A Robust Trust Region Algorithm for Nonlinear Programming

Karen Williamson

CRPC-TR90098
April, 1990

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

Revised May, 1991.

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; Revised May, 1991

A Robust Trust Region Algorithm for Nonlinear Programming

Karen Anne Williamson

Abstract

This work 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 nonlinear programming algorithm, and these results verify that it is indeed robust.

Contents

Abstract ii

List of Tables v

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 8

3 Solution of the Quadratic Program 9

3.1 Calculating a Most Linearly Feasible Step 11

3.2 Directions of Zero or Negative Curvature 18

3.3 Formulation of the Algorithm 20

3.4 Statement of the Algorithm 24

4 Calculation of a Trial Step 27

4.1 Determining the Required Amount of Linear Feasibility and the Choice of the Two-dimensional Subspace 29

4.2 Ill-conditioning of the Linear Feasibility Constraint 32

4.3 Solution of Problem QPTR and Related Subproblems 34

4.4 Statement of the Algorithm 36

5 Solution of the Constrained Trust Region Subproblem 2DCTR 40

5.1 Conversion of Subproblem 2DCTR to two dimensions 45

5.2 Conversion of Problem LF into Standard Trust Region Form 46

6 Solution of the Unconstrained Trust Region Subproblem Restricted to Two Dimensions **48**

6.1 Preliminaries 49
6.2 Characterization of the Solutions for the Degenerate Cases 52
6.3 Calculating the Global Solution in the Non-degenerate Case 64
6.4 Existence and Calculation of the Local Solution in the Non-degenerate Case 70
6.5 Statement of the Algorithm 75
6.5.1 Accuracy in the Trust Region Subproblems 81

7 The Nonlinear Programming Algorithm **83**

7.1 The Choice of a Merit Function 83
7.2 Choice of Lagrange Multiplier Estimates 85
7.3 Choosing the Penalty Constant 92
7.4 Evaluating the Step and Updating the Trust Region Radius 93
7.5 Statement of the Algorithm 97
7.6 Implementation Details 99
7.7 Numerical Results 100

8 Concluding Remarks **106**

A Test Problems **107**

Bibliography **111**

Tables

7.1 Multiplier Test Results 89
7.2 Multiplier Test Results 90
7.3 Multiplier Test Results 91
7.4 Effect of the Initial Trust Region Radius 95
7.5 Effect of the Initial Trust Region Radius 96
7.6 Convergence Results 102
7.7 Convergence Results 103
7.8 Convergence Results 104
7.9 Solutions Found 105

Chapter 1

Introduction

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

Problem ENLP: (1.1)

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad h_i(x) = 0, \quad i = 1, \ldots, m, \end{aligned}$$

where f and h_i are assumed to be smooth nonlinear functions such that $f : \mathbb{R}^n \to \mathbb{R}$, $h_i : \mathbb{R}^n \to \mathbb{R}$ 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 ENLP is the function

$$l(x, \lambda) = f(x) + \lambda^T h(x) \tag{1.2}$$

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

$$L(x, \lambda, \rho) = f(x) + \lambda^T h(x) + \rho h(x)^T h(x) \tag{1.3}$$

with penalty constant $\rho \geq 0$.

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

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

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

3. $z^T \nabla_x^2 l(x_*, \lambda_*) z > 0$ 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 ENLP are that there exists $\lambda_* \in \mathbb{R}^m$ such that (x_*, λ_*) is a solution of the nonlinear system of equations

$$\begin{aligned} \nabla_x l(x,\lambda) &= 0 \\ h(x) &= 0, \end{aligned} \tag{1.4}$$

and

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

Assumption 3 is the standard second-order sufficiency condition requiring the Hessian of the Lagrangian to be positive definite on the null space of ∇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, $\nabla_x^2 l(x,\lambda)$, 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: (1.6)

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

for the step s_c and the associated multipliers $\Delta\lambda_c$. The next iterate and multipliers are taken to be $x_+ = x_c + s_c$ and $\lambda_+ = \lambda_c + \Delta\lambda_c$. Thus, a SQP method solves a sequence of quadratic programming problems of the form (1.6) for the SQP step s_c and multipliers $\Delta\lambda_c$.

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, λ_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 $\nabla h(x_c)$ does not have full rank, the linearized constraints, $\nabla h(x_c)^T s + h(x_c) = 0$, may be degenerate, or there may not exist a feasible point for problem QP at each intermediate iteration. We will discuss ways of 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 $\nabla h(x)^T$. If there is a direction of negative curvature inside the null space of $\nabla h(x)^T$, then the quadratic model of the Lagrangian is unbounded below on the set of feasible points, and the QP will not have a solution. This situation can also happen if there is a direction of zero curvature inside the null space of ∇h^T. Near a solution to problem ENLP (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 Newton step is the unconstrained minimizer of the quadratic 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 a solution of the unconstrained trust region subproblem

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

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, Δ, and calculate a new trial step in this smaller region.

Trust region algorithms for problem ENLP 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 lies outside the region in which we trust our model, 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

QPTR Subproblem (2.1)

$$\begin{aligned} \text{minimize} \quad & \nabla_x {l_c}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla {h_c}^T s + h_c = 0 \\ & \|s\| \leq \Delta_c \,. \end{aligned}$$

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 constraint manifold $\nabla {h_c}^T s + h_c = 0$ may not intersect the trust region.

