use the dogleg step which determined the required linear feasibility as the first direction. Notice that this ensures that the two-dimensional subspace intersects the feasible region given by the two quadratic constraints since the dogleg point determined this region. For the second direction, we will use the SQP step when it exists. If the SQP step does not exist, then we will use a resulting direction of negative or zero curvature inside the null space of \(\nabla h_c^T\) as the second direction.

2.3 Overview of the Algorithm

The remainder of this work is concerned with the specification and solution of the 2DCTR subproblem, and the incorporation of this two-dimensional subproblem into an algorithm for solving problem EQ. Chapter 3 gives our algorithm for solving the quadratic programming problem QP and discusses how we deal with the situation when \(\nabla h_c\) does not have full rank. In addition, we discuss how to obtain a direction of zero or negative curvature when second-order sufficiency does not hold.
In Chapter 4 we give our strategies for calculating a trial step at each iteration of our nonlinear programming algorithm. Included in this chapter is the characterization of the solutions to our two-dimensional subproblem 2DCTR, and the resulting algorithm to solve this constrained trust region subproblem. In essence, this requires the minimization of a non-convex quadratic subject to two quadratic constraints in two dimensions. To solve this subproblem, we will need to be able to find all of the local solutions to the unconstrained trust region subproblem in two dimensions. In Chapter 5, we derive analytical expressions for all the local solutions of the unconstrained trust region subproblem in a number of degenerate situations. We also give necessary and sufficient conditions for the existence of a local, non-global solution in the non-degenerate case. Finally, we give an algorithm for finding all of the local solutions to the two-dimensional unconstrained trust region subproblem. This algorithm completes the calculation of a trial step.

Once we have a trial step, Chapter 6 discusses the criteria we use to accept the step and update the trust region radius. This chapter contains the choice of the merit function which includes the strategy for choosing the penalty constant. We discuss several choices of Lagrange multiplier estimates and numerical experimentation with them. Finally, we give the numerical results for a preliminary implementation of our nonlinear programming algorithm, and we compare it to other existing nonlinear programming codes.
Chapter 3

Solution of the Quadratic Program

In this section we will discuss the solution of the quadratic program

Problem QP:

$$\minimize \quad \nabla x_c^T s + \frac{1}{2} s^T B_c s$$

subject to \( \nabla h^T s + h = 0 \)

for the step \( s_{QP} \) and the associated multipliers \( \Delta \lambda_{QP} \) when the solution exists, and
we will discuss how to handle the quadratic program when the solution does not
exist. Since we will focus attention only on obtaining a solution to problem QP in
this chapter, we will drop the subscript \( c \) which indicates the current iteration in the
nonlinear programming algorithm.

Let us consider the difficulties that can arise in the solution of problem QP. Since
\( \nabla h \) may not have full rank, the algorithm must be able to handle situations in which
the constraints are degenerate. This is not a serious problem, and it can be handled
in a straightforward manner. A more substantial difficulty is that there may not be
a linearly feasible point. In other words, there may not be any step \( s \) which satisfies
the linearized constraints \( \nabla h^T s + h = 0 \). In this case, problem QP will not have a
solution. To overcome this obstacle, let \( s_{LF} \), (a Linearly Feasible step), be a solution
of the linear least-squares problem

$$\minimize \| \nabla h^T s + h \|,$$

and define \( \Theta_{MIN} \) to be the residual of the linearized constraints at \( s_{LF} \),

$$\Theta_{MIN} = \nabla h^T s_{LF} + h.$$  \hspace{1cm} (3.1)

We will replace the linearized constraints in problem QP with \( \nabla h_c^T s + h_c = \Theta_{MIN} \)
which will require the step to be as linearly feasible as possible. Thus, when problem
QP does not have a solution because \( \nabla h \) does not have full rank, we will actually
solve the quadratic program referred to as problem GQP, for a generalized \( s_{QP} \) step.

Problem GQP:

\[
\begin{align*}
\text{minimize} & \quad \nabla x^T c + \frac{1}{2} s^T B c s \\
\text{subject to} & \quad \nabla h^T c s + h_c = \Theta_{MIN}.
\end{align*}
\]  

There should be no confusion in using the notation \( s_{QP} \) in this way since if a linearly feasible point exists, \( \Theta_{MIN} \) will be 0 and problem GQP is identical to problem QP. Thus, if problem QP does not have a solution because there is not a step \( s \) that satisfies \( \nabla h^T s + h = 0 \), then this constraint is relaxed in a meaningful way in problem GQP to obtain a quadratic program which does have a solution.

As indicated in Chapter 1, we will also assume that second-order sufficiency may not hold away from the solution to problem EQ. Recall that the second-order sufficiency condition at the point \( (x, \lambda) \) is

\[ z^T B(x, \lambda) z > 0 \text{ for all } z \neq 0 \text{ satisfying } \nabla h^T(x) z = 0. \]  

If this condition does not hold, then there will be a direction of zero or negative curvature inside the null space of \( \nabla h^T \). We will denote such a direction by \( d_{QP} \).

To see why problem GQP will not have a solution when \( d_{QP} \) is a direction of negative curvature inside the null space of \( \nabla h^T \), consider a step \( s_{LF} \) to the linearized constraint manifold \( \nabla h^T s_{LF} + h = \Theta_{MIN} \). Then, any step of the form \( s = s_{LF} + \alpha d_{QP} \), where \( \alpha \) is a scalar, will also satisfy \( \nabla h^T s + h_c = \Theta_{MIN} \) since \( d_{QP} \) lies in the null space of \( \nabla h^T \). Now the quadratic objective function \( q(s) = \nabla x^T s + \frac{1}{2} s^T B s \) for any step of the form \( s = s_{LF} + \alpha d_{QP} \) is

\[
q(s) = \nabla x^T (s_{LF} + \alpha d_{QP}) + \frac{1}{2} (s_{LF} + \alpha d_{QP})^T B (s_{LF} + \alpha d_{QP})
\]

\[
= q(s_{LF}) + \alpha (\nabla x^T d_{QP} + s_{LF}^T B d_{QP}) + \frac{1}{2} \alpha^2 d_{QP}^T B d_{QP}.
\]

Since \( d_{QP} \) is a direction of negative curvature for \( B \), \( d_{QP}^T B d_{QP} < 0 \). We can choose the sign of \( \alpha \) so that

\[
\alpha (\nabla x^T d_{QP} + s_{LF}^T B d_{QP}) \leq 0.
\]

Then, as we increase the magnitude of \( \alpha \), it is easy to see that \( q(s) \to -\infty \), for any \( s \) of the form \( s_{LF} + \alpha d_{QP} \). Thus, problem GQP will not have a solution since the objective function is unbounded below on the feasible region. Notice that if we choose
the sign of $\alpha$ so that $\alpha \nabla_x l^T d_{QP} \leq 0$, then $s = \alpha d_{QP}$ is a descent direction for the quadratic objective function since $d_{QP}^T B d_{QP} < 0$.

Now, consider the situation when $d_{QP}$ is a direction of zero curvature inside the null space of $\nabla h^T$. Without loss of generality, we will assume that there is not a direction of negative curvature inside the null space of $\nabla h^T$ since we have already shown that problem GQP will not have a solution if such a direction exists. The quadratic objective function for $s = s_{LF} + \alpha d_{QP}$ reduces to

$$q(s) = q(s_{LF}) + \alpha \nabla_x l^T d_{QP}.$$  \hfill (3.4)

If $\nabla x l^T d_{QP} \neq 0$, then $q(s)$ can be shown to be unbounded below in the feasible region. In this case, as in the negative curvature case, problem GQP will not have a solution. Similarly, $s = \alpha d_{QP}$ is a descent direction for $q(s)$ when the sign of $\alpha$ is chosen to satisfy $\alpha \nabla_x l^T d_{QP} \leq 0$. Since we could have more than one direction of zero curvature, problem GQP will not have a solution if any direction of zero curvature is a descent direction for $q(s)$.

On the other hand, if $\nabla x l^T d_{QP} = 0$ for all of the directions of zero curvature inside the null space of $\nabla h^T$, then $q(s) = q(s_{LF})$ for all $\alpha$. Thus, $d_{QP}$ gives us not a descent direction but a direction along which the quadratic is unchanging. However, this does not imply that there is not a descent direction from $s = s_{LF}$. The step $s = s_{LF}$ takes us to the null space of the constraints. To determine if there is a descent direction inside the null space, we will simply minimize the quadratic restricted to the null space. This gives us a step $s_{\text{min}}$ to the minimizer of the quadratic inside the null space. Combining this step with the step to the null space and the direction in which the quadratic is unchanging gives us an infinite number of solutions to problem GQP of the form

$$s = s_{LF} + s_{\text{min}} + \alpha d_{QP}$$  \hfill (3.5)

for all $\alpha$.

Therefore, if there is a direction of zero or negative curvature inside the null space of $\nabla h^T$ which is a descent direction for the quadratic model, the algorithm will calculate the direction $d_{QP}$ as described above and a step $s_{LF}$ to the linearized constraints. If there are directions of zero curvature and none of them are descent directions, then $s = s_{LF} + s_{\text{min}}$ will be one of the infinitely many solutions to problem GQP.
3.1 Formulation of the Algorithm

If $s_{QP}$ is the solution to problem QP and $\Delta \lambda_{QP}$ is the associated multiplier, then $s_{QP}$ and $\Delta \lambda_{QP}$ satisfy

$$
\begin{bmatrix}
B & \nabla h \\
\nabla h^T & 0
\end{bmatrix}
\begin{bmatrix}
s_{QP} \\
\Delta \lambda_{QP}
\end{bmatrix}
= 
- \begin{bmatrix}
\nabla_x \lambda \\
h
\end{bmatrix}.
$$

(3.6)

The first step in the solution of problem GQP is the calculation of the QR decomposition of $\nabla h$ using column pivoting. Notice that this is not the obvious way to solve the linearized constraints in the least-squares sense, but keep in mind that the goal is to solve the system given by (3.6), not just the linearized constraints. The QR decomposition yields

$$
\nabla h = Q R \Pi^T \equiv [Q_1 \ | \ Q_2] R \Pi^T
$$

(3.7)

where $Q$ is $n \times n$ and orthonormal, $R$ is $n \times m$ and upper triangular, and $\Pi$ is $m \times m$ and is the permutation matrix that describes the column pivoting. Let $r$ denote the rank of $\nabla h$. Then, the columns of the matrix $Q$ can be partitioned into two sets, $Q_1$ and $Q_2$. The matrix $Q_1$ has $r$ columns which form an orthonormal basis for the column space of $\nabla h$. The matrix $Q_2$ has $(n - r)$ columns which span the null space of $\nabla h^T$. We will partition $R$ in a similar manner so that

$$
R = \begin{bmatrix}
R_1 & R_2 \\
0 & 0
\end{bmatrix}
$$

where $R_1$ is an $(r \times r)$ nonsingular, upper triangular matrix.

Let $w_1 \in IR^r$ and $w_2 \in IR^{(n-r)}$. The step $s$ can now be represented as the sum of two components, one which lies in the column space of $\nabla h$ and another which lies in the null space of $\nabla h^T$, i.e.

$$
s = Q_1 w_1 + Q_2 w_2.
$$

(3.8)

Since we have not assumed that $\nabla h$ has full rank, we will interpret $\nabla h^T s = -h$ in the least-squares sense. Using the QR decomposition of $\nabla h$ and the representation of $s$ given by (3.8), $\nabla h^T s = -h$ is equivalent to

$$
R_1^T w_1 = -[\Pi^T h]_r
$$

(3.9)
where $[\Pi^T h]_r$ is intended to denote the first $r$ elements of the vector $\Pi^T h$. Therefore, $w_1$, the component of $s$ in the column space of $\nabla h$, can be determined by simply solving the lower triangular system given in (3.9).

Once $w_1$ has been determined, we can calculate $s_{LF}$, a step to the linearized constraint manifold, by

$$s_{LF} = Q_1 w_1. \quad (3.10)$$

We can also calculate $\Theta_{MIN}$, the residual of the linearized constraints, by

$$\Theta_{MIN} = \nabla h^T s_{LF} + h \quad (3.11)$$

$$= \Pi \begin{bmatrix} R_1^T \\ R_2^T \end{bmatrix} w_1 + h \quad (3.12)$$

$$= \Pi \begin{bmatrix} 0 \\ R_2^T w_1 + [\Pi h]_{(m-r)} \end{bmatrix} \quad (3.13)$$

where $[\Pi h]_{(m-r)}$ is intended to denote the last $(m-r)$ elements of the vector $\Pi h$. Notice that if $\nabla h$ has full rank, then $\Theta_{MIN} = 0$.

Now we want to determine if there is a direction of zero or negative curvature for $B$ in the null space of $\nabla h^T$. The Hessian $B$ restricted to the null space of $\nabla h^T$ is $Q^T_2 B Q_2$. Let $\Lambda_1$ denote the smallest eigenvalue of $Q^T_2 B Q_2$, and let $v_1$ denote the corresponding eigenvector. Then, $v_1$ is a direction of negative curvature if $\Lambda_1 < 0$ or a direction of zero curvature if $\Lambda_1 = 0$. If $v_1$ is a direction of zero or negative curvature, we must change from the basis of the null space of $\nabla h^T$ back to the standard basis to obtain $d_{QP}$ by

$$d_{QP} = Q_2 v_1. \quad (3.14)$$

If $\Lambda_1 = 0$ and $\nabla_x l^T Q_2 v_1 = 0$, then $d_{QP} = Q_2 v_1$ is not a descent direction for the quadratic objective function. In this situation, we must determine if there is any eigenvector corresponding to a zero eigenvalue which will give us a descent direction for $q(s)$. If we find a null eigenvector $v_i$ such that $\nabla_x l^T Q_2 v_i \neq 0$, then we will take $d_{QP} = Q_2 v_i$. Otherwise, problem GQP will have an infinite number of solutions. In this case, we must minimize the quadratic restricted to the null space of the constraints for the step $s_{min} = Q_2 v_2^*$. Therefore, $v_2^*$ solves

$$Q_2^T B Q_2 v_2 = -Q_2^T \nabla_x l. \quad (3.15)$$

Thus, if $d_{QP}$ is a direction of zero or negative curvature, then either problem GQP does not have a solution, or it has infinitely many solutions of the form $s_{LF} + s_{min} + \alpha d_{QP}$. In either case, we are finished.
At this point we have determined that there is not a direction of zero or negative curvature inside the null space of $\nabla h^T$. Therefore, we will compute the component of the step $s_{QP}$ that lies inside the null space of $\nabla h^T$ from

$$Bs + \nabla h \Delta \lambda = -\nabla_x l.$$  \hspace{1cm} (3.14)

Substituting the parameterization $s = Q_1 w_1 + Q_2 w_2$ and multiplying from the left by $Q_2^T$ yields

$$[Q_2^T B Q_2] w_2 = -Q_2^T (\nabla_x l + BQ_1 w_1).$$  \hspace{1cm} (3.15)

Since $s_{LF} = Q_1 w_1$, equation (3.15) simplifies to

$$[Q_2^T B Q_2] w_2 = -Q_2^T (\nabla_x l + Bs_{LF}).$$  \hspace{1cm} (3.16)

This linear system can be solved for the remaining component of the step $w_2$ and then the solution of the step is complete with

$$s_{QP} = s_{LF} + Q_2 w_2.$$

The only task remaining is the determination of the associated Lagrange multipliers, $\Delta \lambda$, from equation (3.14). Substituting the QR decomposition of $\nabla h$, equation (3.14) becomes

$$[Q_1 \vert Q_2] \begin{bmatrix} R_1 & R_2 \\ 0 & 0 \end{bmatrix} \Pi \Delta \lambda = -(\nabla_x l + Bs_{QP}).$$  \hspace{1cm} (3.17)

Partition the vector $\Pi \Delta \lambda$ into the first $r$ components, $[\Pi \Delta \lambda]_r$, and the last $m - r$ components, $[\Pi \Delta \lambda]_{(m-r)}$. Since the last $m - r$ components of $\Pi \Delta \lambda$ correspond to dependent columns in $\nabla h$, we will set them to zero, i.e.,

$$[\Pi \Delta \lambda]_{(m-r)} \equiv 0.$$

This allows us to reduce equation (3.17) to

$$Q_1 R_1 [\Pi \Delta \lambda]_r = -(\nabla_x l + Bs_{QP}).$$  \hspace{1cm} (3.18)

Multiplying from the left by $Q_1^T$ yields

$$R_1 [\Pi \Delta \lambda]_r = -Q_1^T (\nabla_x l + Bs_{QP}).$$  \hspace{1cm} (3.19)

The upper triangular system (3.19) can now be solved for $[\Pi \Delta \lambda]_r$, and these elements of $\Pi \Delta \lambda$, along with the elements, $[\Pi \Delta \lambda]_{(m-r)}$, which were set to zero, can be unscrambled to obtain $\Delta \lambda_{QP}$. 
3.2 Statement of the algorithm

The algorithm for the solution of problem GQP when such a solution exists or the determination of a direction of negative or zero curvature inside the null space of $\nabla h$ can be stated as follows.

Algorithm GQP:

0. **Given** $h$, $\nabla h$, $\nabla g$, and $B$, **find** $s_{QP}$ and $\Delta \lambda_{QP}$, if they exist. Otherwise, determine a step $s_{LF}$ that satisfies $\nabla h^T s_{LF} + h = \Theta_{MIN}$ and a direction $d_{QP}$ of zero or negative curvature inside the null space of $\nabla h^T$.

1. Calculate the QR decomposition of $\nabla h$ using column pivoting. 
   $$\nabla h = QR \Pi$$

2. Determine the rank of $\nabla h$. Let the rank of $\nabla h = r$.

3. Partition $Q$ and $R$ such that $Q = [Q_1 \mid Q_2]$ and 
   $$R = \begin{bmatrix} R_1 & R_2 \\ 0 & 0 \end{bmatrix}$$
where $Q_1$ has $r$ columns and $R_1$ is $r \times r$ and upper triangular.

4. Solve $R_1^T w_1 = -[\Pi h]_r$ for $w_1$.

5. Calculate the step to the linearized constraint manifold: $s_{LF} = Q_1 w_1$.

6. Calculate the residual of the linearized constraints:
   $$\Theta_{MIN} = \Pi \begin{bmatrix} 0 \\ R_2^T w_1 + [\Pi h]_{(m-r)} \end{bmatrix}$$

7. Determine if $B$ has a direction of zero or negative curvature inside the null space of $\nabla h^T$.
   Form $Q_2^T B Q_2$.
   Find the smallest eigenvalue of $[Q_2^T B Q_2]$, $\Lambda_1$, and the corresponding eigenvector, $v_1$.

   **If** ($\Lambda_1 \leq 0$), **then**
\[ d_{QP} = Q_2 v_1 \]
\[ \text{negcurv} = \text{true} \]

If \((\Lambda_1 = 0)\) and \((\nabla_x l^T d_{QP} = 0)\), then

Determine if there is any eigenvalue, \(\Lambda_i\), of \(Q_2^T B Q_2\) such that \(\Lambda_i = 0\) and the corresponding eigenvector, \(v_i\), satisfies \(\nabla_x l^T Q_2 v_i \neq 0\).

If (there exists such a pair \((\Lambda_i, v_i)\)), then
\[ d_{QP} = Q_2 v_i \]
Else
Solve \(Q_2^T B Q_2 v_2 = -Q_2^T \nabla_x l\) for \(v_2^*\).
\[ s_{LF} \leftarrow s_{LF} + Q_2 v_2^* \]
End if

Else
\[ \text{negcurv} = \text{false} \]
End if

8. Solve \([Q_2^T B Q_2] w_2 = -Q_2^T (\nabla_x l + B s_{LF})\) for \(w_2\).

9. \[ s_{QP} = s_{LF} + Q_2 w_2 \]

10. Solve \(R_1 [\Pi \Delta \lambda]_r = -Q_1^T (\nabla_x l + B s_{QP})\) for \([\Pi \Delta \lambda]_r\).

11. Set the last \((m - r)\) elements of \(\Pi \Delta \lambda\) to zero.
\[ [\Pi \Delta \lambda]_{(m-r)} \equiv 0. \]

12. \[ \Delta \lambda_{QP} = \Pi [\Pi \Delta \lambda] \]
Return
End
Chapter 4

Calculation of a Trial Step

The focus of this chapter is the calculation of a trial step \( s_c \) at each iteration of our nonlinear programming algorithm. At the current iterate \( x_e \) with multipliers \( \lambda_c \), we will calculate the function information \( f_e \equiv f(x_e) \), \( \nabla f_e \), \( B_c \), \( h_c \) and \( \nabla h_c \). We assume that we have the current trust region radius \( \Delta_c \). Given this information, we want to determine a trial step \( s_c \) which, when added to the current point, will hopefully give us a new iterate \( x_+ = x_e + s_c \) that is a better approximation to \( x_* \) than the current iterate.

First, we will calculate the solution \( s_{QP} \) to the generalized quadratic program GQP,

\[
\text{Problem GQP: } \begin{align*}
\text{minimize} \quad & \nabla x I_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} \quad & \nabla h_c^T s + h_c = \Theta_{MIN},
\end{align*}
\]

as discussed in Chapter 3. Since we want to retain the fast local convergence of the SQP method, we will always take \( s_{QP} \) as our trial step whenever it exists and is inside the trust region.

Now assume that we did not take the SQP step as our trial step since we determined that either the SQP step does not exist or that it is too long. From the solution of problem GQP, we have a step \( s_{LF} \) to the linearized constraint manifold \( \nabla h_c^T s + h_c = \Theta_{MIN} \) and either the solution to the quadratic program \( s_{QP} \) or a direction of negative or zero curvature \( d_{QP} \) inside the null space of \( \nabla h_c^T \).