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

Vardi Subproblem: (2.2)

$$\begin{aligned} \text{minimize} \quad & \nabla_x {l_c}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla {h_c}^T s + h_c = \Theta_c \\ & \|s\| \leq \Delta_c \,, \end{aligned}$$

where $\Theta_c \in I\!R^m$ is a multiple of h_c chosen to ensure that the feasible set is nonempty. 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 Θ_c.

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

CDT Subproblem: (2.3)

$$
\begin{aligned}
\text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\
\text{subject to} \quad & \|\nabla h_c{}^T s + h_c\| \le \theta_c \\
& \|s\| \le \Delta_c\,,
\end{aligned}
$$

where $\theta_c \in I\!R$ is chosen to be $\|\nabla h_c{}^T \hat{s} + h_c\|$ for some $\hat{s}$ 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 θ_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 $\{s : \|s\| \le \Delta_c\}$ 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 the augmented Lagrangian for a merit function with a specific 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 θ_c. They chose it to be any number that satisfies

$$\theta_c = \min\{\|\nabla h_c{}^T s + h_c\| : \|s\| \le \sigma \Delta_c\} \tag{2.4}$$

for some $0 < \sigma \le 1$. This choice of θ_c is computationally more expensive than the choice based on the Cauchy point for the constraints, and 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). A conceptual advantage of θ_c given by (2.4) is that the SQP step would be chosen automatically whenever it is inside the trust region and the linear constraint manifold intersects the smaller trust region of radius $\sigma\Delta_c$. Instead, Celis, Dennis and Tapia [1985] compute the SQP step and take it as the trial step 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 complete characterization of the solutions to the CDT subproblem was not known. In essence, the CDT subproblem requires the minimization of a possibly non-convex quadratic subject to two quadratic constraints. Unfortunately, when both of the quadratic constraints are binding, the CDT subproblem is too difficult and expensive to solve unless the quadratic objective function is strictly convex. (See Y. Zhang [1988] and Y. Yuan [1987], [1988]).

2.2 The New Trust Region Subproblem

Motivated by the work of Byrd, Schnabel and Shultz [1988] on trust region methods for unconstrained optimization, Dennis, Martinez and Williamson [1991] 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

2DCTR Subproblem:

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

We will refer to this subproblem as the 2DCTR (*2-Dimensional Constrained Trust Region*) subproblem. For the required amount of linear feasibility, we choose θ_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 which is a descent direction for the quadratic model 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 ENLP. Chapter 3 gives our algorithm for solving the quadratic programming problem QP and discusses how we deal with the situation when ∇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. The solution of this subproblem requires the minimization of a possibly non-convex quadratic subject to two quadratic constraints in two dimensions. As we will see, 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 6, 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 7 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:

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla h_c{}^T s + h_c = 0 \end{aligned}$$

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.

First we will briefly summarize the standard approach to the solution of problem QP under the ideal conditions; namely that ∇h has full rank and that the Hessian B restricted to the null space of the linear constraints is positive definite. See, for example, Gill, Murray and Wright, [1981]. 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 the linear system

$$\begin{bmatrix} B & \nabla h \\ \nabla h^T & 0 \end{bmatrix} \begin{bmatrix} s_{QP} \\ \Delta\lambda_{QP} \end{bmatrix} = - \begin{bmatrix} \nabla_x l \\ h \end{bmatrix}. \tag{3.1}$$

The first step in the solution of problem QP is the orthogonal factorization of the matrix $\nabla h = Q\,R$ where Q is $(n \times n)$ and orthonormal and R is $(n \times m)$, nonsingular and upper triangular. Then, the first m columns of the matrix Q, which we will denote by Q_1, provide a basis for the range space of ∇h, and the last $(n-m)$ columns of Q, which we will denote by Q_2, provide a basis for the null space of ∇h^T. The range space component of the solution s_{QP} is completely determined from the solution to the lower triangular system

$$R^T w_1 = -h.$$

The null space component of s_{QP} is determined by minimizing the quadratic objective function restricted to the null space. If the reduced Hessian, $Q_2{}^T B Q_2$, is positive definite, then the null space component is uniquely determined from the solution to the linear system

$$\left[Q_2{}^T B Q_2\right] w_2 = -Q_2{}^T \left(B\, s_{LF} + \nabla_x l\right),$$

and the solution to problem QP is given by

$$s_{QP} = Q_1\, w_1 + Q_2\, w_2.$$

Once s_{QP} has been calculated, the multipliers associated with problem QP, $\Delta\lambda_{QP}$, are determined from the solution of the linear system

$$\nabla h \Delta\lambda = -(B s_{QP} + \nabla_x l)$$

using the QR factorization of ∇h.