4.1 Using the Vardi subproblem when \( \nabla h_c h_c = 0 \)

Next we will consider the special situation where \( \nabla h_c h_c = 0 \). This case will occur when either \( x_e \) is a nonlinearly feasible point, i.e. \( h_c = 0 \), or when \( s = 0 \) lies in the linearized constraint manifold \( \nabla h_c^T s + h_c = \Theta_{MIN} \). Recall the trust region subproblem that is
obtained if the trust region constraint is added to problem GQP:

\[
\begin{align*}
\text{minimize} & \quad \nabla_x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \nabla h_c^T s + h_c = \Theta_{MIN} \\
& \quad \|s\| \leq \Delta_c.
\end{align*}
\]

The difficulty with this subproblem is that the feasible set may be empty. However, in the special circumstance that \(\nabla h_c h_c = 0\), then the constraint set is guaranteed to be non-empty. In fact, the linear manifold \(\nabla h_c^T s + h_c = \Theta_{MIN}\) passes through \(s = 0\). In this case, we will solve the following problem for the trial step \(s_c\):

\[
\begin{align*}
\text{minimize} & \quad \nabla_x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \nabla h_c^T s = 0 \\
& \quad \|s\| \leq \Delta_c.
\end{align*}
\] (4.2)

When \(\nabla h_c h_c = 0\), we can show that \(\nabla h_c^T s + h_c = \Theta_{MIN}\) is equivalent to \(\nabla h_c^T s = 0\). By definition, \(\Theta_{MIN}\) is the residual of the linear least squares problem \(\nabla h_c^T s = -h_c\). In addition, \(\nabla h_c h_c = 0\) implies that the projection of \(h_c\) onto the column space of \(\nabla h_c^T\) is zero. Thus, \(\Theta_{MIN} = h_c\) in this case, and \(\nabla h_c^T s + h_c = \Theta_{MIN}\) reduces to \(\nabla h_c^T s = 0\).

There are two different motivations for using the subproblem given above in these circumstances. First, the subproblem given in (4.2) is a special case of the Vardi subproblem (2.1). The difficulty with the Vardi subproblem is the choice of \(\Theta_c\) in the constraint \(\nabla h_c^T s + h_c = \Theta_c\), but in this situation, we clearly know how to choose \(\Theta_c\). Another reason that we choose to solve the subproblem given in (4.2) when \(\nabla h_c h_c = 0\) comes from the form that the constraint in our subproblem would take. In this case, \(\|\nabla h_c^T s + h_c\| \leq \theta_c\) could become \(\|\nabla h_c^T s\| = 0\). However, \(\|\nabla h_c^T s\| = 0\) is essentially an ill-conditioned form of \(\nabla h_c^T s = 0\), Dennis, El Alem, and Tapia [1989], and so numerically we would prefer to solve the subproblem given in (4.2).

Since the subproblem given in (4.2) is a special case of the Vardi subproblem, it can be reduced to a lower dimensional unconstrained trust region subproblem of the form

\[
\begin{align*}
\text{minimize} & \quad \hat{q}(w) \\
\text{subject to} & \quad \|w\| \leq \Delta_c
\end{align*}
\] (4.3)
by the appropriate change of variables. The constraint $\nabla h_c^T s = 0$ requires that the solution lie in the null space of $\nabla h_c^T$. Thus, we can write $s = Q_2 w$ where $Q_2$ is the orthonormal basis for the null space of $\nabla h_c^T$ given in Chapter 3, and the dimension of the resulting unconstrained subproblem will be the dimension of the null space of $\nabla h_c^T$. This standard unconstrained subproblem can then be solved with existing software designed for unconstrained trust region algorithms. See, for example, Moré and Sorensen [1983]. Although we have discussed this special case assuming that we have solved the quadratic program GQP, notice that if $s_{QP}$ exists and lies inside the trust region, then it will be the solution to the subproblem given in (4.2). Thus, for efficiency, the algorithm will solve the subproblem in (4.2) first if it detects the special case of $\nabla h_c h_c = 0$, and in this situation, problem GQP does not need to be solved.

So far, we have chosen the trial step to be $s_{QP}$ if it exists and lies inside the trust region or the solution to the subproblem of the form (4.2) if $\nabla h_c h_c = 0$. To set up our subproblem 2DCTR, we need to determine the two-dimensional subspace that we will work in and how much linear feasibility we will require. These are the topics of the next section.

### 4.2 Determining the Required Amount of Linear Feasibility and the Choice of the Two-dimensional Subspace

Recall that our constrained trust region subproblem 2DCTR has a linear feasibility constraint of the form

$$\|\nabla h_c^T s + h_c\| \leq \theta_c,$$

and in this section we will discuss how to choose $\theta_c$. The strategy we use is based on the idea that if we choose $\theta_c$ to be equal to $\|\nabla h_c^T \hat{s} + h_c\|$ for some $\hat{s}$ inside the trust region, then we are guaranteed that the feasible set of the form

$$\{s : \|\nabla h_c^T s + h_c\| \leq \theta_c \text{ and } \|s\| \leq \Delta_c\}$$

is non-empty. To preclude the possibility that the feasible set given by (4.5) consists of a single point, we choose $\hat{s}$ such that $\|\hat{s}\| \leq .8\Delta_c$. Celis, Dennis and Tapia [1985] chose $\theta_c$ to be $\|\nabla h_c^T s_{CP} + h_c\|$ where $s_{CP} = \alpha_c \nabla h_c h_c$ is the step to the Cauchy point for the constraints, i.e., the minimizer inside the trust region $\{s : \|s\| \leq .8\Delta_c\}$ of $\|\nabla h_c^T s + h_c\|$ along the direction of its negative gradient while Powell and Yuan [1986] chose $\theta_c$ to minimize $\|\nabla h_c^T s + h_c\|$ inside a smaller trust region of radius $\sigma \Delta_c$ for $0 < \sigma \leq 1$. 
Our choice for $\theta_c$ is based on a dogleg strategy similar to the dogleg approach for the solution of the unconstrained trust region subproblem, (Dennis and Schnabel [1983]). In unconstrained optimization, the dogleg approximates the solution curve $s(\mu)$ of the trust region subproblem by a piecewise linear function connecting the Cauchy point to the Newton step. If the Newton step is inside the trust region, then it is the dogleg point. Otherwise, the dogleg step $s_{DL}$ is the point on this polygonal arc such that $\|s_{DL}\| = \Delta_c$. The dogleg has the nice property that the value of the quadratic model decreases monotonically along the curve from $x_c$ to $s_{CP}$ to the Newton step.

We want to determine a dogleg step $s_{DL}$ for the quadratic model of the constraints $\|\nabla h_c^T s + h_c\|^2$. We use the Cauchy point as defined above with a trust region radius of length $.8\Delta_c$ as the first segment of the dogleg. The Cauchy point is given by

$$s_{CP} = -\frac{h_c^T \nabla h_c^T \nabla h_c h_c}{h_c^T \nabla h_c^T \nabla h_c \nabla h_c^T \nabla h_c h_c} \nabla h_c h_c$$  \hspace{1cm} (4.6)

and

$$\text{if } \|s_{CP}\| > .8\Delta_c, \text{ then } s_{CP} = \frac{.8\Delta_c}{\|s_{CP}\|} s_{CP}.$$  

Notice that the Cauchy step cannot be zero since we have already dealt with the case $\nabla h_c h_c = 0$.

Now we need the segment of the dogleg step that will play the role that the Newton step plays in the dogleg for unconstrained optimization. We could use the Levenberg-Marquardt-type step of Powell and Yuan. However, this choice requires the solution of an additional unconstrained trust region subproblem, and we would prefer not to incur this computational expense. Instead, recall that we have a step $s_{LF}$ to the linearized constraint manifold $\nabla h_c^T s + h_c = \Theta_{MIN}$ from the solution of problem GQP, and we will use this step $s_{LF}$ in the dogleg to play the role of the Newton step. If $\|s_{LF}\| \leq .8\Delta_c$, then $s_{DL} = s_{LF}$. If $s_{LF}$ lies outside of the $.8$ trust region, the dogleg step is of the form

$$s_{DL} = s_{CP} + \alpha s_{LF} \text{ such that } \|s_{DL}\| = .8\Delta_c.$$  \hspace{1cm} (4.7)

From application of the standard dogleg analysis to the function $\|\nabla h_c^T s + h_c\|$, we know that $\|\nabla h_c^T s + h_c\|$ decreases as we move along $s_{DL}$ given in (4.7) from $s = 0$ to $s = s_{LF}$. The calculation of the dogleg can be summarized as follows.
Calculating the Dogleg:

If \((\|s_{LF}\| \leq 0.8\Delta_c)\), then

\[ s_{DL} = s_{LF} \]

Else

Calculate the Cauchy point from (4.6).

\[ s_{DL} = s_{CP} + \alpha s_{LF} \text{ such that } \|s_{DL}\| = 0.8\Delta_c \]

End if

The details on how to calculate \(\alpha\) such that \(\|s_{CP} + \alpha s_{LF}\| = 0.8\Delta_c\) can be found in Dennis and Schnabel [1983].

Now that we have found the step which will determine the required linear feasibility, all that remains is to calculate \(\theta_c\) by

\[ \theta_c = \|\nabla h_c^T s_{DL} + h_c - \Theta_{MIN}\|, \tag{4.8} \]

where the inclusion of \(\Theta_{MIN}\) simply translates the constraint so that the minimum value of \(\|\nabla h_c^T s + h_c - \Theta_{MIN}\|\) is zero.

Now we will consider the choice of the two-dimensional subspace. As indicated previously, the first direction we use will be \(s_{QP}\) if it exists. If the SQP step does not exist, then we will have determined a direction of negative or zero curvature inside the null space of \(\nabla h_c^T\), and we will use this direction \(d_{QP}\) as the first direction.

For the second direction we will use the step \(s_{DL}\) that determined the required linear feasibility. This choice will ensure that the intersection of the two-dimensional subspace with the feasibility region given in (4.5) is non-empty. Thus, the two-dimensional subspace will be

\[ s \in \text{span}\{s_{QP} \text{ or } d_{QP}; s_{DL}\}. \tag{4.9} \]

4.3 Using the Vardi subproblem in two more special cases

In this section we will discuss two more special situations that lead us to solve a Vardi subproblem (2.1). The first situation we consider is when the value for \(\theta_c\) from (4.8) is small, and the linear feasibility constraint becomes

\[ \|\nabla h_c^T s + h_c - \Theta_{MIN}\| \approx 0. \]
This is essentially a translated version of the special case discussed Section 4.1, and again, we would prefer a constraint of the form

$$\nabla h_c^T s + h_c - \Theta_{MIN} = 0$$

(4.10)

for numerical conditioning. From the definition of $\Theta_{MIN}$ in (3.11), we have

$$\nabla h_c^T s + h_c = \Theta_{MIN} = \nabla h_c^T s_{LF} + h_c$$

(4.11)

or

$$\nabla h_c^T s = \nabla h_c^T s_{LF}.$$  

(4.12)

This gives us another special case of the Vardi subproblem of the form

$$\begin{align*}
\text{minimize} & \quad \nabla x_1^c s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \nabla h_c^T s = \nabla h_c^T s_{LF} \\
& \quad \|s\| \leq \Delta_c.
\end{align*}$$

(4.13)

The other special case occurs when $s_{LF}$ is inside the trust region of radius $0.8\Delta_c$. In this situation, the linearized constraint manifold $\nabla h_c^T s + h_c = \Theta_{MIN}$ goes through the $0.8$ trust region, and again, we know how to choose the constant $\Theta_c$ in the Vardi subproblem. Namely, $\Theta_c = \nabla h_c^T s + h_c = \Theta_{MIN}$, and this gives us the special case of the Vardi subproblem given in (4.13).

To solve this subproblem, we can write

$$s = Q_1 w_1 + Q_2 w_2$$

(4.14)

where $Q_1$ is an orthonormal basis for the column space of $\nabla h_c$ and $Q_2$ is a basis for the null space of $\nabla h_c^T$. From the definition of $s_{LF}$, given in (3.10), we know that $s_{LF}$ is of the form $Q_1 w_{LF}$ for some $w_{LF} \in IR^n$. Substituting the representation given in (4.14) into (4.12) shows that the solution will be of the form $s = s_{LF} + Q_2 w_2$. With the change of variables $s = Q_2 w_2$, this subproblem becomes a standard unconstrained trust region subproblem of the form

$$\begin{align*}
\text{minimize} & \quad \hat{q}(w_2) \\
\text{subject to} & \quad \|w_2\| \leq \Delta_c,
\end{align*}$$

which can be solved with existing software.
4.4 Solution of constrained trust region subproblem 2DCTR

Now that we have specified the two-dimensional subspace and the amount of linear feasibility that we will require, in the form of $\theta_c$, we are ready to solve the 2DCTR subproblem. Recall that it is

2DCTR Subproblem:

\[
\begin{align*}
\text{minimize} & \quad \nabla x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|\nabla h_c^T s + h_c\|_2 \leq \theta_c \\
& \quad \|s\|_2 \leq \Delta_c \\
& \quad s \in \text{span}\{s_{QP}, d_{QP}, s_{DL}\}.
\end{align*}
\]

This subproblem consists of the minimization of a non-convex quadratic subject to two quadratic constraints in two dimensions. Recently, Dennis, Martinez, Tapia and Williamson [1990] gave a characterization of the solution of the constrained trust region subproblem CDT. Since we will use this characterization as the basis for our algorithm, we will state their result.

**Theorem 4.1** Dennis, Martinez, Tapia, and Williamson [1990].

If $s_c$ is a global solution of the CDT subproblem,

Problem CDT:

\[
\begin{align*}
\text{minimize} & \quad \nabla x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|\nabla h_c^T s + h_c\| \leq \theta_c \\
& \quad \|s\| \leq \Delta_c,
\end{align*}
\]

then either both constraints are binding, $\|\nabla h_c^T s + h_c\| = \theta_c$ and $\|s\| = \Delta_c$, or $s_c$ is a local solution of at least one of the two problems:

Subproblem TR:

\[
\begin{align*}
\text{minimize} & \quad \nabla x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|s\| \leq \Delta_c,
\end{align*}
\]

or

Subproblem LF:

\[
\begin{align*}
\text{minimize} & \quad \nabla x l_c^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|\nabla h_c^T s + h_c\| \leq \theta_c.
\end{align*}
\]
If any global solution of either (4.17) or (4.18) is feasible for both constraints, then it is a global solution of problem CDT.

Theorem 4.1 will be our guide in developing an algorithm to solve the 2DCTR subproblem. Theorem 4.1 clearly will hold for our subproblem 2DCTR since it is a two-dimensional version of the CDT subproblem. This characterization tells us that to find the global solution to the 2DCTR subproblem, we must be prepared to calculate all of the local solutions to the subproblems TR (4.17) and LF (4.18). The subproblem TR (4.17) is obviously the standard unconstrained trust region subproblem. The subproblem LF (4.18) can be transformed into the standard unconstrained trust region subproblem by transforming the elliptical constraint into a spherical constraint. Algorithms for approximating the global solution of the unconstrained trust region subproblem have been well-established. See, for example, Dennis and Schnabel [1983]. However, Theorem 4.1 tells us that the global solution to problem 2DCTR may be a local, non-global solution to one of the unconstrained subproblems TR (4.17) and LF (4.18). We have developed an algorithm to obtain all of the global solutions and the non-global solution, if it exists, to the unconstrained trust region subproblem of the form (4.17), and this work will be described in Chapter 5. For now, we will assume that we can obtain all of the solutions to the subproblems (4.17) and (4.18).

Using Theorem 4.1 as a guide, we give the following rough outline for the solution of the 2DCTR subproblem.

**Outline of the Solution to the 2DCTR Subproblem:**

1. Find all local solutions to subproblem TR given in (4.17).

2. If any global solution to subproblem TR satisfies \( \| \nabla h^T s + h \| \leq \theta_c \), then it is the solution to problem 2DCTR.

3. Find all local solutions to subproblem LF given in (4.18).

4. If any global solution to subproblem LF is inside the trust region, then it is the solution to problem 2DCTR.

5. Determine the points where both constraints are binding.

6. The solution to problem 2DCTR is the point with the smallest value of the quadratic model among:
(a) The points where both constraints are binding.

(b) The non-global solution to the subproblem in (4.17), if it exists and satisfies \( \|\nabla h^T s + h\| \leq \theta_c \).

(c) The non-global solution to the subproblem in (4.18), if it exists and is inside the trust region.

When we have a direction of negative curvature inside the null space of \( \nabla h_c^T \), (or a direction of zero curvature which is a descent direction for the quadratic model), the algorithm will simplify since the trust region constraint must be binding at the solution to the subproblem. To understand this point, consider a step \( \hat{s} \) in the two-dimensional subspace which satisfies the constraint \( \|\nabla h^T s + h\| \leq \theta_c \) and is strictly inside the trust region, \( \|\hat{s}\| < \Delta_c \). Now consider taking a step of the form \( \hat{s} + \alpha d_{QP} \) to the boundary of the trust region, and remember that in this case, \( d_{QP} \) is one of the directions that defines the two-dimensional subspace. The quadratic model for this step is

\[
q(\hat{s} + \alpha d_{QP}) = q(\hat{s}) + \alpha(\nabla s^T d_{QP} + \hat{s}^T B_{QP}) + \frac{1}{2} \alpha^2 d_{QP}^T B_{QP} d_{QP}.
\]

If we choose the sign of \( \alpha \) such that \( \alpha(\nabla s^T d_{QP} + \hat{s}^T B_{QP}) \leq 0 \), then

\[
q(\hat{s} + \alpha d_{QP}) \leq q(\hat{s}),
\]

which shows that the trust region constraint must be binding when \( d_{QP} \) is a direction of negative (or zero curvature which is a descent direction) inside the null space of \( \nabla h_c^T \). In the case of a direction of zero curvature which is not a descent direction for the quadratic model, we have \( q(\hat{s} + \alpha d_{QP}) = q(\hat{s}) \), and so any step of the form \( \hat{s} + \alpha d_{QP} \) that satisfies the trust region constraint will be a solution, including the points on the boundary of the trust region.

Once we know that the trust region constraint is binding, the algorithm will simplify because we do not have to solve subproblem LF. To show this fact, suppose that it is not true. Suppose that a global solution to subproblem 2DCTR \( \hat{s} \) is a solution of problem LF which lies on the boundary of the trust region, but lies strictly inside the linear feasibility region, \( \|\nabla h_c^T s + h_c\|_2 < \theta_c \). However, since it is a global solution of problem 2DCTR, it must have the smallest value of the quadratic model in the region

\[
q(\hat{s}) \leq q(s) \text{ for } \{s : \|s\| = \Delta_c \text{ and } \|\nabla h_c^T s + h_c\|_2 \leq \theta_c\} \quad (4.19)
\]
which includes the points where both constraints are binding. Since \(d_{QP}\) is a descent direction in this case, the trust region constraint is binding, and any solution to problem TR must lie on the boundary of the trust region. But, (4.19) shows that \(\hat{s}\) must be a solution to problem TR, which gives us the necessary contradiction.

We have described our solution procedure for subproblem 2DCTR, and a complete outline can be found in Section 4.7. In the next section, we give some details concerning the conversion of the subproblem to two dimensions. After problem TR has been converted to a standard two-dimensional unconstrained trust region subproblem, it can be solved by the techniques which will be given in Chapter 5. If a global solution to problem TR satisfies the linear feasibility constraint, then we take it as our trial step. Otherwise, if second-order sufficiency holds, we convert problem LF to the standard unconstrained trust region form and use the techniques of Chapter 5 to solve it. If a global solution to problem LF is inside the trust region, then it will be the trial step. Finally, all that remains is to find the points where both constraints are binding, and there can be at most four such points. In two dimensions, this merely requires finding the roots of a fourth degree polynomial.

### 4.5 Conversion of Subproblem 2DCTR to Two dimensions

In this section we will discuss the conversion of the 2DCTR subproblem to two dimensions. The 2DCTR subproblem is

\[
\begin{align*}
\text{2DCTR Subproblem:} \\
\text{minimize} & \quad \nabla_x l_c^T s + \frac{1}{2}s^T B_c s \\
\text{subject to} & \quad \|\nabla h_c^T s + h_c\|_2 \leq \theta_c \\
& \quad \|s\|_2 \leq \Delta_c \\
& \quad s \in \text{span}\{s_{QP} \text{ or } d_{QP}, s_{DL}\}.
\end{align*}
\]

The first step is to orthonormalize the vectors defining the two-dimensional subspace to obtain

\[
\text{span (} s_{QP} \text{ or } d_{QP}; s_{DL} \text{)} = \text{span (} v_1, v_2 \text{)}.
\]

Let \(V\) denote the matrix whose columns are \([v_1 v_2]\). We point out that it is possible but unlikely that \(v_1\) and \(v_2\) are actually the same direction. If this occurs, then we will take a step in the direction \(s_{DL}\) to the boundary of the trust region.
Given the matrix $V$, we write the step as $s = Vz$ where $z \in \mathbb{R}^2$ will be our new variable in the two-dimensional subspace. Then, writing subproblem 2DCTR with the new variables yields

\[
\begin{align*}
\text{minimize} & \quad q_{2D}(z) \\
\text{subject to} & \quad \|\nabla h_{2D}^T z + h_c\|_2 \leq \theta_c \\
& \quad \|z\| \leq \Delta_c
\end{align*}
\]

where

\[
\begin{align*}
q_{2D}(z) &= \nabla x^T_{2D} z + \frac{1}{2} z^T B_{2D} z, \\
\nabla x^T_{2D} &= V^T \nabla x^T_c, \\
B_{2D} &= V^T B_c V, \\
\text{and} \quad \nabla h_{2D} &= V^T \nabla h_c.
\end{align*}
\]

Then, $\nabla x^T_{2D} \in \mathbb{R}^2$, $B_{2D} \in \mathbb{R}^{2 \times 2}$, and $\nabla h_{2D} \in \mathbb{R}^{2 \times m}$.

### 4.6 Conversion of Problem LF into standard trust region form

In this section, we will discuss the conversion of problem LF

\[
\begin{align*}
\text{minimize} & \quad \nabla x^T_c s + \frac{1}{2} s^T B_c s \\
\text{subject to} & \quad \|\nabla h^T_c s + h_c\|_2 \leq \theta_c \\
& \quad s \in \text{span}\{s_{QP}, s_{DL}\}
\end{align*}
\]

into the standard trust region form:

\[
\begin{align*}
\text{minimize} & \quad q_{LF}(y) \\
& \quad \|y\| \leq \theta_c.
\end{align*}
\]

Recall that if we need to solve this subproblem, then we know that $s_{QP}$ exists and satisfies $(-\nabla h^T_c s_{QP} = h_c - \Theta_{MIN})$. Using this relation, (4.25) becomes

\[
\|\nabla h^T_c (s - s_{QP})\| \leq \theta_c.
\]

Recall that our two-dimensional subspace is

\[
\text{span } \{s_{QP}, s_{DL}\} = \text{span } \{v_1, v_2\},
\]
where $V = [v_1 v_2]$. Then, we can find a vector $z_{QP}$ such that $V z_{QP} = s_{QP}$. The determination of $z_{QP}$ merely depends on the procedure we used to orthonormalize $\{s_{QP}; s_{DL}\}$ into $\{v_1, v_2\}$. Substituting $s_{QP} = V z_{QP}$, yields $\|\nabla h_{2D}^T (z - z_{QP})\| \leq \theta_c$ where $\nabla h_{2D}$ was defined in the previous section.

We now replace $(z - z_{QP})$ with $U y$ where $U \in \mathbb{IR}^{2 \times 2}, y \in \mathbb{IR}^2$, and $U$ is orthonormal, and $U$ is chosen such that the columns of $\nabla h_{2D}^T U$ are also orthonormal. Thus,

$$\|\nabla h_c^T s + h_c\| = \|\nabla h_{2D}^T U y\| = \|y\| \leq \theta_c.$$ 

All that remains is to transform the quadratic model into

$$q_{LF} = \nabla x l_{LF}^T y + \frac{1}{2} y^T B_{LF} y$$

where

$$\nabla x l_{LF} = (\nabla x l_{2D}^T U + z_{QP} B_{2D} U)^T \quad (4.26)$$

and

$$B_{LF} = U^T B_{2D} U. \quad (4.27)$$

Now problem LF has been transformed into a standard unconstrained trust region subproblem. We remark, however, that if the columns of $\nabla h_{2D}$, which are $v_1^T \nabla h_c$ and $v_2^T \nabla h_c$, are not linearly independent, the feasible region determined by (4.25) will not be a solid ellipse but instead will be the region between two parallel lines.
4.7 Statement of the Algorithm

In this section, we summarize our strategy for calculating a trial step.

Algorithm Trialstep:

0. **Given** $h_c, \nabla h_c$, the quadratic model, $q_c(s) = \nabla_x l_c^T s + \frac{1}{2} s^T B_c s$, and the trust region radius, $\Delta_c$, calculate a trial step $s_c$.

1. **If** $(\nabla h_c h_c = 0)$, **then**
   
   Solve Problem VARDI: minimize $q_c(s)$
   subject to $\nabla h_c^T s = 0$
   $\|s\| \leq \Delta_c$

   for $s_c$.

   Return

   **End if.**

2. Solve Problem QP: minimize $q_c(s)$
   subject to $\nabla h_c^T s + h_c = \Theta_{MIN}$

   **If** (the solution to the QP exists), **then**
   
   $s_{QP}$ is the solution to the QP
   $\Delta \lambda_{QP}$ is the associated Lagrange multiplier step
   $\text{negcurv} = \text{false}$

   Else
   
   $d_{QP}$ is a direction of zero or negative curvature inside the null space of $\nabla h_c^T$
   $s_{LF}$ is a step to the linearized constraints, $\nabla h_c^T s + h_c = \Theta_{MIN}$
   $\text{negcurv} = \text{true}$

   **End if.**

3. **If** ($(\text{negcurv} = \text{false})$ and $(\|s_{QP}\| \leq \Delta_c))$, **then**
   
   $s_c = s_{QP}$

   Return

   **End if.**
4. Calculate the required Linear Feasibility.

\[
\text{If } (\|s_{LF}\| \leq 0.8\Delta_c), \text{ then } s_{DL} = s_{LF}
\]

\[
\text{Else}
\]

Calculate the Cauchy point for the constraints, \(s_{CP}\)

Find \(\alpha\) such that \(\|s_{CP} + \alpha s_{LF}\| = 0.8\Delta_c\)

\[
s_{DL} = s_{CP} + \alpha s_{LF}
\]

\[
\text{End if}
\]

\[
\theta_c = \|\nabla h_c^T s_{DL} + h_c - \Theta_{MIN}\|
\]

5. \textbf{If} (\(\theta_c\) is too small), \textbf{then}

Solve \textbf{Problem VARDI}: minimize \(q_c(s)\)

subject to \(\nabla h_c^T s = \nabla h_c^T s_{LF}\)

\[
\|s\| \leq \Delta_c
\]

for \(s_c\)

Return

\text{End if}

6. Choose the 2D Subspace.

\textbf{If} (\text{negcurv} = \text{false}), \textbf{then}

\(\{v_1, v_2\}\) spans \(\{s_{QP}, s_{DL}\}\)

\textbf{Else}

\(\{v_1, v_2\}\) spans \(\{d_{QP}, s_{DL}\}\)

\text{End if.}

7. Solve \textbf{Problem TR}: minimize \(q_c(s)\)

subject to \(\|s\| \leq \Delta_c\)

\[
s \in \text{span}\{v_1, v_2\}
\]

for all global solutions, \(s_{TRG}\), and the local, non-global solution, \(s_{TRL}\), if it exists.
8. If ( any $s_{TRG}$ satisfies $\|\nabla h_c^T s_{TRG} + h_c - \Theta_{MIN}\| \leq \theta_c$ ), then

\[ s_c = s_{TRG} \]

Return

End if.

9. If ( negcurv = false ), then

Solve Problem LF: minimize $q_c(s)$
subject to $\|\nabla h^T s + h - \Theta_{MIN}\| \leq \theta_c$
\[ s \in \text{span}\{v_1, v_2\} \]

for all global solutions, $s_{LFG}$, and the local, non-global solution, $s_{LFL}$, if it exists.

If ( any $s_{LFG}$ satisfies $\|s_{LFG}\| \leq \Delta_c$ ), then

\[ s_c = s_{LFG} \]

Return

End if.

10. Calculate the points where both of the 2D constraints are binding, i.e.,

$s_{CB1}, s_{CB2}, s_{CB3}, s_{CB4}$ such that
\[ \|\nabla h_c^T s + h_c\| = \theta_c \]
\[ \|s\| = \Delta_c \]
\[ s \in \text{span}\{v_1, v_2\} \]

There may be 2, 3 or 4 intersection points.

11. Determine if problem TR or problem LF has a local, non-global solution which satisfies the remaining constraint.

If ( ( $s_{TRL}$ exists ) and ( $\|\nabla h^T s_{TRL} + h - \Theta_{MIN}\| \leq \theta_c$ ) ), then

save $s_{TRL}$

End if.

If ( ( $s_{LFL}$ exists ) and ( $\|s_{LFL}\| \leq \Delta$ ) ), then

save $s_{LFL}$

End if.
12. \( s_c = \text{argmin}\{ q(s_{CB1}); \, q(s_{CB2}); \, q(s_{CB3}); \, q(s_{CB4}); \\
q(s_{TRL}), \text{if } s_{TRL} \text{ was saved}; \\
q(s_{LFL}), \text{if } s_{LFL} \text{ was saved} \}. \)

Return

End.
Chapter 5

Solution of the Unconstrained Trust Region Subproblem Restricted to Two Dimensions

Our goal, to develop a nonlinear programming algorithm, requires us to find an algorithm to solve the two-dimensional constrained trust region subproblem 2DCTR. As we have seen, the characterization of the solution to problem 2DCTR, (Theorem 4.1 restricted to two dimensions), tells us that the solution may be any local solution to the standard unconstrained trust region (UTR) subproblem. The unconstrained trust region subproblem minimizes a quadratic model of the objective function subject to a trust region constraint on the length of the step and is of the form

\[
\text{Problem UTR:} \quad \begin{align*}
\text{minimize} \quad & g^T s + \frac{1}{2} s^T H s \\
\text{subject to} \quad & \|s\| \leq \Delta,
\end{align*}
\]

where the Hessian \( H \) is assumed to be symmetric and the trust region radius is assumed to satisfy \( \Delta > 0 \). We use the expression local minimizer to refer to a point that has the smallest function value in an open neighborhood intersecting the feasible region. By global minimizer, we mean a point in the feasible set where the objective function takes on the absolute lowest value. Clearly, all global minimizers are also local minimizers. In addition, we will refer to local minimizers which are not global minimizers as non-global minimizers. Theorem 4.1 requires us to distinguish between global solutions and non-global solutions to problem UTR. When we get to the point where we are ready to determine if any local solution to the subproblems of the form given in (5.1) is the solution to problem 2DCTR, we must treat the global solutions and non-global solutions differently.

This chapter is concerned with finding all of the possible global solutions and the non-global solution, if it exists, to the standard unconstrained trust region subproblem. This is a daunting task, but recall from Chapter 2 that we have restricted our constrained subproblem 2DCTR to a relevant two-dimensional subspace. The
original motivation for this restriction centered on the difficulty in minimizing the quadratic model over the intersection points of the two quadratic constraints, but in two dimensions this becomes easy. We shall see that the restriction to two dimensions also makes the problem of finding the local solutions to problem UTR analytically and computationally more feasible. Thus, we will assume the restriction to the two-dimensional subspace holds throughout the remainder of this chapter, i.e., \( g \in \mathbb{R}^2 \) and \( H \in \mathbb{R}^{2 \times 2} \).

Our approach to solving this problem will break down the analysis into several different cases based primarily on the eigen-decomposition of the Hessian \( H \). These cases can be summarized as follows.

1. \( g = 0 \).
2. \( \Lambda_1 = \Lambda_2 \).
3. \( v_1^T g = 0 \).
4. \( v_2^T g = 0 \).
5. \( g \neq 0, \Lambda_1 < \Lambda_2, v_1^T g \neq 0, \) and \( v_2^T g \neq 0 \).

We will attack each of these cases in this order. We will refer to cases 1, 2, 3 and 4 as degenerate cases, and we will show that for these degenerate cases, all of the global solutions to problem UTR and the non-global solution, if it exists, can be determined analytically. This fact is strongly dependent on the restriction to two dimensions and is the primary reason that finding all of the local solutions to problem UTR is computationally inexpensive. In the non-degenerate case, we will use a modified version of the algorithm given in Moré and Sorensen [1983] to determine the global solution to problem UTR. Then we will determine if a non-global solution exists, and if it does, we will again use a modified version of the algorithm given in Moré and Sorensen [1983] to find it.

### 5.1 Preliminaries

The unconstrained trust region subproblem UTR is the basis for trust region algorithms for unconstrained optimization. Algorithms for determining an approximation to a global solution of problem UTR have been well-established. (See Dennis and Schnabel [1983] for a survey of this area.) We are interested in finding not only a
global solution but all of the local solutions to problem UTR under the assumptions that \( g \in \mathbb{R}^2 \), \( H \in \mathbb{R}^{2 \times 2} \), \( H \) is symmetric, and \( \Delta > 0 \). As mentioned above, we will base our analysis on the eigen-decomposition of the Hessian. Since \( H \) is real and symmetric, we have the real Schur decomposition of \( H \)

\[
Q^T H Q = \Lambda \equiv \text{diag}(\Lambda_1, \Lambda_2)
\]

(5.2)

where \( Q \) is orthogonal and \( \Lambda_1 \leq \Lambda_2 \) are the real eigenvalues of \( H \). (See, for example, Golub and Van Loan [1983].) The columns of \( Q \) are the orthonormal eigenvectors of \( H \), and we will denote the eigenvector corresponding to \( \Lambda_1 \) by \( v_1 \) and the eigenvector corresponding to \( \Lambda_2 \) by \( v_2 \) where

\[
Q = [v_1 \ v_2].
\]

In addition, we use the notation

\[
c_1 = v_1^T g \quad \text{and} \quad c_2 = v_2^T g.
\]

We will now give the tools that we will need to characterize the solutions of the unconstrained trust region subproblem. Sorensen [1982] gives the following characterization of the global solutions to problem UTR, and similar results can be found in Gay [1981].

**Lemma 5.1** Sorensen [1982], Gay [1981].

If \( s^* \) is a (global) solution to problem UTR, then \( s^* \) is a solution to an equation of the form

\[
(H + \mu^* I)s^* = -g
\]

(5.3)

with \( \mu^* \geq 0 \), \( \mu^*(\|s^*\|^2 - \Delta^2) = 0 \) and \( (H + \mu^* I) \) positive semidefinite.

We have inserted the qualifier (global) into the statement of Lemma 5.1 for clarity since we must make the distinction between global and non-global solutions to problem UTR. We point out that the conclusion in Lemma 5.1 that \( (H + \mu^* I) \) is positive semidefinite at a solution depends strongly on the fact that \( s^* \) is a global minimizer. This can be seen in the proof of Lemma 5.1 in Sorensen [1982] where it is assumed that \( s^* \) has the lowest value of the quadratic model on the boundary of the trust region.

Since Lemma 5.1 only gives necessary conditions for a step to be the global solutions to problem UTR, we give another lemma from Sorensen [1982] stating sufficient conditions for a step to be global minimizer.
Lemma 5.2 Sorensen [1982].

Let $\mu$ and $s$ satisfy

$$(H + \mu I)s = -g \text{ with } (H + \mu I) \text{ positive semidefinite.} \quad (5.4)$$

(i) If $\mu = 0$ and $\|s\| \leq \Delta$, then $s$ solves problem UTR.

(ii) If $\|s\| = \Delta$, then $s$ solves $\psi(s) = \min \{\psi(w) : \|w\| = \Delta\}$ where $\psi(w) = g^Tw + \frac{1}{2}w^THw$.

(iii) If $\mu \geq 0$ and $\|s\| = \Delta$, then $s$ solves problem UTR.

If, in fact, $(H + \mu I)$ is positive definite, then $s$ is unique in each of the cases (i), (ii), and (iii).

Since Lemmas 5.1 and 5.2 only address global solutions, we will need another tool to find a characterization of the non-global solutions. We apply the standard second-order sufficiency theorem for general nonlinear programming, which can be found in Avriel [1976], to problem UTR to obtain the following theorem.

Theorem 5.1 Second-order Sufficiency.

If there exists a vector $\mu^*$ such that

$$(H + \mu^* I)s^* = -g \quad (5.5)$$

$$\|s^*\| \leq \Delta \quad (5.6)$$

$$\mu^*(\Delta - \|s^*\|) = 0 \quad (5.7)$$

$$\mu^* \geq 0, \quad (5.8)$$

and for every $z \neq 0$ satisfying

$$z^Ts^* \geq 0 \text{ if } \|s^*\| = \Delta \text{ and } \mu^* = 0 \quad (5.9)$$

$$z^Ts^* = 0 \text{ if } \|s^*\| = \Delta \text{ and } \mu^* > 0, \quad (5.10)$$

it follows that

$$z^T(H + \mu^* I)z > 0, \quad (5.11)$$

then $s^*$ is a strict local minimizer of problem UTR.
Using these tools, we derive analytical expressions for all of the local solutions to problem UTR in the four degenerate cases in the next section. The following section is concerned with finding the global solution in the non-degenerate situation. In particular, we develop a good initial guess for the iterative procedure that we will use. Finally, we derive conditions that will determine if a non-global solution exists in the non-degenerate case, and we discuss how to find it if it exists.

5.2 Characterization of the Solutions for the Degenerate Cases

As indicated at the beginning of this chapter, we will start our analysis with the case where \( g = 0 \). The following theorem gives the solutions to problem UTR in this situation based on the eigenvalue distribution of \( H \).

**Theorem 5.2** Solutions to Problem UTR when \( g = 0 \).

Given \( g \in \mathbb{R}^2 \), \( H \in \mathbb{R}^{2 \times 2} \) with \( H \) symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) be the eigenvalues and \( \{v_1, v_2\} \) be the corresponding orthonormal eigenvectors of \( H \).

If \( (g = 0) \) and \( (\Lambda_1 > 0) \), then problem UTR has one global solution \( s^* = 0 \) with multiplier \( \mu^* = 0 \).

If \( (g = 0) \) and \( (\Lambda_1 = 0) \wedge (\Lambda_2 > 0) \), then problem UTR has an infinite number of global solutions of the form \( s^* = \alpha v_1 \) for all \( \alpha \in [-\Delta, \Delta] \).

If \( (g = 0) \) and \( (\Lambda_1 = \Lambda_2 = 0) \), then any point in the trust region is a global minimizer for problem UTR, \( s^* = \{s : \|s\| \leq \Delta_c\} \).

If \( (g = 0) \) and \( (\Lambda_1 < 0) \wedge (\Lambda_1 < \Lambda_2) \), then problem UTR has two global solutions of the form \( s_1^* = \Delta v_1 \) and \( s_2^* = -\Delta v_1 \) with multiplier \( \mu^* = -\Lambda_1 \).

If \( (g = 0) \) and \( (\Lambda_1 = \Lambda_2 < 0) \), then any point on the boundary of the trust region is a global minimizer of problem UTR, \( s^* = \{s : \|s\| = \Delta_c\} \).

**Proof**

Since \( g = 0, (H + \mu^* I)s^* = -g \) has the form

\[
\begin{pmatrix}
\Lambda_1 + \mu^* & 0 \\
0 & \Lambda_2 + \mu^*
\end{pmatrix}
\begin{pmatrix}
v_1^T s^* \\
v_2^T s^*
\end{pmatrix} = 0.
\] (5.12)
1. \((\Lambda_1 > 0)\).

Since \(\Lambda_1 > 0\) and \(\Lambda_1 \leq \Lambda_2\), the only solution to (5.12) with \(\mu^* \geq 0\) is \(s^* = 0\). Complementarity then requires \(\mu^* = 0\), and \((H + \mu^* I)\) is positive definite. Thus, from Lemma 5.2, problem UTR has a single global solution \(s^* = 0\) with \(\mu^* = 0\).

2. \((\Lambda_1 = 0) \land (\Lambda_2 = 0)\).

In this case, (5.12) reduces to the two equations

\[
(\mu^*)v_1^T s^* = 0 \quad (5.13)
\]
\[
(\Lambda_2 + \mu^*)v_2^T s^* = 0. \quad (5.14)
\]

Since \((\Lambda_2 + \mu^*) > 0\) for all \(\mu^* \geq 0\), equation (5.14) is only satisfied if \(s^*\) is orthogonal to \(v_2\). Since \(v_1\) is orthogonal to \(v_2\) and the dimension of the space is only 2, \(s^*\) must be of the form \(av_1\) for some constant \(a\). Substituting \(s^* = av_1\) into equation (5.13) shows that \(\mu^* = 0\). With this \(\mu^*\), \((H + \mu^* I)\) is positive semidefinite. Then, Lemma 5.2(i) shows that problem UTR has an infinite number of global solutions of the form \(s^* = av_1\) for all \(a \in [-\Delta, \Delta]\) with \(\mu^* = 0\).

3. \((\Lambda_1 = \Lambda_2 = 0)\).

For this case, equation (5.12) reduces to

\[
(\mu^*)v_1^T s^* = 0
\]
\[
(\mu^*)v_2^T s^* = 0.
\]

First consider \(\mu^* = 0\). With this \(\mu^*\), \((H + \mu^* I)\) is positive semidefinite, and \((H + \mu^* I)s^* = -g\) is satisfied for all \(s\). Application of Lemma 5.2(i) gives an infinite number of global solutions of the form \(s^* = \{s : \|s\| \leq \Delta\}\).

Notice that since \((H + \mu^* I)\) is also negative semidefinite, every point in the trust region is also a global maximum. This is geometrically reasonable since the quadratic is completely flat over the entire space in this case.

4. \((\Lambda_1 < 0) \land (\Lambda_2 > 0)\).

Equation (5.12) is satisfied for \(\mu^* = -\Lambda_1\) and \(s^* = av_1\) for some constant \(a\), and \(\mu^* > 0\). Since \((\Lambda_2 - \Lambda_1) > 0\), \((H + \mu^* I)\) is positive semidefinite, and Lemma 5.2(iii) yields two global solutions \(s^* = \Delta v_1\) and \(s^* = -\Delta v_1\) with \(\mu^* = -\Lambda_1\).
Although equation (5.12) is satisfied with \( \mu = -\Lambda_2 \), this cannot correspond to a minimizer since \( \mu < 0 \).

Equation (5.12) is also satisfied with \( s = 0 \), and complementarity would require \( \mu = 0 \). Since \( (H + \mu I) \) is indefinite, Lemma 5.1 shows that \( s = 0 \) cannot be a global minimizer. It is not a local minimizer either. The quadratic model at \( s = 0 \) is \( q(0) = 0 \). Now consider a step of the form \( s = \varepsilon v_1 \) where \( \varepsilon \) is small so that \( \varepsilon v_1 \) is inside the trust region. The quadratic model for this step is \( q(\varepsilon v_1) = 0.5\varepsilon^2 \Lambda_1 \). Since \( \Lambda_1 < 0 \), \( q(\varepsilon v_1) < q(0) \) for all \( 0 < \varepsilon < \Delta \), and \( s = 0 \) cannot be a minimum. It is actually a saddlepoint.