Let us consider the difficulties that can arise in the solution of problem QP. Since ∇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 linear constraints $\nabla h^T s + h = 0$, and problem QP will not have a solution. To overcome this obstacle, let s_{LF}, (a most *Linearly Feasible* step), be a solution of the linear least-squares problem

$$\text{minimize } \|\nabla h^T s + h\|,$$

and define Θ_{MIN} to be the residual of the linear constraints at s_{LF},

$$\Theta_{MIN} = \nabla h^T s_{LF} + h\,. \tag{3.2}$$

We will replace the linear constraints in problem QP with $\nabla h^T s + h = \Theta_{MIN}$ which will require the step to be as linearly feasible as possible. Thus, when problem QP does not have a solution because ∇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: (3.3)

$$\begin{aligned} \text{minimize} \quad & \nabla_x l^T s + \frac{1}{2} s^T B s \\ \text{subject to} \quad & \nabla h^T s + h = \Theta_{MIN}\,. \end{aligned}$$

There should be no confusion in using the notation s_{QP} in this way since if a linearly feasible point exists, Θ_{MIN} will be 0 and problem GQP is identical to problem QP.

Otherwise, 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. In addition, we point out that problem GQP can be formulated as the bi-level optimization problem

$$\begin{aligned} \text{minimize} \quad & \nabla_x l^T s + \frac{1}{2} s^T B s \\ \text{subject to} \quad & s \in \operatorname{argmin} \left\{ \|\nabla h^T s + h\| \right\} . \end{aligned}$$

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

$$z^T B(x,\lambda) z > 0 \text{ for all } z \neq 0 \text{ satisfying } \nabla h^T(x) z = 0 . \tag{3.4}$$

If this condition does not hold, then there will be a direction of zero or negative curvature inside the null space of ∇h^T. We will denote such a direction by d_{QP}. These situations yield three possibilities. First, if d_{QP} is a direction of negative curvature, then the quadratic model is unbounded below on the linear constraints, and the QP does not have a solution. If d_{QP} is a direction of zero curvature which makes a nonzero inner product with the gradient, then again the quadratic model is unbounded below on the linear constraints, and the QP does not have a solution. Finally, there is the possibility that the quadratic model is completely flat along a direction of zero curvature, and the QP has an infinite number of solutions.

Thus, our algorithm for the solution of the quadratic programming problem first computes a most linearly feasible step, s_{LF}, which will be the range space component of the solution, if a solution exists. (The nonlinear programming algorithm will use the step s_{LF} regardless of whether or not problem GQP has a solution.) The next step is to form the reduced Hessian and to determine if a solution exists. If a solution exists, the algorithm will find it and its associated Lagrange multipliers. Otherwise, we will calculate a descent direction of zero or negative curvature inside the null space of ∇h^T.

3.1 Calculating a Most Linearly Feasible Step

The first part of the solution of problem GQP is the determination of the step to the linear constraints, s_{LF}. The first step is the calculation of the QR decomposition of

∇h using column pivoting. Notice that this is not the obvious way to solve the linear constraints in the least-squares sense, but keep in mind that the goal is to solve the system given by (3.1), not just the linear constraints. The QR decomposition yields

$$\nabla h \;=\; Q\,R\,\Pi^T \equiv [Q_1 \mid Q_2]\;R\,\Pi^T \tag{3.5}$$

where Q is $(n \times n)$ and orthonormal, R is $(n \times m)$ and upper triangular, and Π is the $(m \times m)$ permutation matrix that describes the column pivoting. This decomposition allows us to estimate the rank of ∇h, which we will denote by r. 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 ∇h. The matrix Q_2 has $(n-r)$ columns which span the null space of ∇h^T. When ∇h does not have full rank, 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 $w_1 \in I\!R^r$ and $w_2 \in I\!R^{(n-r)}$. The step s can now be represented as the sum of two components, one which lies in the column space of ∇h and another which lies in the null space of ∇h^T, i.e.

$$s \;=\; Q_1 w_1 + Q_2 w_2\,. \tag{3.6}$$

Often, $Q_1 w_1$ is referred to as the vertical component of the step and $Q_2 w_2$ as the horizontal component.

Since we have not assumed that ∇h has full rank, we will interpret $\nabla h^T s = -h$ in the least-squares sense. Namely, we want to find a step s_{LF} which is a solution to

$$\text{minimize} \quad \|\nabla h^T s + h\|. \tag{3.7}$$

In addition, we would like s_{LF} to be the *shortest* step to the linearized constraints since we intend to use it in a trust region algorithm. For example, if s_{LF} is outside the trust region, we will want to conclude that there are no linearly feasible points inside the trust region. Using the QR decomposition of ∇h and the representation of s given by (3.6), the least-squares problem (3.7) is equivalent to

$$\text{minimize} \quad \left\| \begin{bmatrix} R_1^T & 0 \\ R_2^T & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + \Pi^T h \right\|, \tag{3.8}$$

and

$$\text{minimize} \quad \left\| \begin{bmatrix} R_1^T w_1 \\ R_2^T w_1 \end{bmatrix} + \Pi^T h \right\|. \tag{3.9}$$

If ∇h has full rank, then $R_1 \equiv R$, and s_{LF} is easily determined from the solution of the lower triangular system

$$R^T w_1 \quad = \quad -\left[\Pi^T h\right] \tag{3.10}$$

with $s_{LF} = Q_1 w_1$.

When ∇h does not have full rank, the most tempting idea is to simply take w_1 as the solution to

$$R_1^T w_1 \quad = \quad -\left[\Pi^T h\right]_r \tag{3.11}$$

where $\left[\Pi^T h\right]_r$ is intended to denote the first r elements of the vector $\Pi^T h$. However, the resulting step is *not* necessarily a solution of the least-squares problem (3.7) as the following example will show.