5. \((\Lambda_1 < 0) \land (\Lambda_2 = 0)\).

Equation (5.12) reduces to

\[
(\Lambda_1 + \mu^*)v_1^T s^* = 0 \\
(\mu^*)v_2^T s^* = 0,
\]

and is satisfied by \( \mu^* = -\Lambda_1 > 0 \) and \( s^* = av_1 \) for some constant \( a \). \((H + \mu^* I)\) is positive semidefinite. Then, Lemma 5.2(iii) yields two global solutions of the form \( s^* = \Delta v_1 \) and \( s^* = -\Delta v_1 \).

Equation (5.12) is also satisfied by \( \mu = 0 \) and \( s = av_2 \) for some constant \( a \). \((H + \mu I)\) is negative semidefinite, and so \( s = \Delta v_2 \) and \( s = -\Delta v_2 \) are global maximizers.

Equation (5.12) is satisfied by \( s = 0 \). With \( \mu = 0 \), \((H + \mu I)\) is negative semidefinite, and this solution is also a global maximum.

6. \((\Lambda_1 < \Lambda_2 < 0)\).

Equation (5.12) is satisfied for \( \mu^* = -\Lambda_1 \) and \( s^* = av_1 \) for some constant \( a \), and \( \mu^* > 0 \). Since \( (\Lambda_2 - \Lambda_1) > 0 \), \((H + \mu^* I)\) is positive semidefinite, and Lemma 5.2(iii) yields two global solutions \( s^* = \Delta v_1 \) and \( s^* = -\Delta v_1 \) with \( \mu^* = -\Lambda_1 \).

Equation (5.12) is also satisfied for \( \mu = -\Lambda_2 > 0 \) and \( s = av_2 \) for some constant \( a \), and complementarity would allow \( a = \Delta \) and \( a = -\Delta \). Since \( \Lambda_1 - \Lambda_2 < 0 \), \((H + \mu I)\) is negative semidefinite. Lemma 5.2(ii) applied to the corresponding maximization problem shows that \( s = \Delta v_2 \) and \( s = -\Delta v_2 \) maximize the
quadratic model on the boundary of the trust region. Thus, they cannot be minimizers.

Again, \( s = 0 \) satisfies equation (5.12), and complementarity would require \( \mu = 0 \). Then, \((H + \mu I)\) is negative definite, and \( s = 0 \) is a global maximum.

7. \((\Lambda_1 = \Lambda_2 < 0)\).

Equation (5.12) is satisfied for \( \mu^* = -\Lambda_1 > 0 \) and any \( s \), and \((H + \mu^* I)\) is positive semidefinite. Application of Lemma 5.2(iii) yields an infinite number of global solutions of the form \( s^* = \{ s : \|s\| = \Delta \} \).

The only other step which satisfies (5.12) is \( s = 0 \). With \( \mu = 0 \), \((H + \mu I)\) is negative definite, and \( s = 0 \) is a global maximum.

Now that we have enumerated all the possible solutions when \( g = 0 \), we will assume \( g \neq 0 \) and consider the situation when \( H \) has two equal eigenvalues. Again, all possible solutions can be determined analytically, and they are given in the following theorem.

**Theorem 5.3** Solutions to Problem UTR when \((\Lambda_1 = \Lambda_2)\).

Given \( g \in \mathbb{R}^2 \), \( H \in \mathbb{R}^{2 \times 2} \) with \( H \) symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) be the eigenvalues and \( \{v_1, v_2\} \) be the corresponding orthonormal eigenvectors of \( H \). Assume that \( \|g\| \neq 0 \). Let

\[
\mu_+ = -\Lambda_1 + \frac{\|g\|}{\Delta}. \tag{5.15}
\]

If \((\Lambda_1 = \Lambda_2)\) and \((\mu_+ \geq 0)\) where \(\mu_+\) is given by (5.15), then problem UTR has one global solution of the form

\[
s^* = -\left(\frac{1}{\Lambda_1 + \mu^*}\right)g
\]

where the multiplier \(\mu^* = \mu_+\).

If \((\Lambda_1 = \Lambda_2)\) and \((\mu_+ < 0)\) where \(\mu_+\) is given by (5.15), then the Newton step is inside the trust region, and problem UTR has one global solution of the form \( s^* = -(1/\Lambda_1)g \) with \(\mu^* = 0\).
Proof In this case, \((H + \mu I)s = -g\) reduces to
\[
s = -\frac{g}{(\Lambda_1 + \mu)},
\]  
(5.16)
and this equation is well-defined since \(\mu = -\Lambda_1\) is not a solution to \((H + \mu I)s = -g\).

Complementarity requires the trust region constraint to be binding when \(\mu \neq 0\).

Thus, there are only two solutions to \((H + \mu I)s = -g\) satisfying \(\|s\| = \Delta\), and they are
\[
\mu_+ = -\Lambda_1 + \frac{\|g\|}{\Delta}, \quad \mu_- = -\Lambda_1 - \frac{\|g\|}{\Delta}.
\]  
(5.17)
(5.18)

First we will consider whether or not \(\mu_+\) corresponds to a minimizer.

1. \((\mu_+ \geq 0)\).

Since \(\mu_+ \geq 0\), \(\mu_+\) can correspond to a minimizer, and \(\mu^* = \mu_+\). From (5.16), the step must be of the form
\[
s^* = -\frac{\Delta}{\|g\|}g,
\]
and \((H + \mu^* I)\) is positive definite. Lemma 5.2 verifies that \(s^*\) is the unique global minimizer in this case.

2. \((\mu_+ < 0)\).

In this case, \(\mu_+\) cannot correspond to a minimizer. However, we can show that \(H\) is positive definite since \(\mu_+ < 0\) implies \(\Lambda_1 > 0\). With \(\mu^* = 0\), (5.16) becomes
\[
s^* = -\frac{1}{\Lambda_1}g,
\]
and this is the Newton step. From \((\mu_+ < 0)\), we can show that the Newton step is inside the trust region by
\[
\|s^*\| = \frac{\|g\|}{\Lambda_1} < \Delta.
\]

Thus, for this case, the Newton step is the unique global minimizer.

Now we will consider whether \(\mu_-\) can correspond to a minimizer, and we obviously need only to consider \(\mu_- \geq 0\). In this case, \(\mu_- < -\Lambda_1\), and this implies that \((\Lambda_1 + \mu_-) < 0\). Thus, for all \(\mu_- \geq 0\), \((H + \mu_- I)\) is negative definite, and \(\mu_-\) cannot correspond to a minimizer. \qed
We have given all of the possible solutions to problem UTR for the special cases \( g = 0 \) and \( \Lambda_1 = \Lambda_2 \). The next special case we shall consider occurs when \( g \) is orthogonal to the eigenvector corresponding to the smallest eigenvector. Note that this situation will include what Moré and Sorensen [1983] call the hard case.

**Theorem 5.4** Solutions to Problem UTR when \( (v_1^T g = 0) \).

Given \( g \in \mathbb{R}^2 \), \( H \in \mathbb{R}^{2 \times 2} \) with \( H \) symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) be the eigenvalues and \( \{v_1, v_2\} \) be the corresponding orthonormal eigenvectors of \( H \). Assume that \( \|g\| \neq 0 \) and \( \Lambda_1 \neq \Lambda_2 \). Let

\[
\mu_+ = -\Lambda_2 + \frac{|c_2|}{\Delta} \quad (5.19)
\]

If \( (v_1^T g = 0) \) and \( (\Lambda_1 > 0) \land (\mu_+ \leq 0) \) where \( \mu_+ \) is given by (5.19), then the Newton step is inside the trust region, and problem UTR has one global solution of the form \( s^* = -(c_2/\Lambda_2)v_2 \).

If \( (v_1^T g = 0) \) and \( (\Lambda_1 > 0) \land (\mu_+ > 0) \) where \( \mu_+ \) is given by (5.19), then problem UTR has one global solution of the form \( s^* = -\text{sign}(c_2)\Delta v_2 \) with multiplier \( \mu^* = \mu_+ \).

If \( (v_1^T g = 0) \) and \( (\Lambda_1 \leq 0) \land (\mu_+ > -\Lambda_1) \) where \( \mu_+ \) is given by (5.19), then problem UTR has one global solution of the form \( s^* = -\text{sign}(c_2)\Delta v_2 \) with multiplier \( \mu^* = \mu_+ \).

The following situations are referred to as the hard case in Moré and Sorensen [1983].

If \( (v_1^T g = 0) \) and \( (\Lambda_1 \leq 0) \land (\mu_+ = -\Lambda_1) \), where \( \mu_+ \) is given by (5.19), then problem UTR has one global solution of the form

\[
s^* = -\left(\frac{c_2}{\Lambda_2 - \Lambda_1}\right)v_2. \quad (5.20)
\]

If \( (v_1^T g = 0) \) and \( (\Lambda_1 < 0) \land (\mu_+ < -\Lambda_1) \) where \( \mu_+ \) is given by (5.19), then problem UTR has two global solutions of the form

\[
s^* = -\left(\frac{c_2}{\Lambda_2 - \Lambda_1}\right)v_2 + \tau v_1 \quad (5.21)
\]

and \( s^* = -\left(\frac{c_2}{\Lambda_2 - \Lambda_1}\right)v_2 - \tau v_1 \quad (5.22)\)
where
\[ \tau = \left( \Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2} \right)^{\frac{1}{2}}. \] (5.23)

If \( v_1^T g = 0 \) and \( (\Lambda_1 = 0) \wedge (\mu_+ < -\Lambda_1) \) where \( \mu_+ \) is given by (5.19), then problem UTR has an infinite number of global solutions of the form
\[ s^* = -\left( \frac{c_2}{\Lambda_2} \right) v_2 + (2\gamma - 1)\tau v_1 \text{ for all } \gamma \in [0, 1] \]
where \( \tau \) is given by equation (5.23).

**Proof** First we will show that \( \mu = -\Lambda_2 \) cannot correspond to a minimizer. For this case, \( (H + \mu I)s = -g \) reduces to
\[
\begin{align*}
(\Lambda_1 + \mu)v_1^T s^* &= 0 \quad (5.24) \\
(\Lambda_2 + \mu)v_2^T s^* &= -v_2^T g. \quad (5.25)
\end{align*}
\]
For \( \mu = -\Lambda_2 \), there is no finite \( s \) such that equation (5.25) can be satisfied since \( v_2^T g \neq 0 \).

For \( \mu \neq -\Lambda_2 \), equations (5.24) and (5.25) require
\[ s = -\left( \frac{v_2^T g}{\Lambda_2 + \mu} \right) v_2, \] (5.26)
since \( v_1 \) is orthogonal to \( g \). Unless \( \mu^* = 0 \), complementarity requires \( \|s^*\| = \Delta \). There are only two choices of \( \mu \) that satisfy the complementarity condition and (5.26), and they are
\[
\begin{align*}
\mu_+ &= -\Lambda_2 + \frac{|c_2|}{\Delta}, \\
\mu_- &= -\Lambda_2 - \frac{|c_2|}{\Delta}. \quad (5.27) \quad (5.28)
\end{align*}
\]
First we will show that \( \mu_- \) cannot correspond to a minimizer. For \( \mu_- \),
\[
(H + \mu_- I) = Q \begin{pmatrix} \Lambda_1 - \Lambda_2 - \frac{|c_2|}{\Delta} & 0 \\ 0 & -\frac{|c_2|}{\Delta} \end{pmatrix} Q^T. \quad (5.29)
\]
Since \( \Lambda_1 < \Lambda_2 \), \( (H + \mu_- I) \) is negative definite, and \( \mu_- \) could only correspond to maximizers.

Now consider \( \mu_+ \).
1. \((\Lambda_1 > 0)\).

(a) \((\mu_+ \leq 0)\).

In this situation, the Newton step, \(s^* = -(c_2/\Lambda_2)v_2\), is in the trust region, and \(\mu^* = 0\). Since \(\mu_+ \leq 0\), we know that \(|c_2| \leq \Lambda_2 \Delta\), and we can show that \(s^*\) is in the trust region by

\[
||s^*|| = \frac{c_2^2}{\Lambda_2^2} \leq \Delta^2.
\]

With \(\mu^* = 0\) and \(\Lambda_1 > 0\), \((H + \mu^* I)\) is positive definite, and the Newton step is the unique global minimizer.

(b) \((\mu_+ > 0)\).

Since \(\Lambda_1 > 0\) and \(\mu_+ > 0\), \(\Lambda_1 + \mu_+ > 0\), and \((H + \mu_+ I)\) is positive definite. Substituting \(\mu^* = \mu_+\) into equation (5.26), we have

\[
s^* = -\text{sign}(c_2)\Delta v_2
\]

where \(\text{sign}(c_2) = 1\) if \(c_2 \geq 0\) and \(\text{sign}(c_2) = -1\) if \(c_2 < 0\). Since \((H + \mu^* I)\) is positive definite, \(s^*\) is the unique global minimizer.

2. \((\Lambda_1 \leq 0)\).

(a) \((\mu_+ > -\Lambda_1)\).

In this case, \(\Lambda_1 \leq 0\) implies \(\mu_+ > 0\). With \(\mu_+, (\mu_+ > -\Lambda_1)\) ensures that \((H + \mu_+ I)\) is positive definite. Thus, as in Case 1b, \(s^* = -\text{sign}(c_2)\Delta v_2\) is the unique global minimizer.

(b) \((\mu_+ \leq -\Lambda_1)\).

This situation is referred to in Moré and Sorensen [1983] as the hard case. It has the characteristic difficulty that \(||s(\mu)|| < \Delta\) for \(s(\mu)\) satisfying \((H + \mu I)s = -g\) when \((H + \mu I)\) is positive definite. Moré and Sorensen [1983] prove that solutions to problem UTR are of the form \(s^* = p + \tau v_1\) where

\[
(H - \Lambda_1 I)p = -g
\]

and \(\tau\) is chosen so that \(||p + \tau v_1|| = \Delta\). Notice that the resulting solutions still satisfy

\[
(H - \Lambda_1 I)(p + \tau v_1) = -g.
\]
Equation (5.30) reduces to \((\Lambda_2 - \Lambda_1)v_1^Ts = -v_1^Tg\), and this gives
\[
p = -\frac{c_2}{\Lambda_2 - \Lambda_1}v_2.
\] (5.31)

There are two values of \(\tau\) that satisfy \(\|p + \tau v_1\| = \Delta\), and they are
\[
\tau_+ = \left(\Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2}\right)^{\frac{1}{2}},
\] (5.32)
\[
\tau_- = -\left(\Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2}\right)^{\frac{1}{2}}.
\] (5.33)

Notice that \(\mu_+ \leq -\Lambda_1\) implies
\[
\Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2} \geq 0,
\]
and so the values for \(\tau\) given in (5.32) and (5.33) are well-defined.

i. \((\mu_+ = -\Lambda_1)\).
In this case, \(\tau_+ = \tau_- = 0\), and \(s^* = p\), which extends to the boundary of the trust region, is the single global minimizer.

ii. \((\Lambda_1 < 0)\).
In this case, there are two global solutions
\[
s^* = p + \tau_+ v_1 \text{ and } s^* = p + \tau_- v_1
\]
where \(p\) is given by (5.31) and \(\tau_+\) and \(\tau_-\) are given by (5.32) and (5.33).

We can show that they have the same value of the quadratic since
\[
q(p + \tau v_1) = av_2^Tg + \frac{1}{2}(\Lambda_1 \tau^2 + \Lambda_2 a^2) \text{ with } a = -\frac{c_2}{\Lambda_2 - \Lambda_1}
\]
does not depend on the sign of \(\tau\).

iii. \((\Lambda_1 = 0)\).
In this case, the quadratic reduces to
\[
q(p + \tau v_1) = av_2^Tg + \frac{1}{2}\Lambda_2 a^2;
\]
and it does not depend on \(\tau\) at all. Thus, every step of the form \(p + \tau v_1\) for all \(\tau\) has the same value of the quadratic model. Also, in this case, \(\mu^* = -\Lambda_1 = 0\) implies that the trust region radius is no
longer binding, and any step of the form \( p + \tau v_1 \) which lies in the trust region is a global solution. So we have an infinite number of global solutions which can be written as

\[
s^* = p + (2\gamma - 1)\tau_+ v_1 \text{ for all } \gamma \in [0, 1]
\]

where \( p \) is given by (5.31) and \( \tau_+ \) is given by (5.32).

The last special case we shall consider is when \( g \) is orthogonal to the eigenvector corresponding to the largest eigenvalue. The following theorem gives analytical expressions for the possible solutions in this situation.

**Theorem 5.5** Solutions to Problem UTR when \( (v_1^T g = 0) \).

Given \( g \in \mathbb{R}^2 \), \( H \in \mathbb{R}^{2 \times 2} \) with \( H \) symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) be the eigenvalues and \( \{v_1, v_2\} \) be the corresponding orthonormal eigenvectors of \( H \). Let \( c_1 = v_1^T g \) and \( c_2 = v_2^T g \). Assume that \( \|g\| \neq 0 \), \( \Lambda_1 \neq \Lambda_2 \) and \( v_1^T g \neq 0 \). Let

\[
\mu_- = -\Lambda_1 - \frac{|c_1|}{\Delta} \quad \text{and} \quad \mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta}.
\]

(5.34)

If \( (v_1^T g = 0) \) and \( (\Lambda_1 \geq 0) \land (\mu_+ \leq 0) \), then the Newton step is inside the trust region and problem UTR has one global solution of the form \( s^* = -(1/\Lambda_1)g \).

If \( (v_1^T g = 0) \) and \( (\mu_+ > 0) \) where \( \mu_+ \) is given by (5.34), then problem UTR has one global solution of the form \( s^* = \text{sign}(c_1)\Delta v_1 \) with multiplier \( \mu^* = \mu_+ \) given in (5.34).

If \( (v_1^T g = 0) \) and \( (\Lambda_1 < 0) \land (\mu_- > 0) \land (\mu_- > -\Lambda_2) \) where \( \mu_- \) is given in (5.34), then problem UTR has a non-global minimizer of the form \( s^* = \text{sign}(c_1)\Delta v_1 \) with multiplier \( \mu^* = \mu_- \).

**Proof** First we will show that \( \mu = -\Lambda_1 \) cannot correspond to a minimizer. For this case, \((H + \mu I)s = -g\) reduces to

\[
(\Lambda_1 + \mu)v_1^T s^* = -v_1^T g
\]

(5.35)

\[
(\Lambda_2 + \mu)v_2^T s^* = 0.
\]

(5.36)
For \( \mu = -\Lambda_1 \), there is no finite \( s \) such that equation (5.35) can be satisfied since \( v_1^T g \neq 0 \).

For \( \mu \neq -\Lambda_1 \), equations (5.35) and (5.36) require

\[
s = -\left( \frac{v_1^T g}{\Lambda_1 + \mu} \right) v_1,
\]

(5.37)
since \( v_2 \) is orthogonal to \( g \). Unless \( \mu^* = 0 \), complementarity requires \( \|s^*\| = \Delta \). There are only two choices of \( \mu \) that satisfy the complementarity condition and (5.37), and they are

\[
\mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta},
\]

(5.38)

\[
\mu_- = -\Lambda_1 - \frac{|c_1|}{\Delta}.
\]

(5.39)

Now we will consider if \( \mu_+ \) and \( \mu_- \) correspond to minimizers.

1. \( (\Lambda_1 \geq 0) \).

(a) \( (\mu_+ \leq 0) \).

Since \( g \neq 0 \) and \( v_2 \) is orthogonal to \( g \), we know that \( c_1 \neq 0 \). Since \( \Lambda_1 \geq 0 \), \( \mu_+ \leq 0 \) implies \( \Lambda_1 > 0 \) and

\[
\frac{|c_1|}{\Delta} \leq \Lambda_1.
\]

(5.40)

Thus, \( H \) is positive definite, and the Newton step is

\[
s^* = -\frac{1}{\Lambda_1} g
\]

with multiplier \( \mu^* = 0 \). We can write \( g = c_1 v_1 + c_2 v_2 \), and so \( \|g\| = c_1^2 + c_2^2 \).

In this case, \( \|g\| = c_1^2 \). We can now show that the Newton step is inside the trust region by

\[
\|s^*\|^2 = \frac{\|g\|^2}{\Lambda_1^2} = \frac{c_1^2}{\Lambda_1^2} \leq \Delta^2.
\]

Therefore, the Newton step is the unique global solution.

(b) \( (\mu_+ > 0) \).

Substituting \( \mu_+ \) into equation (5.37) yields

\[
s^* = -\frac{c_1}{|c_1|} \Delta v_1 = -\text{sign}(c_1) \Delta v_1
\]

with \( \mu^* = \mu_+ \). Since \( (H + \mu^* I) \) is positive definite, \( s^* \) corresponds to a global minimizer.
(c) Consider \(\mu_-\).
Since \(\Lambda_1 \geq 0\), \(\mu_- < 0\), and so \(\mu_-\) cannot correspond to a minimizer.

2. \((\Lambda_1 < 0)\).

(a) Consider \(\mu_+\).
Since \(\Lambda_1 < 0\), we know that \(\mu_+ > 0\). As in Case 1(b), \((H + \mu_+I)\) is positive definite, and

\[s^* = -\text{sign}(c_1)\Delta v_1\]

with \(\mu^* = \mu_+\) is the unique global minimizer.

(b) Consider \(\mu_-\).
Clearly, we are only interested in \(\mu_- \geq 0\). Using \(\mu_-\) in equation (5.37) yields

\[s = \text{sign}(c_1)\Delta v_1.\]

i. Suppose \(\mu_- > 0\).
Then, for Theorem 5.1, all \(z \neq 0\) satisfying \(z^T s = 0\) can be written as \(z = av_2\) for all \(a \neq 0\). Then,

\[z^T (H + \mu_-I)z = a^2(\Lambda_2 + \mu_-).\]  \hspace{1cm} (5.41)

Thus, from Theorem 5.1, when \(\mu_- > 0\) and \(\mu_- > -\Lambda_2\),

\[s^* = \text{sign}(c_1)\Delta v_1\]

with multiplier \(\mu^* = \mu_-\) corresponds to a strict local minimizer. From Lemma 5.1, this is not a global minimizer since \((H + \mu^*I)\) is indefinite.

ii. Suppose \(\mu_- = 0\).
If \(\Lambda_2 \leq 0\), then \((H + \mu_-I)\) with \(\mu_- = 0\) is negative semidefinite, and \(\mu_- = 0\) would correspond to a global maximizer.
Now consider \(\Lambda_2 > 0\). In this case, \(H\) is indefinite and geometrically \(\hat{s} = \text{sign}(c_1)\Delta v_1\) is a saddlepoint for the quadratic model. Thus, we will be able to show that \(\hat{s}\) is not a local minimizer by showing that the quadratic model decreases as we move inside the trust region along the direction \(v_1\) from \(\hat{s}\). The quadratic at \(\hat{s}\) is

\[q(\hat{s}) = |c_1|\Delta + \frac{1}{2}\Delta^2 \Lambda_1.\]  \hspace{1cm} (5.42)
Now consider a step \( \tilde{s} \) which is slightly inside the trust region from \( \hat{s} \) along \( v_1 \) of the form

\[
\tilde{s} = \text{sign}(c_1)(\Delta - \varepsilon)v_1
\]  

for some small \( \varepsilon > 0 \). Then, the quadratic evaluated at \( \tilde{s} \) is

\[
q(\tilde{s}) = |c_1|(\Delta - \varepsilon) + \frac{1}{2}(\Delta - \varepsilon)^2\Lambda_1. \tag{5.44}
\]

To show that \( q(s) \) decreases as we move inside the trust region, we must show that \( q(\tilde{s}) < q(\hat{s}) \). Subtracting \( q(\tilde{s}) \) from \( q(\hat{s}) \) yields

\[
q(\hat{s}) - q(\tilde{s}) = -|c_1|\varepsilon + \frac{1}{2}\Lambda_1 ((\Delta - \varepsilon)^2 - \Delta^2) \tag{5.45}
\]

\[
= -|c_1|\varepsilon + \Lambda_1 \Delta \varepsilon + \frac{1}{2}\Lambda_1 \varepsilon^2. \tag{5.46}
\]

But, since \( \mu_- = 0 \), we know that \( -\Lambda_1 \Delta = |c_1| \), and so,

\[
q(\hat{s}) - q(\tilde{s}) = \frac{1}{2}\Lambda_1 \varepsilon^2 < 0.
\]

Thus, \( \mu_- = 0 \) does not correspond to a local minimizer.

Thus, we have analytical expressions based on the eigen-decomposition of \( H \) for all of the possible global solutions and the non-global solution, if it exists, to problem UTR for the degenerate cases. The possibilities include a single global solution, two global solutions, a global solution and a non-global solution, and an infinite number of global solutions. When there is an infinite number of solutions, the shape of the solution set can be a line segment, the boundary of the trust region, or every point in the trust region.

### 5.3 Calculating the Global Solution in the Non-degenerate Case

In this section we will discuss how to find the global minimizer of problem UTR in the non-degenerate case. By non-degenerate we mean \( g \neq 0 \), \( \Lambda_1 \neq \Lambda_2 \), and \( g \) is not orthogonal to either eigenvector.
In this situation, we know there is a unique global minimizer with multiplier \( \mu^* \) in the interval \([-\Lambda_1, \infty)\). The solution is the Newton step if it is inside the trust region. Otherwise, it is the solution to

\[
(H + \mu I)s = -g \text{ such that } \|s\| = \Delta \text{ and } \mu \in (-\Lambda_1, \infty).
\] (5.47)

Using the eigen-decomposition of \( H \), we can write \( s \) as

\[
s(\mu) = -Q(\Lambda + \mu I)^{-1}Q^Tg.
\] (5.48)

With the notation \( c_1 = v_1^Tg \) and \( c_2 = v_2^Tg \), (5.48) becomes

\[
s(\mu) = -\left( \frac{c_1}{\Lambda_1 + \mu} \right)v_1 - \left( \frac{c_2}{\Lambda_2 + \mu} \right)v_2.
\] (5.49)

Adding the trust region constraint yields

\[
\|s(\mu)\|^2 = \frac{c_1^2}{(\Lambda_1 + \mu)^2} + \frac{c_2^2}{(\Lambda_2 + \mu)^2} = \Delta^2.
\] (5.50)

Note that \( s(\mu) \) is well-defined in the sense that there is no finite step satisfying \( (H + \mu I)s = -g \) for multipliers \( \mu = -\Lambda_1 \) and \( \mu = -\Lambda_2 \).

Moré and Sorensen [1983] give an effective algorithm for determining an approximation to a global solution of the n-dimensional trust region subproblem. Their algorithm is a safeguarded Newton’s method on the function

\[
\frac{1}{\Delta} - \frac{1}{\|(H + \mu I)^{-1}g\|_2} = 0.
\] (5.51)

Newton’s method is very efficient when applied to (5.51) since this nonlinear function is almost linear on \((-\Lambda_1, \infty)\), and the safeguarding strategy serves to confine the steps that Newton’s method takes to the interval of interest. We will use a simplified version of Moré and Sorensen’s algorithm to find an approximate global solution. Their algorithm has an additional level of complexity designed to detect a hard case solution of the form \( p + \tau v_1 \). We do not need this feature because the hard case occurs when \( g \) is orthogonal to \( v_1 \), and this is one of the special situations where we can calculate the solutions analytically. Since Moré and Sorensen’s algorithm is designed to solve the n-dimensional problem, they do not have the luxury of the eigen-decomposition of \( H \). In two dimensions, though, the eigen-decomposition of \( H \) is inexpensive, and we will use this information as much as possible.
We define \( \phi(\mu) \) as the following function

\[
\phi(\mu) = \frac{1}{\| (H + \mu I)^{-1} g \|_2} - \frac{1}{\Delta} = 0, \quad (5.52)
\]

and we will consider applying Newton’s method to it. Given a starting point \( \mu_0 \), the iterates that Newton’s method generates are of the form

\[
\mu_+ = \mu_c - (\phi'(\mu_c))^{-1} \phi(\mu_c).
\]

The linear system \((H + \mu_+ I)s = -g\) then determines \( s(\mu_+) \). As mentioned above, \( \phi(\mu) \) is almost linear on the interval \((-\Lambda_1, \infty)\), and the following lemma gives the slope of the line tangent to \( \phi(\mu) \) as \( \mu \to -\Lambda_1 \) from both the right and left sides. We will use this information to calculate an initial guess for Newton’s method.

**Lemma 5.3** Given \( g \in \mathbb{R}^2 \), \( H \in \mathbb{R}^{2 \times 2} \) where \( H \) is symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) denote the eigenvalues of \( H \), and let \( v_1 \) and \( v_2 \) denote the corresponding orthonormal eigenvectors. Let \( c_1 = v_1^T g \) and \( c_2 = v_2^T g \). Assume that \( g \neq 0 \), \( \Lambda_1 < \Lambda_2 \), and that \( g \) is not orthogonal to \( v_1 \) or \( v_2 \). Let

\[
\phi(\mu) = \frac{1}{\| (H + \mu I)^{-1} g \|_2} - \frac{1}{\Delta}. \quad (5.53)
\]

Then,

\[
\lim_{\mu \to -\Lambda_1^-} \phi'(\mu) = - \frac{1}{| c_1 |} \quad (5.54)
\]

and,

\[
\lim_{\mu \to -\Lambda_1^+} \phi'(\mu) = \frac{1}{| c_1 |}. \quad (5.55)
\]

**Proof** First,

\[
\phi(\mu) = \left( \frac{c_1^2}{(\Lambda_1 + \mu)^2} + \frac{c_2^2}{(\Lambda_2 + \mu)^2} \right)^{-\frac{1}{2}} - \frac{1}{\Delta}, \quad (5.56)
\]

and

\[
\phi'(\mu) = \left( \frac{c_1^2}{(\Lambda_1 + \mu)^3} + \frac{c_2^2}{(\Lambda_2 + \mu)^3} \right) \left( \frac{c_1^2}{(\Lambda_1 + \mu)^2} + \frac{c_2^2}{(\Lambda_2 + \mu)^2} \right)^{-\frac{3}{2}}. \quad (5.57)
\]

Since

\[
\lim_{\mu \to -\Lambda_1^-} \phi'(\mu) = \lim_{\varepsilon \to 0^-} \phi'(-\Lambda_1 + \varepsilon),
\]
we will consider $\phi'(-\Lambda_1 + \varepsilon)$.

$$\phi'(-\Lambda_1 + \varepsilon) = \left( \frac{c_1^2}{\varepsilon^3 + \left(\frac{c_1^2}{\varepsilon^2 - \frac{c_2^2}{\varepsilon^2 - \frac{c_1^2}{\varepsilon^2} + \frac{c_2^2}{\varepsilon^2}}\right)} \right) \left( \frac{c_1^2}{\frac{c_1^2}{\varepsilon^2} - \frac{c_2^2}{\varepsilon^2}} \right)^{-\frac{3}{2}}$$

$$= \left( \frac{c_1^2}{\varepsilon^3} \frac{(\Lambda_2 - \Lambda_1 + \varepsilon)^3 + c_2^2 \varepsilon^3}{(\Lambda_2 - \Lambda_1 + \varepsilon)^3} \right) \left( \frac{c_1^2}{\frac{c_1^2}{\varepsilon^2} - \frac{c_2^2}{\varepsilon^2} + c_2^2 \varepsilon^2} \right)^{-\frac{3}{2}}$$

$$= \text{sign}(\varepsilon) \text{sign}(\Lambda_2 - \Lambda_1 + \varepsilon) \left( \frac{c_1^2}{\varepsilon^3} \frac{(\Lambda_2 - \Lambda_1 + \varepsilon)^2 + c_2^2 \varepsilon^2}{(\Lambda_2 - \Lambda_1 + \varepsilon)^2 + c_2^2 \varepsilon^2} \right)^{\frac{3}{2}} \cdot (5.58)$$

Thus,

$$\lim_{\varepsilon \to 0^-} \phi'(-\Lambda_1 + \varepsilon) = (-1) \frac{c_1^2 (\Lambda_2 - \Lambda_1 + \varepsilon)}{c_1^2 (\Lambda_2 - \Lambda_1)^2}$$

and so,

$$\lim_{\varepsilon \to 0^-} \phi'(-\Lambda_1 + \varepsilon) = -\frac{1}{|c_1|}.$$ 

Similarly, from (5.58),

$$\lim_{\varepsilon \to 0^+} \phi'(-\Lambda_1 + \varepsilon) = \frac{c_1^2 (\Lambda_2 - \Lambda_1 + \varepsilon)}{c_1^2 (\Lambda_2 - \Lambda_1)^2}$$

and so,

$$\lim_{\varepsilon \to 0^+} \phi'(-\Lambda_1 + \varepsilon) = \frac{1}{|c_1|}.$$ 

The same type of relations can be shown as $\mu \to -\Lambda_2$, and they are

$$\lim_{\mu \to -\Lambda_2^-} \phi'(\mu) = -\frac{1}{|c_2|} \quad (5.59)$$

and,

$$\lim_{\mu \to -\Lambda_2^+} \phi'(\mu) = \frac{1}{|c_2|} \quad (5.60)$$

We now have the slope of $\phi(\mu)$ as $\mu \to -\Lambda_1$, which is also the slope of the function $\phi(\mu) = \|s(\mu)\|^{-1}$. The next lemma shows that the line tangent to $\phi(\mu)$ as $\mu \to -\Lambda_1$ from both the right and the left is always greater than or equal to $\phi(\mu)$.

**Lemma 5.4** Given $g \in IR^2$, $H \in IR^{2 \times 2}$ where $H$ is symmetric, and $\Delta > 0$, let $\Lambda_1 \leq \Lambda_2$ denote the eigenvalues of $H$, and let $v_1$ and $v_2$ denote
the corresponding orthonormal eigenvectors. Let

\[ \tilde{\phi}(\mu) = \frac{1}{\| (H + \mu I)^{-1}g \|_2}. \] (5.61)

Assume that \( g \neq 0, \Lambda_1 < \Lambda_2 \), and that \( g \) is not orthogonal to \( v_1 \) or \( v_2 \). Let \( l^{-}(\mu) \) denote the line tangent to \( \tilde{\phi}(\mu) \) as \( \mu \rightarrow -\Lambda_1^- \). Then,

\[ \tilde{\phi}(\mu) \leq l^{-}(\mu) \quad \text{for} \quad \mu \in (-\infty, -\Lambda_1]. \] (5.62)

Let \( l^{+}(\mu) \) denote the line tangent to \( \tilde{\phi}(\mu) \) as \( \mu \rightarrow -\Lambda_1^+ \). Then,

\[ \tilde{\phi}(\mu) \leq l^{+}(\mu) \quad \text{for} \quad \mu \in [-\Lambda_1, \infty). \] (5.63)

**Proof** The line tangent to \( \tilde{\phi}(\mu) \) as \( \mu \rightarrow -\Lambda_1^- \) is

\[ l^{-}(\mu) = -\frac{1}{|c_1|} (\mu + \Lambda_1). \] (5.64)

Consider \( \tilde{\phi}(\mu)^2 \):

\[
\tilde{\phi}(\mu)^2 = \left( \frac{c_1^2}{(\Lambda_1 + \mu)^2} + \frac{c_2^2}{(\Lambda_2 + \mu)^2} \right)^{-1} \\
= (\Lambda_1 + \mu)^2 \left( \frac{c_1^2(\Lambda_2 + \mu)^2 + c_2^2(\Lambda_1 + \mu)^2}{c_1^2(\Lambda_2 + \mu)^2 + c_2^2(\Lambda_1 + \mu)^2} \right) \\
= (\Lambda_1 + \mu)^2 \left( c_1^2 + c_2^2 \left( \frac{\Lambda_1 + \mu}{\Lambda_2 + \mu} \right)^2 \right)^{-1}.
\]

Thus,

\[ \tilde{\phi}(\mu)^2 \leq (\Lambda_1 + \mu)^2 \left( \frac{1}{c_1^2} \right), \]

and so,

\[ \tilde{\phi}(\mu) \leq \frac{|\Lambda_1 + \mu|}{|c_1|}. \] (5.65)

For \( \mu \in (-\infty, -\Lambda_1], |\Lambda_1 + \mu| = -(\Lambda_1 + \mu) \), and so

\[ \tilde{\phi}(\mu) \leq \frac{(\Lambda_1 + \mu)}{|c_1|}. \] (5.66)

Thus,

\[ \tilde{\phi}(\mu) \leq l^{-}(\mu) \]
for $\mu \in (-\infty, -\Lambda_1]$.

The line tangent to $\bar{\phi}(\mu)$ as $\mu \to -\Lambda_1^+$ is

$$l^+(\mu) = \frac{1}{|c_1|} (\mu + \Lambda_1).$$

(5.67)

For $\mu \in [-\Lambda_1, \infty), (\mu + \Lambda_1) \geq 0$, and from (5.65),

$$\bar{\phi}(\mu) \leq \frac{(\Lambda_1 + \mu)}{|c_1|}.$$

Thus,

$$\bar{\phi}(\mu) \leq l^+(\mu)$$

for $\mu \in [-\Lambda_1, \infty)$. □

The point at which the tangent line $l^+(\mu) = 1/\Delta$ is

$$\mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta}.$$ (5.68)

Let $\mu^*$ denote the solution to $\phi(\mu) = 0$, and recall that $\mu^*$ corresponds to the global solution to problem UTR. Since we know that $\mu^* \geq 0$, we will take

$$\mu_0 = \max\{0, \mu_+\}$$ (5.69)

as our starting point. From Lemma 5.4, we know that the tangent line $l^+(\mu) \geq \bar{\phi}(\mu)$, and this tells us that $\mu_+ \leq \mu^*$. Since $\mu^* \geq 0$, we have $\mu_0 \leq \mu^*$. Thus, Newton's method started from $\mu_0$ produces a monotonically increasing sequence converging to the solution of $\phi(\mu) = 0$. We point out that since we know the eigenvalues of $H$, and we know that our starting iterate is smaller than the solution, we do not need the safeguarding feature. The final ingredient we need is the stopping criteria for the algorithm. However, we need only test to see if either we have the Newton step,

$$\|s_c\| \leq \Delta \text{ and } \mu_c = 0,$$ (5.70)

or we have a step that is sufficiently close to the boundary of the trust region,

$$\| \Delta - \|s_c\| \| \leq \sigma \Delta$$ (5.71)

for some tolerance $\sigma$. Thus, using Newton's method from $\mu_0$ given in (5.69), we can find the global solution in the non-degenerate case.
5.4 Existence and Calculation of the Local Solution in the Non-degenerate Case

Now that we have found the global solution in the non-degenerate case, all that remains is to determine if there is a non-global solution and to find it, if it exists.

Clearly, \( \mu \in (-\infty, -\Lambda_2) \) cannot correspond to a local minimizer since \((H + \mu I)\) is negative definite. Recall that there is no finite \( s \) satisfying \((H + \mu I)s = -g\) for \( \mu = -\Lambda_2 \). This leaves the interval \((-\Lambda_2, -\Lambda_1)\) in which we will search for a local minimizer. Note that any local minimizer with \( \mu^* \) in this interval cannot be a global minimizer since \((H + \mu^* I)\) will not be positive semidefinite. If a local solution exists, it must satisfy

\[
(H + \mu I)s = -g \text{ such that } \|s\| = \Delta, \mu \in (-\Lambda_2, -\Lambda_1) \text{ and } \mu \geq 0. \tag{5.72}
\]

Obviously, if \( \Lambda_1 \geq 0 \), then there will not be a local solution.

Consider the function

\[
\varphi(\mu) = \|(H + \mu I)^{-1} g\|^2 \tag{5.73}
\]
on the interval \((-\Lambda_2, -\Lambda_1)\). Dennis, Martinez, Tapia and Williamson [1990] proved the following facts concerning a local solution to problem UTR.

1. \( \varphi(\mu) \) is strictly convex for \( \mu \in (-\Lambda_2, -\Lambda_1) \), and \( \lim_{\mu \to -\Lambda_1^-} \varphi(\mu) = \infty \).

2. The equation \( \varphi(\mu) = \Delta^2 \) has at most two roots in \((-\Lambda_2, -\Lambda_1)\).

3. If a non-global solution exists, it must be the largest root of \( \varphi(\mu) = \Delta^2 \) and satisfy \( \varphi'(\mu) > 0 \).

The following theorem gives necessary and sufficient conditions for the equation \( \varphi(\mu) = \Delta^2 \) to have roots in the interval \((-\Lambda_2, -\Lambda_1)\).

**Theorem 5.6** Given \( g \in IR^2 \), \( H \in IR^{2 \times 2} \) where \( H \) is symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) denote the eigenvalues of \( H \), and let \( v_1 \) and \( v_2 \) denote the corresponding eigenvectors. Let \( c_1 = v_1^T g \) and \( c_2 = v_2^T g \). Assume that \( g \neq 0 \), \( \Lambda_1 < \Lambda_2 \), and that \( g \) is not orthogonal to \( v_1 \) or \( v_2 \). Let

\[
\mu_0 = \frac{\alpha \Lambda_2 - \Lambda_1}{1 - \alpha} \text{ where } \alpha = -\left(\frac{c_2^2}{c_1^2}\right)^{\frac{1}{3}}. \tag{5.74}
\]
Then, the equation \( \|(H + \mu I)^{-1}g\|_2^2 = \Delta^2 \) has a solution on the interval \((-\Lambda_2, -\Lambda_1)\) if and only if
\[
\|(H + \mu_0 I)^{-1}g\|_2^2 \leq \Delta^2.
\]

**Proof** Expanding \( \varphi(\mu) \) gives
\[
\varphi(\mu) = \frac{c_1^2}{(\Lambda_1 + \mu)^2} + \frac{c_2^2}{(\Lambda_2 + \mu)^2}.
\]
Then, \( \varphi(\mu) \) will have a unique minimizer for \( \mu \in (-\Lambda_2, -\Lambda_1) \) since it is strictly convex on this interval. To find this minimum, we set \( \varphi'(\mu) = 0 \).
\[
\varphi'(\mu) = \frac{c_1^2}{(\Lambda_1 + \mu)^3} + \frac{c_2^2}{(\Lambda_2 + \mu)^3}.
\]
So, \( \varphi'(\mu) = 0 \) is equivalent to
\[
\frac{(\Lambda_1 + \mu)^3}{(\Lambda_2 + \mu)^3} = -\frac{c_1^2}{c_2^2}
\]
\[
\frac{(\Lambda_1 + \mu)}{(\Lambda_2 + \mu)} = -\left(\frac{c_1^2}{c_2^2}\right)^{\frac{1}{3}}.
\]
Let
\[
a = -\left(\frac{c_1^2}{c_2^2}\right)^{\frac{1}{3}}.
\]
Then, equation (5.76) becomes \( \Lambda_1 + \mu = a(\Lambda_2 + \mu) \), and it is easy to see that
\[
\mu_0 = \frac{a\Lambda_2 - \Lambda_1}{1 - a}
\]
where \( a \) is given by (5.77). Notice that \( \mu_0 \) given by (5.78) is well-defined since
\[
(1 - a) > 1.
\]
This follows from (5.77) and the fact that \( \Lambda_1 < \Lambda_2 \). Clearly \( \mu_0 \) minimizes \( \varphi(\mu) \) since \( \varphi''(\mu_0) > 0 \).