Example: (3.12)

$$\begin{aligned} \nabla h &= Q R \Pi^T, \quad Q = I, \quad \Pi^T = I, \\ R &= \begin{bmatrix} -4 & 1 & .5 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad \text{and} \quad h = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}. \end{aligned}$$

Solving equation (3.11) yields

$$\hat{s} = -Q_1 \, R_1{}^{-T} \, \left[\Pi^T h\right]_r = \begin{bmatrix} 0.25 \\ -1.25 \\ 0.00 \end{bmatrix}$$

and $\|\nabla h^T \hat{s} + h\| = 1.2748$. On the other hand, the direct calculation of the minimum norm least-squares solution using the pseudo-inverse of ∇h^T yields

$$s^+ = -\left(\nabla h^T\right)^+ h = \begin{bmatrix} 0.2481 \\ -1.1860 \\ 0.0000 \end{bmatrix}$$

and $\|\nabla h^T s^+ + h\| = 1.2117$. Clearly $\hat{s}$ obtained from equation (3.11) is not a least-squares solution. Thus, when ∇h does not have full rank, the QR decomposition of ∇h is not sufficient to determine a least-squares solution to $\nabla h^T s + h = 0$.

Can we use the QR decomposition that we already have to obtain the minimum norm least-squares solution? Surprisingly enough, we can. Contemplation of the structure of the least-squares problem in (3.8) and (3.9) suggests that we would like to eliminate the R_2 term. To accomplish this, we further decompose R using Householder transformations into

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

where T is an $(r \times r)$ nonsingular, upper triangular matrix and Z is an $(m \times m)$ orthonormal matrix. Pivoting is not necessary since we have already selected the columns in R_1 to have full rank. The complete decomposition is then

$$\begin{aligned} \nabla h &= Q R \Pi^T = Q T Z^T \Pi^T && (3.13) \\ &= [Q_1 | Q_2] \begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix} Z^T \Pi^T. && (3.14) \end{aligned}$$

Combining this with (3.8) yields the least-squares problem

$$\text{minimize} \quad \left\| \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + Z^T \Pi^T h \right\|,$$

which reduces to

$$\text{minimize} \quad \left\| T^T w_1 + \left[Z^T \Pi^T h \right]_r \right\|. \tag{3.15}$$

Therefore, w_1 is the unique solution of the lower triangular system

$$T^T w_1 = -\left[Z^T \Pi^T h \right]_r, \tag{3.16}$$

and $s_{LF} = Q_1 w_1$.

Returning to the example in (3.12), we further decompose R into

$$\begin{aligned} R &= \begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix} Z^T \\ &= \begin{bmatrix} 4.0156 & -1.0607 & 0.0000 \\ 0.0000 & -1.4142 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix} \begin{bmatrix} -0.9961 & 0.0623 & -0.0623 \\ 0.0000 & -0.7071 & -0.7071 \\ -0.0880 & -0.7044 & 0.7044 \end{bmatrix}. \end{aligned}$$

Using this decomposition, we can solve equation (3.16) for w_1, and $s_{LF} = Q_1 w_1 = (0.2481, -1.1860, 0.0000)^T$, the same as the solution obtained using the pseudo-inverse of ∇h^T.

The following theorems verify that the decomposition of ∇h given in (3.14) can be used to obtain the minimum norm least-squares solution to $\nabla h^T s + h = 0$.

Theorem 3.1 Let ∇h be an $(n \times m)$ matrix, for $m \leq n$, with the QR decomposition

$$\nabla h \;=\; Q\,R\,\Pi^T \tag{3.17}$$

where Q is $n \times n$ and orthonormal, R is $n \times m$ and upper triangular, and Π is the $m \times m$ permutation matrix that describes the column pivoting. If ∇h has full rank, then the pseudo-inverse of ∇h^T is given by

$$(\nabla h^T)^+ \;=\; Q\,R^{-T}\,\Pi^T. \tag{3.18}$$

Let r denote the rank of ∇h. If ∇h does not have full rank, then R has the structure

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

where R_1 is an $(r \times r)$ nonsingular, upper triangular matrix. The matrix R can be further decomposed so that ∇h can be written as

$$\nabla h \;=\; Q \begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix} Z^T\,\Pi^T, \tag{3.19}$$

where T is an $(r \times r)$ nonsingular, upper triangular matrix and Z is an $(m \times m)$ orthonormal matrix. Then, the pseudo-inverse of ∇h^T can be represented as

$$(\nabla h^T)^+ \;=\; Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T\,\Pi^T. \tag{3.20}$$

Proof To show that the expressions given in equations (3.18) and (3.20) represent the pseudo-inverse of ∇h^T, we must verify that they satisfy the four Moore-Penrose conditions:

1. $\nabla h^T\,(\nabla h^T)^+\,\nabla h^T = \nabla h^T$

2. $(\nabla h^T)^+ \, \nabla h^T \, (\nabla h^T)^+ = (\nabla h^T)^+$

3. $(\nabla h^T \, (\nabla h^T)^+)^T = \nabla h^T \, (\nabla h^T)^+$

4. $((\nabla h^T)^+ \, \nabla h^T)^T = (\nabla h^T)^+ \, \nabla h^T$.