Thus, we have established that \( \mu_0 \) is the unique global minimizer of \( \varphi(\mu) \) for \( \mu \in (-\Lambda_2, -\Lambda_1) \), and consequently,
\[
\|(H + \mu_0 I)^{-1}g\|_2^2 < \|(H + \mu I)^{-1}g\|_2^2 \text{ for all } \mu \in (-\Lambda_2, -\Lambda_1) \text{ with } \mu \neq \mu_0.
\]
From this, it is obvious that \( \varphi(\mu) \) will intersect the horizontal line \( \Delta^2 \) if and only if

\[
\|(H + \mu_0 I)^{-1}g\|^2 \leq \Delta^2.
\]

\[ \square \]

The next theorem gives conditions that are necessary and sufficient for a local minimizer to exist in the interval \((-\Lambda_2, -\Lambda_1)\).

**Theorem 5.7**

Given \( g \in \mathbb{R}^2, H \in \mathbb{R}^{2 \times 2} \) where \( H \) is symmetric, and \( \Delta > 0 \), let \( \Lambda_1 \leq \Lambda_2 \) denote the eigenvalues of \( H \), and let \( v_1 \) and \( v_2 \) denote the corresponding orthonormal eigenvectors. Let \( c_1 = v_1^T g \) and \( c_2 = v_2^T g \). Assume that \( g \neq 0 \), \( \Lambda_1 < \Lambda_2 \), and that \( g \) is not orthogonal to \( v_1 \) or \( v_2 \). Let \( \mu_0 \) be given by (5.74), and let

\[
\mu_- = -\Lambda_1 - \frac{|c_1|}{\Delta} \quad \text{and} \quad \mu_l = \max(0, \mu_0). \tag{5.80}
\]

If \( (\mu_0 < 0) \), then problem UTR has a non-global solution on the interval \((-\Lambda_2, -\Lambda_1)\) if and only if

\[
\Lambda_1 < 0 \quad \text{and} \quad \|(H + \mu_l I)^{-1}g\|_2 = \|H^{-1}g\| \leq \Delta. \tag{5.81}
\]

If \( (\mu_0 \geq 0) \), then problem UTR has a local minimizer on the interval \((-\Lambda_2, -\Lambda_1)\) if and only if

\[
\Lambda_1 < 0 \quad \text{and} \quad \|(H + \mu_0 I)^{-1}g\|_2 < \Delta. \tag{5.82}
\]

Furthermore, the non-global solution, if it exists, is contained in the interval \( \mu^* \in [\mu_l, \mu_-] \).

**Proof**

Let \( \mu^* \) denote the multiplier contained in \((-\Lambda_2, -\Lambda_1)\) which corresponds to a non-global minimizer, if one exists.

1. **ONLY IF**: Show \( \mu^* \) exists implies either condition (5.81) or condition (5.82).

   From Theorem 5.6 and the fact that \( \mu_0 \) is not a minimizer, we know \( \mu^* \in (\mu_0, -\Lambda_1) \). Since \( \mu^* \) must be greater than or equal to zero, then \( \Lambda_1 < 0 \).
(a) \((\mu_0 < 0)\)

In this case, we have \(\mu_0 < 0 \leq \mu^* < -\Lambda_1\). Since \(\|(H + \mu I)^{-1}g\|^2\) is strictly increasing on the interval \([\mu_0, -\Lambda_1]\), and \(\|(H + \mu^* I)^{-1}g\|^2 = \Delta^2\), we have

\[
\|(H)^{-1}g\|^2 \leq \|(H + \mu^* I)^{-1}g\|^2 = \Delta^2
\]

which gives us the desired result.

(b) \((\mu_0 \geq 0)\).

In this case, we have \(0 \leq \mu_0 < \mu^* < -\Lambda_1\). Since \(\|(H + \mu I)^{-1}g\|^2\) is strictly increasing on the interval \([\mu_0, -\Lambda_1]\), and \(\|(H + \mu^* I)^{-1}g\|^2 = \Delta^2\), we have

\[
\|(H + \mu_0 I)^{-1}g\|^2 < \|(H + \mu^* I)^{-1}g\|^2 = \Delta^2
\]

which gives us the desired result.

2. IF: Show that conditions (5.81) and (5.82) imply \(\mu^*\) exists.

We know that a local solution must be a non-negative root of the equation

\[
\|(H + \mu I)^{-1}g\|^2 = \Delta^2. \tag{5.83}
\]

From Theorem 5.6 we know that (5.83) will have roots in the interval \((-\Lambda_2, -\Lambda_1)\) if and only if

\[
\|(H + \mu_0 I)^{-1}g\|_2 \leq \Delta \tag{5.84}
\]

where \(\mu_0\) is given by (5.74).

(a) \((\mu_0 < 0)\).

Since \((\mu_0 < 0)\) and \(\Lambda_1 < 0\), we have \(\mu_0 < 0 < -\Lambda_1\). Theorem 5.6, together with \(\|(H)^{-1}g\|^2 \leq \Delta^2\), and the fact that \(\|(H + \mu I)^{-1}g\|^2 \to \infty\) as \(\mu \to -\Lambda_1^{-1}\), shows that there exists \(\mu^* \in [0, -\Lambda_1)\) such that \(\|(H + \mu^* I)^{-1}g\|^2 = \Delta^2\), which is the local minimizer.

(b) \((\mu_0 \geq 0)\).

In this case, we do not actually need \(\Lambda_1 < 0\) since it follows from \(\mu_0 \geq 0\). Theorem 5.6, together with \(\|(H + \mu I)^{-1}g\|^2 < \Delta^2\), and the fact that \(\|(H + \mu I)^{-1}g\|^2 \to \infty\) as \(\mu \to -\Lambda_1^{-1}\), shows that there exists \(\mu_0 \geq 0\) and \(\mu^* \in (\mu_0, -\Lambda_1)\) such that \(\|(H + \mu^* I)^{-1}g\|^2 = \Delta^2\), which is the local minimizer.
From the definition of \( \mu_t \) and the above arguments, we know \( \mu_t \leq \mu^* \). The constant \( \mu_- \) given in (5.80) is the point where the tangent line \( l^-() \) intersects the trust region constraint. From Lemma 5.4, we know \( l^-() \geq \| (H + \mu I)^{-1}g \|^{-1} \) for \( \mu \in (-\Lambda_2, -\Lambda_1) \), and this implies \( \mu^* \leq \mu_- \). Therefore, \( \mu^* \in [\mu_t, \mu_-] \), if it exists.

From Theorem 5.7, we can use the following logic to determine whether or not a non-global solution exists.

If \( (\Lambda_1 < 0) \), then
  \( \text{If} \ (\mu_0 < 0), \text{then} \)
    \( \text{If} \ (\| H^{-1}g \|^2 \leq \Delta^2), \text{then} \)
      A non-global solution exists on \((\mu_t, \mu_-)\)
    \( \text{Else} \)
      No non-global solution exists
  \( \text{End if} \)
  \( \text{Else} \)
    \( \text{If} \ (\| (H + \mu_0 I)^{-1}g \|^2 < \Delta^2), \text{then} \)
      A non-global solution exists on \((\mu_t, \mu_-)\)
    \( \text{Else} \)
      No non-global solution exists
  \( \text{End if} \)
\( \text{Else} \)
  No non-global solution exists
\( \text{End if} \)

Once we have determined that a local solution exists, we use essentially the same algorithm we used to find the global solution in Section 5.3. We start the algorithm with \( \mu_0 = \mu_- \), and since we know \( \mu^* \leq \mu_- \), Newton’s method produces a monotonically decreasing sequence converging to the solution of \( \phi(\mu) = 0 \). Since the Newton step is not a possibility, we only need the stopping criteria to test that the step is sufficiently close to the boundary of the trust region,

\[
| \Delta - \| s_c \| | \leq \sigma \Delta
\]  

(5.85)

for some tolerance \( \sigma \). Thus, using Newton’s method from \( \mu_- \), we can find the non-global solution in the non-degenerate case if we have determined that it exists.
5.5 Statement of the Algorithm

The following statement of the algorithm first summarizes the analytical expressions for the solutions to problem UTR in the four degenerate cases. These degenerate cases are $g = 0$, $\Lambda_1 = \Lambda_2$, $g$ orthogonal to $v_1$, and $g$ orthogonal to $v_2$. Then, we give the details concerning the iterative procedures we use to find approximations to the global solution and the local, non-global solution, if it exists, in the non-degenerate case. This includes conditions to determine if the local solution exists.

**Algorithm UTR:**

0. **Given** $g \in \mathbb{R}^2$, $H \in \mathbb{R}^{2 \times 2}$ where $H$ is symmetric, and $\Delta > 0$, **find** the global solutions, $s^*_g$ and $s^{**}_g$, and the local solution, $s^*_l$, to problem UTR.

1. Calculate the eigen-decomposition of $H$. Let $\Lambda_1 \leq \Lambda_2$ denote the eigenvalues, and let $v_1$ and $v_2$ denote the corresponding orthonormal eigenvectors.

2. If $(g = 0)$, then,
   
   If $(\Lambda_1 > 0)$, then
   
   $s^*_g = 0$
   
   $\mu^* = 0$

   Else

   If $(\Lambda_1 = 0)$, then
   
   If $(\Lambda_2 > 0)$, then
   
   $s^*_g = (2\gamma - 1)\Delta v_1$ for $\gamma \in [0, 1]$

   Else

   $s^*_g = \{s : \|s\| \leq \Delta\}$

   End if

   Else

   If $(\Lambda_1 = \Lambda_2)$, then
   
   $s^*_g = \{s : \|s\| = \Delta\}$

   Else

   $s^*_g = \Delta v_1$
   
   $s^{**}_g = -\Delta v_1$

   $\mu^* = -\Lambda_1$
3. If \( \Lambda_1 = \Lambda_2 \), then
\[
\mu_+ = -\Lambda_1 + \frac{\|g\|}{\Delta}
\]

If \( \mu_+ \geq 0 \), then
\[
a = \Delta/\|g\|
\]
\[
s_g^* = -a \, g
\]
\[
\mu^* = \mu_+
\]
Else
\[
s_g^* = -\left(1/\Lambda_1\right) \, g
\]
\[
\mu^* = 0
\]
End if
Return
End if

4. Calculate \( c_1 = v_1^T g \)
\[
c_2 = v_2^T g
\]

5. If \( v_1^T g = 0 \), then
\[
\mu_+ = -\Lambda_2 + \frac{|c_2|}{\Delta}
\]

If \( \Lambda_1 > 0 \), then
If \( \mu_+ > 0 \), then
\[
s_g^* = -\text{sign}(c_2) \, \Delta \, v_2
\]
\[
\mu^* = \mu_+
\]
Else
\[
s_g^* = -(c_2/\Lambda_2) \, v_2
\]
\[
\mu^* = 0
\]
End if
Else

If \( \mu_+ > -\Lambda_1 \), then
\[
s_g^* = -\text{sign}(c_2) \Delta v_2 \\
\mu^* = \mu_+
\]
End if

if \( \mu_+ = -\Lambda_1 \), then
\[
a = -c_2/(\Lambda_2 - \Lambda_1) \\
s_g^* = av_2
\]
End if

If \( \mu_+ < -\Lambda_1 \), then
If \( \Lambda_1 \neq 0 \), then
\[
a = -c_2/(\Lambda_2 - \Lambda_1) \\
p = av_2 \\
\tau = \sqrt{\Delta^2 - a^2} \\
s_g^* = p + \tau v_1 \\
s_g^{**} = p - \tau v_1
\]
Else
\[
b = -c_2/\Lambda_2 \\
\tau = \sqrt{\Delta^2 - b^2} \\
s_g^* = bv_2 + (2\gamma - 1)\tau v_1
\]
for \( \gamma \in [0, 1] \)
End if
End if
Return

End if

6. If \( v_2^T g = 0 \), then
\[
\mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta}
\]

If \( \Lambda_1 \geq 0 \), then
If \( \mu_+ > 0 \), then
\[
\begin{align*}
\quad & s_g^* = -\text{sign}(c_1) \Delta v_1 \\
\quad & \mu^* = \mu_+
\end{align*}
\]
Else
\[
\quad s_g^* = -(1/\Lambda_1) g
\]
\( \mu^* = 0 \)

**End if**

**Else**

\[ s_g^* = -\text{sign}(c_1) \Delta v_1 \]
\[ \mu^* = \mu_+ \]

\[ \mu_- = -\Lambda_1 - (|c_1|/\Delta) \]

**If** (\( \mu_- \geq 0 \) and \( \mu_- > -\Lambda_2 \)), then

\[ s_i^* = \text{sign}(c_1) \Delta v_1 \]
\[ \mu_i^* = \mu_- \]

**End if**

**End if**

**End if**

**Return**

---

7. **Iterative Method to determine the global solution with \( \mu^* \in (-\Lambda_1, \infty) \)**

\[ \mu_+ = -\Lambda_1 + (|c_1|/\Delta) \]
\[ \mu_0 = \max(0, \mu_+) \]
\[ k = 0 \]

i. Solve \((H + \mu_k I)p = -g\) for \(p\)

ii. Check Convergence Criteria

**If** (\(|\Delta - \|p\| | \leq \sigma \Delta\) or \(|\|p\| \leq \Delta \text{ and } \mu_k = 0\)), then

\[ s_g^* = p \]
\[ \mu^* = \mu_k \]

GoTo 8.

**End if**

iii. Take a Newton step

\[ \alpha = \frac{c_1^2}{(\Lambda_1 + \mu_k)^3} + \frac{c_2^2}{(\Lambda_2 + \mu_k)^3} \]
\[ \mu_{k+1} = \mu_k + \frac{\|p\|^2}{\alpha} \left( \frac{\Delta - \|p\|}{\Delta} \right) \]

iv. \( k = k + 1 \)

GoTo 7i.

8. Determine if there is a local solution with \( \mu^* \in (-\Lambda_2, -\Lambda_1) \)

\[ \text{If (} \Lambda_1 < 0, \text{ then) } \]
\[ \alpha = -\left( \frac{c_1^2}{c_2^2} \right)^\frac{1}{3} \]
\[ \mu_0 = (\alpha \Lambda_2 - \Lambda_1)/(1 - \alpha) \]

\[ \mu_1 = \max(0, \mu_0) \]
\[ \mu_- = -\Lambda_1 - (|c_1|/\Delta) \]

\[ \text{If (} \mu_0 < 0, \text{ then, then) } \]
\[ \text{If (} \|H^{-1}g\|^2 \leq \Delta^2 \text{), then} \]
\[ \text{A non-global solution exists} \]
\[ \text{on } (\mu_1, \mu_-) \]

Else

\[ \text{No non-global solution exists} \]

Return

End if

Else

\[ \text{If (} \|(H + \mu_0 I)^{-1}g\|^2 < \Delta^2 \text{), then} \]
\[ \text{A non-global solution exists} \]
\[ \text{on } (\mu_1, \mu_-) \]

Else

\[ \text{No non-global solution exists} \]

Return

End if

End if

Else

\[ \text{No non-global solution exists} \]
9. Iterative Method to determine the local solution with $\mu_i^* \in (\mu_1, \mu_-)$

$k = 0$
$\mu_0 = \mu_-$

i. Solve $(H + \mu_k I)p = -g$ for $p$

ii. Check Convergence Criteria

$$\text{If } (|\Delta - \|p\| | \leq \sigma \Delta), \text{ then}$$
$$s_i^* = p$$
$$\mu_i^* = \mu_k$$
Return

End if

iii. Take a Newton step

$$\alpha = \frac{c_1^2}{(\Lambda_1 + \mu_k)^3} + \frac{c_2^2}{(\Lambda_2 + \mu_k)^3}$$
$$\mu_{k+1} = \mu_k + \frac{\|p\|^2}{\alpha} \left( \frac{\Delta - \|p\|}{\Delta} \right)$$

iv. $k = k + 1$
GoTo 9i.
5.5.1 Accuracy in the Trust Region Subproblems

In this section, we will consider how accurately we need to solve the two-dimensional trust region subproblems

\textbf{Problem TR:} \begin{align*}
\text{minimize} & \quad q_c(s) \\
\text{subject to} & \quad ||s|| \leq \Delta_c \\
& \quad s \in \text{span}\{v_1, v_2\}
\end{align*}

and

\textbf{Problem LF:} \begin{align*}
\text{minimize} & \quad q \\
\text{subject to} & \quad ||\nabla h_c^T s + h_c - \Theta_{MIN}|| \leq \theta_c \\
& \quad s \in \text{span}\{v_1, v_2\}.
\end{align*}

In unconstrained optimization, the trust region subproblem is usually not solved to any great accuracy. See, for example, Dennis and Schnabel [1983]. Since the trust region radius is never increased or decreased by a factor smaller than 2, it is reasonable to ask only that a solution to the unconstrained trust region subproblem, \( s(\mu) \), satisfy

\begin{equation}
|\Delta_c - ||s(\mu)||| \leq \sigma_\Delta \Delta_c \tag{5.86}
\end{equation}

when \( s(\mu) \) is not the Newton step. Typically, \( \sigma_\Delta \in (.1, .5) \).

Constrained optimization problems, on the other hand, are complicated by the interaction between the objective function and the constraints and thus require more care in the determination of \( \sigma_\Delta \). In the course of calculating a trial step, there are four situations where we will need to use a test like (5.86). These situations are deciding if a step is an acceptable solution to Problem TR, determining if a solution to Problem TR satisfies the required linear feasibility, deciding if a step is an acceptable solution to Problem LF, and determining if a solution to Problem LF satisfies the trust region constraint.

First consider finding an approximate solution to Problem TR. Recall that we defined the required amount of linear feasibility, \( ||\nabla h_c^T s + h_c - \Theta_{MIN}|| \leq \theta_c \), based on a trust region of \(.8\Delta_c\) to insure that the intersection of this constraint with the trust region constraint yields a feasible region containing more than a single point. Suppose that \( s_{TR} \) is an approximate solution to Problem TR and that it is not the Newton
step. How accurately do we need to compute this approximate solution? Clearly, if the exact solution to this subproblem satisfies the required linear feasibility, we would like our approximate solution to also satisfy it. This suggests that we choose \( \sigma_{\Delta} < .2 \) to insure that the approximate solution will lie outside of the .8 trust region. For this implementation, we have chosen a conservative \( \sigma_{\Delta} = .05 \). A conservative choice for \( \sigma_{\Delta} \) will not noticeably affect the amount of computation since we are solving only two-dimensional subproblems.

Once we have calculated an approximate solution \( s_{TR} \) to Problem TR, we now want to determine if this solution satisfies the required linear feasibility. The obvious test is

\[
\| \nabla h_c^T s_{TR} + h_c - \Theta_{MIN} \| \leq (1 + \sigma_\theta)\theta_c \tag{5.87}
\]

where \( \sigma_\theta = \sigma_{\Delta} \). However, our strategy for updating the penalty constant requires

\[
\| \nabla h_c^T s_c + h_c - \Theta_{MIN} \| < \| h_c \|. \tag{5.88}
\]

Condition (5.88) can be enforced by choosing \( \sigma_\theta \) in (5.87) as

\[
\sigma_\theta = \min \left( \sigma_{\Delta}, \gamma \left( \frac{\| h_c \|}{\theta_c} - 1 \right) \right), \tag{5.89}
\]

where \( 0 < \gamma < 1 \). For example, \( \gamma = 0.95 \).

We must also enforce (5.88) when deciding if a step is an acceptable approximate solution to Problem LF when the solution is not the Newton step. To accomplish this, we use \( \sigma_\theta \) given by (5.89) in

\[
| \theta_c - \| \nabla h_c^T s + h_c - \Theta_{MIN} \| | \leq \sigma_\theta \theta_c .
\]

The more liberal, but unsymmetric test

\[
(1 - \sigma_{\Delta})\theta_c \leq \| \nabla h_c^T s - h_c + \Theta_{MIN} \| \leq (1 + \sigma_\theta)\theta_c
\]

is also sufficient. Once we have a solution to Problem LF, we will use (5.86) to determine if this solution also satisfies the trust region constraint.
Chapter 6

The Nonlinear Programming Algorithm

In this chapter, we will discuss the remaining ingredients in our nonlinear programming algorithm. We have presented the solution of our trust region subproblem and the calculation of a trial step. Now we must consider how to evaluate the trial step. This requires the choice of a merit function, the determination of the penalty parameter in the merit function, and the calculation of Lagrange multiplier estimates. Although we will discuss the choice of each of these components separately, they are all interrelated.

Finally, after we have presented the entire algorithm, we will give a few of the details about our preliminary implementation of the algorithm. Then we will give numerical results for this implementation, and we will compare it to other available nonlinear programming codes.

6.1 The Choice of a Merit Function

The merit function plays an important role in trust region algorithms. It is used to decide whether the step obtained from the subproblem gives a new iterate that is a better approximation to the solution \( x^* \) than the current iterate. The merit function is used to accept or reject the trial step and to update the radius of the trust region. The choice of a merit function in trust region algorithms for unconstrained optimization is obvious; simply use the objective function. However, in constrained optimization the situation is more complex. Any measure of improvement must balance improvement in the objective function with improvement in the constraint error. Thus, an effective merit function for a constrained optimization algorithm will include a weighted combination of the objective function and the error in the constraints. Given a particular form of merit function, it is often the choice of the weights that is one of the most difficult and elusive tasks in the implementation of the algorithm.
Vardi [1980], [1985] and Byrd, Schnabel and Schultz [1987] choose the \( \ell_1 \) penalty function
\[
\phi_1(x) = f(x) + \sum_{i=1}^{m} \rho_i | h_i(x) |
\]
for the merit function. Both of these applications require that the penalty constants (weights) \( \rho_i \) be sufficiently large.

Celis, Dennis and Tapia [1985] and Powell and Yuan [1986] choose for the merit function the augmented Lagrangian
\[
L(x, \lambda) = f(x) + \lambda^T h(x) + \rho h(x)^T h(x).
\]
However, they made different choices for the Lagrange multipliers \( \lambda \) and the penalty constant \( \rho \).

Powell and Yuan choose for the multipliers in the augmented Lagrangian the least squares multipliers
\[
\lambda_+ = -\left( \nabla h(x_+)^T \nabla h(x_+) \right)^{-1} \nabla h(x_+)^T \nabla f(x_+) \tag{6.1}
\]
which is the least squares solution to \( \nabla_x l(x_+, \lambda) = 0 \). With this choice of multipliers, the augmented Lagrangian becomes a function of \( x \) alone and becomes what Powell refers to as the Fletcher exact penalty function, Tapia [1983]. However, it has the computational disadvantage of requiring the evaluation of \( \nabla h(x_+) \) and computation of the QR factorization of \( \nabla h(x_+) \) for every trial step. This work will be wasted if the step is not accepted. Powell and Yuan also require \( \rho \) to be sufficiently large and define it iteratively so that it attains this goal.

El-Alem [1988] also used the augmented Lagrangian as the merit function to prove global convergence of the CDT algorithm. However, given a trial step \( s_c \), he made the following choice for the multiplier update:
\[
\Delta \lambda_c = -\left( \nabla h_c^T \nabla h_c \right)^{-1} (B_c s_c + \nabla_x l(x_c, \lambda_c)) \tag{6.2}
\]
Following Celis, Dennis and Tapia [1985] and El-Alem [1988], we will use the augmented Lagrangian as the merit function in our algorithm. The multipliers that we will use incorporates both the least squares multipliers (6.1) and the multipliers given by (6.2), and they can be considered to be an efficient implementation of the least squares multipliers.
6.2 Choice of Lagrange Multiplier Estimates

In this section we will discuss the choice of the Lagrange multipliers and the numerical experimentation that led to this choice. Our strategy was forced on us by some interesting behavior we observed in situations when negative curvature existed inside the null space of $\nabla h_c^T$. This caused us to treat the three roles of the Lagrange multipliers separately. The multipliers are used in deciding whether or not to accept the step, in testing for convergence, and in building a new quadratic model for the next iteration.

After we have a trial step $s_c$, we want to use the information we have about the model at the current point to calculate a multiplier update $\Delta \lambda_c$, and we will use the trial multiplier $\dot{\lambda}_c = \lambda_c + \Delta \lambda_c$ to decide whether or not to accept the step. The multiplier update we first tested is the update which is obtained as a least squares solution of

$$\nabla h_c \Delta \lambda = -(B_c s_c + \nabla_x l(x_c, \lambda_c)),$$  \hfill (6.3)

and the resulting $\dot{\lambda}_c$ is then the least squares solution to $\nabla_x l(x_c, \lambda_c + \Delta \lambda_c) = 0$. This is the multiplier update that El-Alem [1988] used to prove the global convergence of the original CDT algorithm. We will denote this update by $\Delta \lambda_M$ for model multipliers since they use only the current model information. This update has some nice properties. First, if $s_c$ is the SQP step, then $\Delta \lambda_c$ is the SQP multiplier $\Delta \lambda_{QP}$ that we obtained during the solution of the QP. If $s_c = 0$, then the multiplier update (6.3) is equivalent to the multiplier update given by (6.1) evaluated at $x_c$. Thus, the multiplier update (6.3) varies smoothly between the multiplier update given by (6.1) and the QP multipliers.

Numerical experience indicates that using the trial multipliers determined from (6.3) to decide whether to accept the step, to test for convergence, and as the multipliers in the quadratic model at the iteration works well when second-order sufficiency holds. However, in an effort to improve the observed performance and robustness of the algorithm, we use different trial multipliers when second-order sufficiency does not hold. To motivate our choice of multipliers in this situation, consider problem QP when second-order sufficiency does not hold. When we have a direction of negative or zero curvature which is a descent direction for the quadratic model inside the null space of $\nabla h_c^T$, the quadratic model is unbounded below on the feasible region (linearized constraint manifold). Thus, the linearized constraints are active, but not binding in the sense that moving off of the constraints will not give further decrease
in the quadratic model. This interpretation led us to set $\Delta \lambda_c = 0$ when second-order sufficiency does not hold in the QP.

This strategy usually works well, but it has one subtle flaw. Numerical experience has shown that the algorithm could obtain the solution $x_*$ at which second-order sufficiency would hold if it had the correct multipliers $\lambda_*$. However, with the current estimates of the multipliers that were obtained from the model, the algorithm may not recognize that it has the solution. This situation occurs when the reduced Hessian at $x_*$ with the current multipliers is not be positive definite. When the reduced Hessian is not positive definite, this strategy will use $\Delta \lambda_c = 0$, and the correct multipliers $\lambda_*$ will not be obtained. Without the correct multipliers, the convergence test cannot recognize the solution. In fact, the algorithm with this choice of multipliers exhibited this unacceptable behavior on several of the test problems.

To overcome this difficulty, we use a two-step approach to updating the multipliers. First we use information we have about the model to find trial multipliers to use in accepting the step and updating the trust region radius. (The procedure for evaluating the step and updating the trust region will be discussed in a later section.) If we do not accept the step, then we will reduce the trust region and calculate another trial step from $(x_c, \lambda_c)$. If we accept the step, then we will calculate function information at the new point, $\nabla h(x_+)$ and $\nabla f(x_+)$, to test for convergence and to prepare for the next iteration. Once we have this new function information, we will use it to obtain a better estimate of the new Lagrange multipliers $\lambda_+$ to use in the convergence test.

The second multiplier update $\Delta \lambda_c$ is chosen to be the least squares solution to

$$
\nabla h(x_+) \Delta \lambda = - \left[ \nabla f(x_+) + \nabla h(x_+) \left( \lambda_c + \Delta \lambda_c \right) \right],
$$

(6.4)

and then the new multipliers are

$$
\lambda_+ = \lambda_c + \Delta \lambda_c + \Delta \lambda_c.
$$

(6.5)

Unlike Powell and Yuan [1986], the work needed to solve for the least squares multipliers will not be wasted if the step is rejected, since we have already accepted the step and need $\nabla h(x_+)$ and its factorization for the next iteration.

Table (6.1) gives the numerical test results for each choice of the multipliers discussed above. All of the conditions under which these tests were done are identical to the numerical testing conditions that will be described in the section on numerical results. Section 6.7 is primarily concerned with comparing our algorithm to other
available codes. The test problems are all from Hock and Schittkowski [1981], and the problem numbers refer to the numbers given there. Table 6.1 does not include the test problems for which second-order sufficiency held at every iteration, and for which all versions of the algorithm required the same number of iterations to converge.

The first set of columns in Table 6.1 are the results for the versions of the algorithm using a single multiplier update at each iteration, i.e., \( \lambda_+ = \lambda_+ = \lambda_c + \Delta \lambda_c \). The column labelled \( \Delta \lambda_c = \Delta \lambda_M \) always took the multiplier update to be the least squares solution to (6.3). The column labelled \( \Delta \lambda_c = 0 \) used \( \Delta \lambda_M \) except when second-order sufficiency did not hold, and then \( \Delta \lambda_c \) was set to 0. In several cases, this version of the algorithm failed to find the solution. The column labelled Curv indicates whether or not second-order sufficiency always held. An 'N' or 'Z' indicates that negative or zero curvature was encountered inside the null space of \( \nabla h_c^T \) while 'P' indicates that the reduced Hessian was positive definite at every iteration.

The second set of columns are the results for the versions of the algorithm using a two-step approach to computing the multipliers. The columns labelled \( \Delta \lambda_c = 0 \) and \( \Delta \lambda_c = \Delta \lambda_M \) correspond to the same choices as before for the first trial update which is used to evaluate the step. Then, if the step is accepted, \( \Delta \lambda_c \) is computed to be the least squares solution of (6.1), and the new multipliers are \( \lambda_+ = \lambda_c + \Delta \lambda_c + \Delta \lambda_c \). These strategies for computing the Lagrange multipliers are detailed in following outline where the choices for the first update are (i, ii) and to update a second time or not using the least squares multipliers is determined by choices iii or iv.

Calculating Lagrange Multiplier Estimates:

0. **Given** \( \lambda_c, s_c, \nabla_x l(x_c, \lambda_c), \nabla h_c, \) and \( B_c \):

1. **If** \( (s_c = s_{QP}) \), **Then**

   \[
   \Delta \lambda_c = \Delta \lambda_{QP}
   \]

   **Else**

   **If** (Second-Order Sufficiency Holds), **Then**

   Solve \( \nabla h_c \Delta \lambda = -(B_c s_c + \nabla_x l(x_c, \lambda_c)) \) for \( \Delta \lambda_c \)

   **Else**

   i. \( \Delta \lambda_c = 0 \),

   ii. \( \Delta \lambda_c \) solves \( \nabla h_c \Delta \lambda = -(B_c s_c + \nabla_x l(x_c, \lambda_c)) \)

   **End if**
End if

2. \( \hat{\lambda}_+ = \lambda_c + \Delta \lambda_c \)

3. Evaluate the step.

4. If (Step is accepted), Then

   iii. \( \lambda_+ = \hat{\lambda}_+ \),

   iv. \( \lambda_+ = \hat{\lambda}_+ + \Delta \lambda_c \) where

   \[ \Delta \lambda_c \text{ solves } \nabla h(x_+) \Delta \lambda = - \left[ \nabla f(x_+) + \nabla h(x_+) \hat{\lambda}_+ \right] \]

End if

Notice from the table that for the problems where second-order sufficiency always holds that the two-step approach is usually more efficient than the single update based only on the model information. This is reasonable since the two-step approach uses the newest function information to calculate the new multipliers.

It is interesting to note that the choice of multipliers influenced which solution the algorithm converged to. For the problems for which different versions of the algorithm converged to different local solutions, the solution that was found is indicated by the Roman numeral i, ii, or iii in parenthesis after the number of iterations, and a list of these solutions can be found in Appendix A.

For the problems that encountered zero or negative curvature, the version of the algorithm which sets the single multiplier correction to 0 in this situation failed to find the solution for a significant number of the test problems, as mentioned earlier, and so we will not consider this version further.

Each of the three remaining multiplier strategies should be evaluated for robustness and efficiency. The two-step approach using \( \Delta \lambda_c = 0 \) when second-order sufficiency does not hold has only one failure but the other two strategies each have two or more failures. To consider efficiency, we can compute the average number of iterations per problem for the subset of problems for which all three methods converged. The average number of iterations per problem is 22.04 for the single update with \( \Delta \lambda_c = \Delta \lambda_M \), 14.77 for the two-step multiplier strategy using \( \Delta \lambda_c = 0 \) and 14.81 for the two-step strategy using \( \Delta \lambda_c = \Delta \lambda_M \). Given these considerations, we slightly
prefer the two-step multiplier strategy using $\Delta \lambda_c = 0$, and we will use it to state the algorithm in the remainder of this work.
### Table 6.1 Multiplier Test Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Starting Point</th>
<th>Convergence (Number of Iterations)</th>
<th>$\lambda_+ = \lambda_c + \Delta \lambda_c$</th>
<th>$\lambda_+ = \lambda_c + \Delta \lambda_c + \Delta \lambda_e$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Curv $\Delta \lambda_c = 0$</td>
<td>$\Delta \lambda_c = \Delta \lambda_M$</td>
<td>Curv $\Delta \lambda_c = 0$</td>
</tr>
<tr>
<td>6</td>
<td>-1.2 1</td>
<td>N F 13</td>
<td>N 13</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>-2.4</td>
<td>N F 16</td>
<td>N 16</td>
<td>16</td>
</tr>
<tr>
<td>26</td>
<td>0 0 0</td>
<td>Z/N F 17</td>
<td>N 19</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>1 -1 1</td>
<td>N F 19</td>
<td>N 18</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>5 -5 5</td>
<td>N 28 27</td>
<td>N 25</td>
<td>25</td>
</tr>
<tr>
<td>27</td>
<td>2 2 2</td>
<td>N 17 8</td>
<td>N 15</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1 4 2</td>
<td>N 16 19</td>
<td>N 17</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>1 4 4</td>
<td>N 19 17</td>
<td>N 34</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>-4 -2 -1</td>
<td>N 458 18</td>
<td>P 18</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>0 2 2</td>
<td>N 10 12</td>
<td>N 10</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>-10 -10 -10</td>
<td>P 49 49</td>
<td>P 32</td>
<td>32</td>
</tr>
<tr>
<td>39</td>
<td>2 2 2 2</td>
<td>N 114 104</td>
<td>P 11</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>0 2 0 0</td>
<td>Z/N F F</td>
<td>Z/N 5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>-10 -10 10 10</td>
<td>N F 18</td>
<td>N 14</td>
<td>14</td>
</tr>
<tr>
<td>40</td>
<td>-1 -1 -1 -1</td>
<td>N 11 24</td>
<td>N 12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>-2 -4 -4 -2</td>
<td>N F 79(ii)</td>
<td>N 14(i)</td>
<td>14(i)</td>
</tr>
<tr>
<td></td>
<td>1 0 -1 0</td>
<td>N 7 7</td>
<td>N 7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>-2 -4 6 2</td>
<td>N F 21</td>
<td>N 14</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>0 -0.5 1 0</td>
<td>N 14(ii) 13(ii)</td>
<td>N 13(i)</td>
<td>13(i)</td>
</tr>
<tr>
<td>60</td>
<td>10 10 10</td>
<td>P 12 12</td>
<td>P 11</td>
<td>11</td>
</tr>
<tr>
<td>77</td>
<td>2 2 2 2 2</td>
<td>P 8 8</td>
<td>P 9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>10 10 10 10</td>
<td>P 18 18</td>
<td>P 17</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>4 3 7 -5 -3</td>
<td>N 34 23</td>
<td>N F</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>-1 3 -0.5 -2 -3</td>
<td>N F F</td>
<td>N 335r</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>-2 -2 -2 -2</td>
<td>N F F</td>
<td>N 124(iii)</td>
<td>236(iii)</td>
</tr>
<tr>
<td>78</td>
<td>-10 10 10 -10 10</td>
<td>N F 13(ii)</td>
<td>P 7(i)</td>
<td>7(i)</td>
</tr>
<tr>
<td></td>
<td>-1 -1 -1 -1 -1</td>
<td>N F 6</td>
<td>P 5</td>
<td>5</td>
</tr>
<tr>
<td>79</td>
<td>-1 2 1 -2 -2</td>
<td>N 7 9</td>
<td>N 7</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>10 10 10 10</td>
<td>N 15 15</td>
<td>N 18</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>-2 -2 -2 -2</td>
<td>P 11 11</td>
<td>P 8</td>
<td>8</td>
</tr>
</tbody>
</table>

**F** Failed to converge.

**P** Second-order sufficiency always holds.

**Z,N** Zero, negative curvature encountered in the null space of $\nabla h^T$.

**r** Restart penalty constant.

**i,ii,iii** Local solution found.
6.3 Choosing the Penalty Constant

The global convergence theory of the original CDT algorithm, El-Alem [1988], requires that the sequence of penalty constants, \( \{\rho_0, \rho_1, \rho_2, \ldots\} \), be nondecreasing and that the predicted reduction in the merit function at each iteration be at least as much as a fraction of Cauchy decrease in \( \|h_c + \nabla h_c^T s\| \). El-Alem [1988] gives a scheme for updating the penalty constant to achieve these objectives. In his scheme, the penalty constant is updated before every step is evaluated. However, numerical experience has shown that success of the algorithm depends on keeping the penalty constant as small as possible. Thus, we have modified the penalty constant given in El-Alem [1988] slightly. We will update the penalty constant \( \rho_c \) to \( \hat{\rho}_c \) before we evaluate each step, and, if we accept the step, then we keep will keep the updated penalty constant, \( \rho_+ = \hat{\rho}_c \). However, if we do not accept the step, we will not keep the update, and \( \rho_+ = \rho_c \). This strategy is designed to keep the penalty constant from becoming unnecessarily large in situations where we must calculate several trial steps while reducing the trust region radius before we find an acceptable step. The penalty constant update depends on the predicted reduction in the merit function \( pred_c \) which is given by (6.8). We use the following strategy for updating the penalty constant.

**Updating the Penalty Constant:**

- **Given** a small, fixed constant \( \beta > 0 \), let \( \rho_0 = 1.0 \).

- **If** \( pred_c \geq \frac{1}{2} \rho_c (\|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2) \), **then**
  \[
  \hat{\rho}_c = \rho_c
  \]
- **Else**
  \[
  \hat{\rho}_c = 2 \frac{\nabla s_c^T s_c + \frac{1}{2} s_c^T B_c s_c + \Delta h_c T (h_c + \nabla h_c^T s_c)}{\|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2} + \beta
  \]
- **End if**

- **If** (Step is Accepted), **Then**
  \[
  \rho_+ = \hat{\rho}_c
  \]
- **Else**
  \[
  \rho_+ = \rho_c
  \]
- **End if**
Notice that this strategy requires that we predict a reduction in the model of the constraints,
\[ \|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2 > 0, \] (6.6)
to insure that the sequence of penalty constants is nondecreasing. If condition (6.6) does not hold, then it is possible that the updated penalty constant \( \hat{p}_c \) could be negative. If the trust region subproblem was solved exactly, then (6.6) will hold. However, in practice, the subproblem is only solved approximately, and care must be taken to ensure (6.6) holds. See Section 5.5.1 which concerns the accuracy in the trust region subproblems for more details. In addition, a necessary property of the merit function is that it must predict improvement for some \( s \) unless \( x_c \) is optimal. This property holds if the penalty constant is sufficiently large and the predicted reduction in the model of the constraints is positive.

Numerical experience indicates that the algorithm does not perform well when the penalty constant becomes too large. We have found that it is advantageous to ‘restart’ the algorithm when the penalty constant becomes large. After we have accepted the step, we reset the penalty parameter to \( p_0 \) if \( p_+ > p_{MAX} \), and for this implementation, we set \( p_{MAX} = 1 \times 10^6 \). This restarting of the penalty constant is indicated in Table 6.1 by an ‘r’ following the number of iterations. As the table shows, only two of the problems actually restarted successfully. The first of these is problem 77 from \( x_0 = (-1, 3, -0.5, -2, -3)^T \). With the restarting procedure, this problem converged in 335 iterations while without restarting the penalty constant, the algorithm failed to converge in 500 iterations. For problem 77 from \( x_0 = (-2, -2, -2, -2, -2)^T \), the algorithm converged in 236 iterations with the restarting procedure, and the algorithm converged in 453 iterations without the restarting procedure.

The values of the constants that we use in our implementation of this penalty constant strategy are \( p_0 = 1.1, \beta = 0.1, \) and \( p_{MAX} = 1 \times 10^6 \). We comment that for a larger value of \( p_{MAX} \), the penalty constant became too large and the algorithm failed for one of the test cases.

### 6.4 Evaluating the Step and Updating the Trust Region Radius

Once we have all of the ingredients in the merit function, we are ready to evaluate the trial step \( s_c \). To measure improvement, we compare the actual reduction in the
augmented Lagrangian from the current iterate \((x_c, \lambda_c)\) to the new iterate \((x_+, \hat{\lambda}_+)\),

\[
ared_c = L(x_c, \lambda_c) - L(x_+, \hat{\lambda}_+)
\]

\[
= l(x_c, \lambda_c) - l(x_+, \lambda_c) - (\hat{\lambda}_+ - \lambda_c)^T h_+ \\
+ \frac{1}{2} \hat{\rho}_c (\|h_c\|^2 - \|h_+\|^2)
\]

(6.7)

to the predicted reduction,

\[
pred_c = -\nabla_x l(x_c, \lambda_c)^T s_c - \frac{1}{2} s_c^T B_c s_c - (\hat{\lambda}_+ - \lambda_c)^T (h_c + \nabla h_c^T s_c) \\
+ \frac{1}{2} \hat{\rho}_c [\|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2]
\]

(6.8)

If the agreement between the actual and predicted reduction is good, i.e.,

\[
\frac{ared_c}{pred_c} \geq \eta_1
\]

where \(\eta_1 \in (0, 1)\) is a small, fixed constant, then the point \(x_+ = x_c + s_c\) is accepted and \(\lambda_+\) is computed from (6.4). Otherwise, we will reject the step and set \(x_+ = x_c\) and \(\lambda_+ = \lambda_c\). We will compute a shorter step by decreasing the trust region radius by \(\Delta_+ = \alpha_2 \|s_c\|\) where \(0 < \alpha_2 < 1\).

If the step is accepted, then we update the trust region radius by comparing the actual and predicted reduction in the merit function. Namely, if

\[
\eta_1 \leq \frac{ared_c}{pred_c} \leq \eta_2
\]

where \(\eta_2 \in (\eta_1, 1)\) is a fixed constant, then the radius of the trust region is possibly decreased by the rule

\[
\Delta_+ = \min\{\Delta_c, \alpha_3 \|s_c\|\}
\]

where \(\alpha_3 > 1\). However, if \(ared_c/pred_c > \eta_2\), then we increase the radius of the trust region by

\[
\Delta_+ = \min\{\Delta_{MAX}, \max\{\Delta_c, \alpha_3 \|s_c\|\}\}
\]

where \(\Delta_{MAX}\) is the maximum allowable trust region radius.

The values of the trust region constants that we use in our implementation are \(\eta_1 = 0.001\), \(\eta_2 = 0.5\), \(\alpha_2 = 0.5\), \(\alpha_3 = 4.0\), and \(\Delta_{MAX} = 20 \Delta_0\).