See, for example, Golub and Van Loan [1983]. In the full rank case, it is an easy exercise to verify that equation (3.18) satisfies the Moore-Penrose conditions.

When ∇h is rank deficient, we will show that equation (3.20) satisfies conditions 1 - 4. First,

$$\begin{aligned}
\nabla h^T \, (\nabla h^T)^+ \, \nabla h^T &= \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T \, \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T \\
&= \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T \\
&= \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} Q^T \\
&= \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T = \nabla h^T.
\end{aligned}$$

Next,

$$\begin{aligned}
&(\nabla h^T)^+ \, \nabla h^T \, (\nabla h^T)^+ = \\
&\qquad Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T \, \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T \\
&= Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T \\
&= Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T \\
&= Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T = (\nabla h^T)^+.
\end{aligned}$$

Now we will verify the symmetry requirements.

$$\begin{aligned}
\nabla h^T \, (\nabla h^T)^+ &= \Pi \, Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T \\
&= \Pi \, Z \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} Z^T \, \Pi^T,
\end{aligned}$$

which is symmetric. Finally,

$$\begin{aligned} (\nabla h^T)^+ \nabla h^T &= Q \begin{bmatrix} T^{-T} & 0 \\ 0 & 0 \end{bmatrix} Z^T \Pi^T \Pi Z \begin{bmatrix} T^T & 0 \\ 0 & 0 \end{bmatrix} Q^T \\ &= Q \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} Q^T, \end{aligned}$$

which is also symmetric. Thus, the pseudo-inverse of ∇h^T can be expressed as equation (3.20) when ∇h does not have full rank. □

Theorem 3.2 Let ∇h, Q, R, Π, T, and Z be given as in Theorem 3.1. Then, if ∇h has full rank, the minimum norm solution to

$$\text{minimize} \quad \|\nabla h^T s + h\| \tag{3.21}$$

is given by

$$s_{LF} \;=\; -Q_1 R^{-T} \Pi^T h. \tag{3.22}$$

If ∇h is rank deficient with rank r, then the minimum norm solution to the least-squares problem (3.21) is given by

$$s_{LF} \;=\; -Q_1 T^{-T} \left[Z^T \Pi^T h\right]_r \tag{3.23}$$

where $\left[Z^T \Pi^T h\right]_r$ denotes the first r elements of the vector $Z^T \Pi^T h$.

Proof The proof follows from the fact that the pseudo-inverse yields the minimum norm solution to the linear least-squares problem and from Theorem 3.1. □

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

$$s_{LF} = Q_1 w_1. \tag{3.24}$$

We can also calculate Θ_{MIN}, the residual of the linear constraints, by

$$\Theta_{MIN} \;=\; \nabla h^T s_{LF} + h, \tag{3.25}$$

and if ∇h has full rank, $\Theta_{MIN} = 0$.

3.2 Directions of Zero or Negative Curvature

Now that we have calculated a step to the linear constraints, we must determine if problem GQP has a solution or if there is a descent direction of zero or negative curvature for the quadratic inside the null space of ∇h^T.

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

$$\begin{aligned} q(s) &= \nabla_x l^T (s_{LF} + \alpha d_{QP}) + \frac{1}{2}(s_{LF} + \alpha d_{QP})^T B (s_{LF} + \alpha d_{QP}) \\ &= q(s_{LF}) + \alpha(\nabla_x l^T d_{QP} + s_{LF}^T B d_{QP}) + \frac{1}{2}\alpha^2 d_{QP}^T B d_{QP}\,. \end{aligned}$$

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

$$\alpha(\nabla_x l^T d_{QP} + s_{LF}^T B d_{QP}) \leq 0.$$

Then, as we increase the magnitude of α, it is easy to see that $q(s) \rightarrow -\infty$, for any step 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. (As an aside, notice that if we choose the sign of α so that $\alpha \nabla_x l^T d_{QP} \leq 0$, then a step of the form $s = \alpha d_{QP}$ is a (perhaps infeasible) 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 ∇h^T. Without loss of generality, we will assume that there is not a direction of negative curvature inside the null space of ∇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

$$q(s) = q(s_{LF}) + \alpha \nabla_x l^T d_{QP}\,. \tag{3.26}$$

If $\nabla_x l^T d_{QP} \neq 0$, then $q(s)$ will 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 (possibly infeasible) descent direction for $q(s)$ when the sign of α 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 ∇h^T, then $q(s) = q(s_{LF})$ for all α. Thus, d_{QP} gives us not a descent direction but a direction along which the quadratic is unchanging. However, there may still be a linearly feasible descent direction from $s = s_{LF}$. The step $s = s_{LF}$ takes us to the linearized constraint manifold. 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_{MIN} to the minimizer of the quadratic inside the null space. Combining this step with the step to the null space and the *nzero* directions of zero curvature in which the quadratic is unchanging gives us an infinite number of solutions to problem GQP of the form

$$s_{QP} = s_{LF} + s_{MIN} + \sum_{i=1}^{nzero} \alpha(i) d_{QP}(i) \tag{3.27}$$

for all scalars $\alpha(i), i = 1, \ldots, nzero$.

To illustrate this case, consider the following trivial example,

$$\begin{aligned} &\text{Example:} \\ &\qquad \text{minimize} \quad s_1 - s_2 + \frac{1}{2}s_1^2 + \frac{1}{2}s_2^2 \\ &\qquad \text{subject to} \quad s_1 = -1\,, \end{aligned} \tag{3.28}$$

for $s = (s_1, s_2, s_3)^T$. The step from $s = 0$ to the linear constraint is $s_{LF} = (-1, 0, 0)^T$. Restricting the quadratic to the null space of the constraint yields the reduced quadratic, $-s_2 + \frac{1}{2}s_2^2$. Since the reduced quadratic does not depend on s_3, $d_{QP} = (0, 0, 1)^T$ is a direction of zero curvature inside the null space of the constraint along which the quadratic is constant. Minimizing the reduced quadratic in the s_2 variable gives $s_{MIN} = (0, 1, 0)^T$. Then, our example has an infinite number of solutions of the form

$$s_{QP} = s_{LF} + s_{MIN} + \alpha d_{QP} = \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} + \alpha \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.29}$$

for all scalars α.

Thus, there are three possible solution cases. If second-order sufficiency holds, i.e., if the reduced Hessian is positive definite, then the QP will have a single, unique solution. If there is a direction of negative or zero curvature inside the null space of

∇h^T which is a descent direction for the quadratic model, then the QP does not have a solution. In this situation, the algorithm will calculate a direction d_{QP} as described above and a step s_{LF} to the linear constraints. Finally, if there are $nzero$ directions of zero curvature inside the null space of ∇h^T and none of them are descent directions, then the QP will have an infinite number of solutions of the form given in (3.27). If the algorithm detects this case, (which we admit is unlikely), it will calculate s_{MIN} in addition to s_{LF}.

3.3 Formulation of the Algorithm

The first part of the solution of problem GQP is the determination of the step to the linear constraints, s_{LF} as discussed in Section 3.1. Now we want to determine whether or not the generalized QP has a single solution, no solution, or an infinite number of solutions. The Hessian B restricted to the null space of ∇h^T is $Q_2^T B\, Q_2$. If the reduced Hessian is positive definite, then the QP has a single solution. Let Λ_1 denote the smallest eigenvalue of $Q_2^T B\, Q_2$, and let v_1 denote the corresponding eigenvector. Then, v_1 is a direction of negative curvature inside the null space if $\Lambda_1 < 0$ or a direction of zero curvature if $\Lambda_1 = 0$. If the smallest eigenvalue of the reduced Hessian is negative, then the quadratic is unbounded below on the feasible set, and the QP does not have a solution. Changing the basis from that of the null space to $\mathbb{R}^n$ gives us a direction of negative curvature d_{QP} by

$$d_{QP} = Q_2\, v_1\,.$$

Although there may be more than one direction of negative curvature inside the null space, we will calculate only the one corresponding to the smallest eigenvalue of the reduced Hessian. This direction is the "steepest" direction of negative curvature in the sense that it gives the most negative value of ${d_{QP}}^T B d_{QP}$.

If the smallest eigenvalue of the reduced Hessian is zero, then we must distinguish between flat directions of zero curvature and descent directions of zero curvature along which the quadratic will be unbounded below. From the expression for the quadratic objective function for a step of the form $s = s_{LF} + \alpha d_{QP}$ given in (3.26), the quadratic will be unbounded below on the feasible set if any direction of zero curvature makes a nonzero inner product with the gradient, i.e., if

$$\nabla_x l^T (Q_2 v_i) \neq 0 \text{ for any } i = 1, \ldots, nzero,$$

where v_i, for $i = 1, \ldots, nzero$, are the eigenvectors of the reduced Hessian corresponding to zero eigenvalues. If there is more than one descent direction of zero curvature, then we will take the one which makes the smallest angle with the gradient to obtain d_{QP}.

If the quadratic objective function is flat along all of the directions of zero curvature, then problem GQP will have an infinite number of solutions of the form

$$s_{QP} = s_{LF} + s_{MIN} + \sum_{i=1}^{nzero} \alpha_i d_{QP}(i) \tag{3.30}$$

where $d_{QP}(i) = Q_2 v_i$, for $i = 1, \ldots, nzero$. In this case, s_{MIN} is obtained by minimizing the quadratic restricted to the null space minus the flat directions of zero curvature. Thus, $s_{MIN} = Q_2 w_2$ for

$$w_2 = -(Q_2^T B\, Q_2)^+ \, Q_2^T(\nabla_x l + B s_{LF}) \tag{3.31}$$

where $(\cdot)^+$ denotes the pseudo-inverse of the matrix. Since we will use s_{QP} in a trust region algorithm, we need the following lemma for completeness.

> **Lemma 3.1** Assume that the smallest eigenvalue of the reduced Hessian is zero. Let Q be an orthonormal matrix with the partition given in (3.5), where Q_1 is a basis for the column space of ∇h and Q_2 is a basis for the null space of ∇h^T. Let $s_{LF} = Q_1 w_1$ where w_1 is determined from (3.10) or (3.16), and let $s_{MIN} = Q_2 w_2$ where w_2 is given by (3.31). Let $d_{QP}(i) = Q_2 v_i$ where v_i, for $i = 1, \ldots, nzero$, are the null orthonormal eigenvectors of the reduced Hessian $Q_2^T B\, Q_2$. Assume that
>
> $$\nabla_x l^T d_{QP}(i) = 0 \text{ for all } i = 1, \ldots, nzero\,. \tag{3.32}$$
>
> Then,
>
> $$\|s_{LF} + s_{MIN}\| \leq \|s_{LF} + s_{MIN} + \sum_{i=1}^{nzero} \alpha_i d_{QP}(i)\| \tag{3.33}$$
>
> for all constants α_i.