In the future, we plan to consider the strategy of internal doubling when we update the trust region radius as a way of increasing efficiency, Dennis and Schnabel [1983].
6.5 Statement of the Algorithm

Now that we have discussed each piece of the merit function and the strategy for accepting the step and updating the trust region, we can fit all of these pieces together into the following nonlinear programming algorithm.

The Nonlinear Programming Algorithm:

0. Initialization:
   Get $x_0$, $\lambda_0$, $\Delta_0$, $\rho_0$, and the constants
   $\Delta_{MAX}$, $\rho_{MAX}$, $\beta$, $\varepsilon$, $0 < \alpha_2 < 1, \alpha_3 > 1$, and $0 < \eta_1 < \eta_2 < 1$

1. Calculate a trial step $s_c$.

2. $x_+ = x_c + s_c$.

3. Estimate the Lagrange multiplier update:
   If $(s_c = s_{QP})$, Then
   $$\Delta \lambda_c = \Delta \lambda_{QP}$$
   Else
   If (Second-Order Sufficiency Holds), Then
   Solve $\nabla h(x_c) \Delta \lambda = -(B_c s_c + \nabla_x l(x_c, \lambda_c))$
   for $\Delta \lambda_c$
   Else
   $$\Delta \lambda_c = 0$$
   End if

4. Get $f_+ \equiv f(x_+)$ and $h_+ \equiv h(x_+)$.

5. Calculate the Predicted Reduction:
   $$\text{pred}_c = -\nabla_x l_c^T s_c - \frac{1}{2} s_c^T B_c s_c - \Delta \lambda_c^T (h_c + \nabla h_c^T s_c)
   + \rho_c \left( \|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2 \right) .$$

5. Update the Penalty Constant:
Given a small fixed constant $\beta > 0$;

If $(pred_c \geq \frac{1}{2} \rho_c (\|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2))$, then

$$\rho_c = \rho_c$$

Else

$$\rho_c = 2^{\frac{\nabla x^T s_c + \frac{1}{2} s_c^T B_c s_c + \Delta \lambda_c^T (h_c + \nabla h_c^T s_c)}{\|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2}} + \beta$$

$$pred_c = \nabla x^T s_c + \frac{1}{2} s_c^T B_c s_c - \Delta \lambda_c^T (h_c + \nabla h_c^T s_c) + \rho_c (\|h_c\|^2 - \|h_c + \nabla h_c^T s_c\|^2).$$

End if

6. Calculate the Actual Reduction:

$$ared_c = l_c - l(x_c, \lambda_c) - \Delta \lambda_c^T h(x_c) + \rho_c (\|h_c\|^2 - \|h(x_c)\|^2).$$

7. Evaluate the Step and Update the Trust Region:

Given constants $0 < \alpha_2 < 1, \alpha_3 > 1$, and $0 < \eta_1 < \eta_2 < 1$,

If $(ared_c < \eta_1)$, then

Do not accept the step:

$$x_+ = x_c$$

$$\lambda_+ = \lambda_c$$

Reduce the trust-region radius:

$$\Delta_+ = \alpha_2 \|s_c\|$$

End if

If $(\eta_1 \leq \frac{ared_c}{pred_c} \leq \eta_2)$, then

Accept the step.

Possibly reduce the trust-region radius:

$$\Delta_+ = \min\{ \Delta_c, \alpha_3 \|s_c\| \}$$

End if
\begin{itemize}
    \item \textbf{If} \( \eta_2 < \frac{\text{pred}_{c}}{\text{red}_{c}} \), \textbf{then}
        \begin{align*}
            \text{Accept the step.}
            \end{align*}
        \begin{align*}
            \text{Possibly increase the trust-region radius: }
            \Delta_+ = \min\{ \Delta_{MAX}, \max\{ \Delta_c, \alpha_3 \| s_c \| \} \}
        \end{align*}
\end{itemize}

8. \textbf{If} (Step Not Accepted), \textbf{Then}
    \begin{align*}
        \rho_+ &= \rho_c \\
        f_+ &= f_c, \quad h_+ = h_c, \quad \nabla f_+ = \nabla f_c, \quad \nabla h_+ = \nabla h_c, \quad \text{and} \quad B_+ = B_c \\
        \text{Go To 1.}
    \end{align*}
    \textbf{Else}
    \begin{align*}
        \rho_+ &= \hat{\rho}_c \\
        \text{If} \ (\rho_+ > \rho_{MAX}), \quad \rho_+ = \rho_0
    \end{align*}
\end{itemize}

9. Get \( \nabla f_+ \equiv \nabla f(x_+) \) and \( \nabla h_+ \equiv \nabla h(x_+) \)

10. Update the Lagrange Multipliers:
    \begin{align*}
        \text{Solve} \quad \nabla h_+ \Delta \lambda &= -\nabla f_+ - \nabla h_+ \left( \lambda_c + \Delta \lambda_c \right) \quad \text{for} \quad \Delta \lambda_c \\
        \lambda_+ &= \lambda_c + \Delta \lambda_c + \Delta \lambda_c
    \end{align*}

11. Test for Convergence:
    \begin{itemize}
        \item \textbf{If} \( \| \nabla_x l(x_+, \lambda_+) \| + \| h(x_+) \| \leq \varepsilon \), \textbf{Then}
            \begin{align*}
                \text{Solution found, Stop.}
            \end{align*}
        \item \textbf{Else}
            \begin{align*}
                \text{Get} \quad B_+ \equiv B(x_+, \lambda_+). \\
                \text{Go To 1.}
            \end{align*}
    \end{itemize}
\end{itemize}
6.6 Implementation Details

We have given the constants that we need to evaluate the step, update the trust region, and determine the penalty constant, and we will now give the initialization procedures and the stopping criteria that we use in the preliminary implementation of the algorithm. Given a starting point $x_0$, we use the least squares solution to

$$\nabla h(x_0)\lambda = -\nabla f(x_0)$$

for the initial multipliers $\lambda_0$.

We choose the initial trust region radius to be $\Delta_0 = 1.25\|s_{QP}\|$ when the solution to the QP at $(x_0, \lambda_0)$ exists. If the solution to the QP does not exist, then we use 1.5 times the length of the Cauchy step for the Gauss-Newton model of the constraints as the initial trust region radius. If the Cauchy step from $x_0$ is zero and a direction of negative or zero curvature exists inside the null space of $\nabla h^T$, then there is no obvious way to choose the initial trust region radius, and so we will simply set $\Delta_0 = 1$. We would prefer a choice of $\Delta_0$ that comes from the problem, and in the future, we will perhaps consider $\Delta_0 = c\|x_0\|$.

Finally, we consider $(x_+, \lambda_+)$ to be an acceptable solution based on the stopping criteria

$$\|\nabla x^T(x_+, \lambda_+)\| + \|h(x_+)\| \leq \epsilon$$

where $\epsilon = 1 \times 10^{-6}$. In addition, we consider the algorithm to have failed if it does not converge to a solution in 500 iterations or if the trust region radius falls below $1 \times 10^{-12}$. This part of our preliminary implementation, the stopping criteria, the trust region constants, and the restarting of the penalty constant, for example, have been rather arbitrarily set and will need further work in the future.

6.7 Numerical Results

In this section we report the numerical results for the preliminary implementation of our trust-region algorithm NLPTR in order to evaluate its effectiveness. For comparison, we give results for the n-dimensional trust region algorithm NDIM by Celis, Dennis, and Tapia [1984], and three SQP approaches: VF02AD by Powell [1977], which is available in the Harwell Subroutine Library, NPSOL by Gill, Murray, Saunders, and Wright [1983], and DNCONG by Schittkowski [1986], which is available in the IMSL MATH/LIBRARY. The results for VF02AD and NDIM are taken from Celis [1985] since
we do not have access to either of them at this time. Both NPSOL Version 4.02 and DNCONG were tested using the default stopping criteria and analytic gradient information for both the objective function and the constraints. The maximum number of iterations allowed was 500. All tests were performed in double precision, and NLPTR and DNCONG were tested on a Sun 4 while NPSOL was only available on a SUN 3/160 with a 68881 floating point co-processor.

The problems we tested are from Hock and Schittkowski [1981] and will be referenced by the number given there. These problems are given in an appendix along with possible constrained local minimizers and the function value at these points.

In order to study the robustness of the algorithm, we tested each problem from several starting points. The results are reported in Tables 6.2 and 6.3. The numbers in the columns labeled Convergence indicate the number of iterations required for convergence when convergence was achieved, and the letter F indicates that the algorithm failed to converge.

The number of iterations required for convergence does not provide an accurate comparison of the efficiency of each of the algorithms for a variety of reasons. One of the most significant of these is that NLPTR and NDIM both use exact Hessian information while VF02AD, NPSOL and DNCONG do not. In addition, our algorithm is still in the preliminary implementation stage, and it has not been refined for efficiency.

The quality we are really interested in is robustness, and these results clearly show that our algorithm is slightly more robust than DNCONG and significantly more robust than VF02AD and NPSOL. Our algorithm only failed to converge from one starting point. Careful analysis of this problem in the region at which the algorithm failed suggests that this failure could be due to scaling difficulties. Finally, each of the algorithms did not always converge to the same solution, and we have tabulated the solutions which each algorithm found in Table 6.4.
<table>
<thead>
<tr>
<th>Problem</th>
<th>Starting Point</th>
<th>Convergence (Number of Iterations)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>VF02AD</td>
</tr>
<tr>
<td>6</td>
<td>-121</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-24</td>
<td>*</td>
</tr>
<tr>
<td>26</td>
<td>-2622</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>000</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-1-1-1</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>1-11</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>5-55</td>
<td>*</td>
</tr>
<tr>
<td>27</td>
<td>222</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>142</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>144</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-4-2-1</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>022</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-10-10-10</td>
<td>*</td>
</tr>
<tr>
<td>39</td>
<td>22222</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>0200</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-10-101010</td>
<td>*</td>
</tr>
<tr>
<td>40</td>
<td>.8 .8 .8 .8</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-1 -1 -1 -1</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-2 -4 -4 -2</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>1010</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>-2 -4 6 2</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>0-0.5 1 0</td>
<td>*</td>
</tr>
<tr>
<td>60</td>
<td>222</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>111213</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>2.72 .93 .8</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1.41.51.9</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>123</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1.51.51.5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>101010</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>272938</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>111</td>
<td>*</td>
</tr>
</tbody>
</table>

**F** Failed to converge.

† Converged to point which is not a local minimizer.

* Result not available.

r Restart penalty constant.

Table 6.2  Convergence Results
<table>
<thead>
<tr>
<th>Problem</th>
<th>Starting Point</th>
<th>Convergence (Number of Iterations)</th>
<th>VF02AD</th>
<th>NPSOL</th>
<th>DNCONG</th>
<th>NDIM</th>
<th>NLPTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>77</td>
<td>2 2 2 2 2</td>
<td>15</td>
<td>14</td>
<td>15</td>
<td>12</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 10 10 10 10</td>
<td>F</td>
<td>F</td>
<td>61</td>
<td>21</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-3 -3 3 9 0</td>
<td>21</td>
<td>F</td>
<td>26</td>
<td>21</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 8 3 3 0</td>
<td>18</td>
<td>15</td>
<td>20</td>
<td>17</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 3 7 -5 -3</td>
<td>F</td>
<td>F</td>
<td>39</td>
<td>21</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 3 -0.5 -2 -3</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>18</td>
<td>335r</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12 13 14 15 7</td>
<td>50</td>
<td>F</td>
<td>70</td>
<td>21</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-2 -2 -2 -2 -2</td>
<td>39</td>
<td>F</td>
<td>41</td>
<td>F</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 1 1 1 1</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>12</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>-2 1.5 2 -1 -1</td>
<td>*</td>
<td>10</td>
<td>8</td>
<td>*</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 1.5 2 -1 -2</td>
<td>10</td>
<td>14</td>
<td>11</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-10 10 10 -10 -10</td>
<td>21</td>
<td>21</td>
<td>13</td>
<td>9</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 2 1 -2 -2</td>
<td>9</td>
<td>9</td>
<td>10</td>
<td>6</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 -1 -1 -1 -1</td>
<td>9</td>
<td>11</td>
<td>9</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-2 2 2 2 2</td>
<td>7</td>
<td>16</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>2 2 2 2 2</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 3 -0.5 -2 -3</td>
<td>10</td>
<td>8</td>
<td>10</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 2 1 -2 -2</td>
<td>16</td>
<td>19</td>
<td>20</td>
<td>5</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 1 1 1 1</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 10 10 10 10</td>
<td>18</td>
<td>21</td>
<td>19</td>
<td>14</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-2 -2 -2 -2 -2</td>
<td>18</td>
<td>20</td>
<td>25</td>
<td>11</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

F Failed to converge.

† Converged to point which is not a local minimizer.

* Result not available.

r Restart penalty constant.

Table 6.3 More Convergence Results
<table>
<thead>
<tr>
<th>Problem</th>
<th>Starting Point</th>
<th>Solution found</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>.8 .8 .8 .8</td>
<td>* i i * i</td>
</tr>
<tr>
<td></td>
<td>-1 -1 -1 -1</td>
<td>* ii ii * ii</td>
</tr>
<tr>
<td></td>
<td>-2 -4 -4 -2</td>
<td>* ii ii * ii</td>
</tr>
<tr>
<td></td>
<td>1 0 1 0</td>
<td>* ii ii * ii</td>
</tr>
<tr>
<td></td>
<td>-2 -4 6 2</td>
<td>* iii† iii† * ii</td>
</tr>
<tr>
<td></td>
<td>0 -0.5 1 0</td>
<td>* F F * i</td>
</tr>
<tr>
<td>77</td>
<td>2 2 2 2 2</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>10 10 10 10</td>
<td>F F i i i</td>
</tr>
<tr>
<td></td>
<td>-3 -3 3 9 0</td>
<td>ii F ii ii ii</td>
</tr>
<tr>
<td></td>
<td>-1 8 3 3 0</td>
<td>ii ii ii ii</td>
</tr>
<tr>
<td></td>
<td>4 3 7 -5 -3</td>
<td>F F i ii F</td>
</tr>
<tr>
<td></td>
<td>-1 3 -0.5 -2 -3</td>
<td>F F F iv ii</td>
</tr>
<tr>
<td></td>
<td>12 13 14 15 7</td>
<td>i F i i i</td>
</tr>
<tr>
<td></td>
<td>-2 -2 -2 -2</td>
<td>iii F iii F</td>
</tr>
<tr>
<td></td>
<td>1 1 1 1 1</td>
<td>i i i i i</td>
</tr>
<tr>
<td>78</td>
<td>-2 1.5 2 -1 -1</td>
<td>* i i * i</td>
</tr>
<tr>
<td></td>
<td>-1 1.5 2 -1 -2</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>-10 10 10 -10</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>-1 2 1 -2 -2</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>-1 -1 -1 -1</td>
<td>iii iii iii iii</td>
</tr>
<tr>
<td></td>
<td>-2 2 2 2 2</td>
<td>i ii ii i ii</td>
</tr>
<tr>
<td>79</td>
<td>2 2 2 2 2</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>-1 3 -0.5 -2 -3</td>
<td>i ii ii i ii</td>
</tr>
<tr>
<td></td>
<td>-1 2 1 -2 -2</td>
<td>ii i i ii ii</td>
</tr>
<tr>
<td></td>
<td>1 1 1 1 1</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>10 10 10 10</td>
<td>i i i i i</td>
</tr>
<tr>
<td></td>
<td>-2 -2 -2 -2</td>
<td>ii iii ii iii</td>
</tr>
</tbody>
</table>

* Failed to converge.

Result not available.

† Converged to a point which is not a local minimizer.

Table 6.4 Local Solutions Found
Chapter 7

Concluding Remarks

In summary, we have developed a trust region algorithm to solve the equality con­strained optimization problem. Our goal was to develop a robust algorithm which can handle lack of second-order sufficiency away from the solution, and our numerical results show that we have achieved this goal. We gave an algorithm for solving the quadratic programming problem which handles rank degeneracy in the gradient of the constraints in a natural way and provides a direction of zero or negative curvature in­side the null space of $\nabla h^T$ when the solution to the quadratic program does not exist because second-order sufficiency does not hold. Our trust region algorithm is based on the restriction of the original CDT trust subproblem to a relevant two-dimensional subspace, and we give an algorithm for solving our trust region subproblem. As part of the solution of our trust region subproblem, we had to develop a method to de­termine all of the global solutions, and the non-global solution, if it exists, to the standard unconstrained trust region subproblem in two dimensions. Our analysis of this problem led to analytical expressions for the solutions in a number of degener­ate cases, and an algorithm to find the solutions in the non-degenerate case. In the non-degenerate case, we derived necessary and sufficient conditions for the existence of a non-global solution to the unconstrained trust region subproblem. Finally, we investigated the role of the Lagrange multipliers when second-order sufficiency did not hold.
Appendix A

Test Problems

The following test problems can be found in Hock and Schittkowski [1981].

Hock and Schittkowski 6
\begin{align*}
\text{minimize} & \quad (1 - x_1)^2 \\
\text{subject to} & \quad 10(x_2 - x_1^2) = 0
\end{align*}
\[ x_\ast = (1.0, 1.0)^T; \quad f(x_\ast) = 0.0 \]

Hock and Schittkowski 26
\begin{align*}
\text{minimize} & \quad (x_1 - x_2)^2 + (x_2 - x_3)^4 \\
\text{subject to} & \quad (x_2^2 + 1)x_1 + x_3^4 - 3 = 0
\end{align*}
\[ x_\ast = (1.0, 1.0, 1.0)^T; \quad f(x_\ast) = 0.0 \]

Hock and Schittkowski 27
\begin{align*}
\text{minimize} & \quad 0.01(x_1 - 1)^2 + (x_2 - x_1^2)^2 \\
\text{subject to} & \quad x_1 + x_3^2 + 1 = 0
\end{align*}
\[ x_\ast = (-1.0, 1.0, 0.0)^T; \quad f(x_\ast) = 0.04 \]

Hock and Schittkowski 39
\begin{align*}
\text{minimize} & \quad -x_1 \\
\text{subject to} & \quad x_2 - x_1^3 - x_3^2 = 0 \\
& \quad x_1^2 - x_2 - x_4^2 = 0
\end{align*}
\[ x_\ast = (1.0, 1.0, 0.0, 0.0)^T; \quad f(x_\ast) = -1.0 \]
Hock and Schittkowski 40
minimize $-x_1x_2x_3x_4$
subject to $x_1^2 + x_2^2 - 1 = 0$
$x_1x_4 - x_3 = 0$
$x_4^2 - x_2 = 0$

i: $x_* = (0.7937, 0.7071, 0.5297, 0.8409)^T$; $f(x_*) = -0.25$
ii: $x_* = (0.7937, 0.7071, -0.5297, -0.8409)^T$; $f(x_*) = -0.25$
iii: $x_* \approx (0, 1.0, 0, 1.0)^T$; $f(x_*) = 0.0$; (Not a local minimizer)

Hock and Schittkowski 60
minimize $(x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^4$
subject to $x_1(1 + x_2^2) + x_3^4 - 4 - 3\sqrt{2} = 0$

$x_* = (1.105, 1.197, 1.535)^T$; $f(x_*) = 0.0326$

Hock and Schittkowski 77
minimize $(x_1 - 1)^2 + (x_1 - x_2)^2 + (x_3 - 1)^2 + (x_4 - 1)^4 + (x_5 - 1)^6$
subject to $x_1x_4 + \sin(x_4 - x_5) - 2\sqrt{2} = 0$
$x_2 + x_3^4x_4^2 - 8 - \sqrt{2} = 0$

i: $x_* = (1.166, 1.182, 1.380, 1.506, 0.6109)^T$; $f(x_*) = 0.2415$
ii: $x_* = (-1.029, -1.017, 1.355, 1.760, 0.4531)^T$; $f(x_*) = 4.603$
iii: $x_* = (1.089, 1.178, -1.281, 1.748, 0.8912)^T$; $f(x_*) = 5.533$
iv: $x_* = (-0.9896, -0.9142, -1.3028, 1.8932, 0.4975)^T$; $f(x_*) = 9.909$

Hock and Schittkowski 78
minimize $x_1x_2x_3x_4x_5$
subject to $x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$
$x_2x_3 - 5x_4x_5 = 0$
$x_1^3 + x_2^3 + 1 = 0$

i: $x_* = (-1.717, 1.596, 1.827, -0.7636, -0.7636)^T$; $f(x_*) = -2.920$
ii: $x_* = (-1.717, 1.596, 1.827, 0.7636, 0.7636)^T$; $f(x_*) = -2.920$
iii: \( x^* = (-0.6991, -0.8700, -2.790, -0.6967, -0.6967)^T; \quad f(x^*) = -0.8236 \)

Hock and Schittkowski 79

\[
\begin{align*}
\text{minimize} \quad & (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + (x_3 - x_4)^4 + (x_4 - x_5)^4 \\
\text{subject to} \quad & x_1 + x_2^2 + x_3^3 - 2 - 3\sqrt{2} = 0 \\
& x_2 - x_3^2 + x_4 + 2 - 2\sqrt{2} = 0 \\
& x_1 x_5 - 2 = 0
\end{align*}
\]

i: \( x^* = (1.191, 1.362, 1.473, 1.635, 1.679)^T; \quad f(x^*) = 0.0788 \)
i: \( x^* = (-0.7662, 2.667, -0.4682, -1.619, -2.610)^T; \quad f(x^*) = 27.45 \)
iii: \( x^* = (-2.702, -2.990, 0.1719, 3.848, -0.7401)^T; \quad f(x^*) = 649.1 \)
Bibliography