Proof Expanding the norm yields

$$\begin{aligned} \|s_{LF} + s_{MIN} + \sum_{i=1}^{nzero} \alpha_i d_{QP}(i)\|^2 \;=\;& \|s_{LF}\|^2 + \|s_{MIN}\|^2 + \|\sum_{i=1}^{nzero} \alpha_i d_{QP}(i)\|^2 + \\ & 2{s_{LF}}^T s_{MIN} + 2\sum_{i=1}^{nzero}(\alpha_i {s_{LF}}^T d_{QP}(i)) + \\ & 2\sum_{i=1}^{nzero}(\alpha_i {s_{MIN}}^T d_{QP}(i)). \end{aligned} \tag{3.34}$$

Since the null eigenvectors of the reduced Hessian are orthonormal,

$$d_{QP}(i)^T d_{QP}(j) = v_i{}^T Q_2{}^T Q_2 v_j = v_i{}^T v_j = 0, \text{ for } i \neq j ,$$

and $\|d_{QP}(i)\| = 1$, for $i = 1, \ldots, nzero$. Therefore,

$$\| \sum_{i=1}^{nzero} \alpha_i d_{QP}(i) \|^2 = \sum_{i=1}^{nzero} \alpha_i^2 \|d_{QP}(i)\|^2 = \sum_{i=1}^{nzero} \alpha_i^2 . \tag{3.35}$$

Since $s_{LF} = Q_1 w_1$ and $s_{MIN} = Q_2 w_2$,

$$s_{LF}{}^T s_{MIN} = w_1{}^T Q_1{}^T Q_2 w_2 = 0 . \tag{3.36}$$

Similarly, since each $d_{QP}(i)$ can be written as $Q_2 v_i$,

$$s_{LF}{}^T d_{QP}(i) = w_1{}^T Q_1{}^T Q_2 v_i = 0 , \text{ for } i = 1, \ldots, nzero . \tag{3.37}$$

Expanding the remaining term gives

$$\begin{aligned} d_{QP}(i)^T s_{MIN} &= -v_i{}^T Q_2{}^T Q_2 (Q_2^T B\, Q_2)^+ \, Q_2^T (\nabla_x l + B s_{LF}) & (3.38) \\ &= -v_i{}^T (Q_2^T B\, Q_2)^+ \, Q_2^T (\nabla_x l + B s_{LF}) . & (3.39) \end{aligned}$$

We can write the pseudo-inverse of the reduced Hessian in terms of its eigen decomposition, i.e.,

$$(Q_2^T B\, Q_2)^+ = V(\Lambda)^+ V^T, \tag{3.40}$$

where the columns of V are the orthonormal eigenvectors and Λ is a diagonal matrix whose diagonal elements are the eigenvalues of the reduced Hessian. Let e_i denote the i^{th} unit vector. Then,

$$\begin{aligned} d_{QP}(i)^T s_{MIN} &= -v_i{}^T V(\Lambda)^+ V^T Q_2^T (\nabla_x l + B s_{LF}) & (3.41) \\ &= -e_i{}^T (\Lambda)^+ V^T Q_2^T (\nabla_x l + B s_{LF}) & (3.42) \\ &= 0 , & (3.43) \end{aligned}$$

since the i^{th} diagonal element of $(\Lambda)^+$ corresponds to a zero eigenvalue and is 0. Combining these relations yields

$$\|s_{LF} + s_{MIN} + \sum_{i=1}^{nzero} \alpha_i d_{QP}(i)\|^2 = \|s_{LF}\|^2 + \|s_{MIN}\|^2 + \sum_{i=1}^{nzero} \alpha_i{}^2 , \tag{3.44}$$

and the desired result follows. □

Thus, when the smallest eigenvalue of the reduced Hessian is zero and none of the resulting directions of zero curvature are descent directions for the quadratic objective function, we will take $s_{QP} = s_{LF} + s_{MIN}$ as our particular solution to the quadratic program. Lemma 3.1 guarantees that this particular solution is the minimum norm solution. In the nonlinear programming context, if $s_{QP} = s_{LF} + s_{MIN}$ is not inside the trust region, then none of the infinite number of solutions to problem GQP are inside the trust region.

Therefore, 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, the only remaining possibility is that the reduced Hessian is positive definite. Then, we will compute the component of the step s_{QP} that lies inside the null space of ∇h^T from

$$Bs + \nabla h \Delta\lambda = -\nabla_x l\,. \tag{3.45}$$

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

$$\left[Q_2^T B\, Q_2\right] w_2 = -Q_2^T(\nabla_x l + BQ_1 w_1)\,. \tag{3.46}$$

Since $s_{LF} = Q_1 w_1$, equation (3.46) simplifies to

$$\left[Q_2^T B\, Q_2\right] w_2 = -Q_2^T(\nabla_x l + Bs_{LF})\,. \tag{3.47}$$

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, $\Delta\lambda_{QP}$, from equation (3.45). Substituting the decomposition of ∇h, equation (3.45) becomes

$$[Q_1 \mid Q_2] \begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix} Z^T \Pi^T \Delta\lambda = -(\nabla_x l + Bs_{QP})\,. \tag{3.48}$$

Partition the vector $Z^T \Pi^T \Delta\lambda$ into the first r components, $\left[Z^T \Pi^T \Delta\lambda\right]_r$, and the last $(m-r)$ components, $\left[Z^T \Pi^T \Delta\lambda\right]_{(m-r)}$. The first r elements are determined from the solution of the upper triangular system

$$T\left[Z^T \Pi^T \Delta\lambda\right]_r = -Q_1{}^T(\nabla_x l + Bs_{QP})\,, \tag{3.49}$$

and the last $(m-r)$ components will be set to zero, i.e.,

$$\left[Z^T \Pi^T \Delta\lambda\right]_{(m-r)} \equiv 0\,.$$

Application of the orthogonal transformations represented by Z and the pivot interchanges represented by Π to the resulting vector yield the multipliers associated with the quadratic program, $\Delta\lambda_{QP}$. In the event that ∇h has full rank, the multipliers are simply determined by application of the pivot information Π to the solution of the upper triangular system

$$R\left[\Pi^T \Delta\lambda\right] = -Q_1^T(\nabla_x l + B s_{QP})\,. \tag{3.50}$$

3.4 Statement of the Algorithm

The algorithm for the solution of problem GQP will calculate s_{QP} and $\Delta\lambda_{QP}$ when such a solution exists. When a solution does not exist, a step s_{LF} that satisfies $\nabla h^T s_{LF} + h = \Theta_{MIN}$ and a descent direction of negative or zero curvature inside the null space of ∇h will be found. The preliminary implementation of this algorithm uses the full eigen decomposition of the reduced Hessian to determine the necessary curvature information. A topic for the future will be to replace this decomposition with a symmetric indefinite factorization. The algorithm can be stated as follows.

Algorithm GQP:

1. Obtain h, ∇h, $\nabla_x l$, and B.

2. Calculate the QR decomposition of ∇h using column pivoting.

   (a) $\nabla h = Q\,R\,\Pi^T$.

   (b) Determine the rank of ∇h. Let the rank of $\nabla h = r$.

   (c) 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.

   (d) **If** (∇h not full rank), **then**

   - Eliminate R_2 using Householder transformations to obtain

     $$\nabla h \;=\; Q\begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix} \quad \Pi^T,$$

     where T is an $(r \times r)$ nonsingular, upper triangular matrix and Z is an $(m \times m)$ orthonormal matrix.

End if

3. Calculate the step to the linear constraint manifold.

   (a) **If** (∇h full rank), **then**

   - Solve $R^T w_1 = -\left[\Pi^T h\right]$ for w_1.

   Else

   - Solve $T^T w_1 = -\left[Z^T \Pi^T h\right]_r$ for w_1.

   End if

   (b) $s_{LF} = Q_1 w_1$.

   (c) Calculate the residual of the linear constraints Θ_{MIN}.

4. Determine if problem GQP has a solution or if B has a descent direction of zero or negative curvature inside the null space of ∇h^T.

   (a) Form the reduced Hessian, $Q_2^T B Q_2$.

   (b) Find the smallest eigenvalue of $\left[Q_2^T B Q_2\right]$, Λ_1, and the corresponding eigenvector, v_1.

   (c) **If** ($\Lambda_1 > 0$), **then**

   - solution $= true$

   Else

   - **If** ($\Lambda_1 < 0$), **then**
     * curvature $= negative$
     * solution $= false$
     * $d_{QP} = Q_2 v_1$

     Else

     * curvature $= zero$
     * Determine the number of zero eigenvalues of $Q_2^T B Q_2$, $nzero$, and the corresponding eigenvectors, v_i.
     * **If** ($\nabla_x l^T Q_2 v_i \neq 0$ for any $i \in [1, nzero]$), **then**
       ◇ solution $= false$
       ◇ $imax = \text{argmax}\ \{\ |\nabla_x l^T Q_2 v_i|\ ,\ i = 1, \ldots, nzero\}$

◇ $d_{QP} = Q_2 v_{imax}$

Else

◇ solution $= true$

◇ $d_{QP}(i) = Q_2 v_i \ , i = 1, \ldots, nzero$

End if

End if

End if

5. Calculate the horizontal component of the step.

   (a) **If** (solution $= true$), **then**

   - **If** (curvature $= zero$), **then**
     * $w_2 = -(Q_2^T B\, Q_2)^+ \, Q_2^T(\nabla_x l + Bs_{LF})$.

     Else
     * $w_2 = -(Q_2^T B\, Q_2)^{-1} \, Q_2^T(\nabla_x l + Bs_{LF})$

     End if
   - $s_{QP} = s_{LF} + Q_2 \, w_2$.

   End if

6. Calculate the multipliers.

   (a) **If** (solution $= true$), **then**

   - **If** (∇h full rank), **then**
     * Solve $R \left[\Pi^T \Delta\lambda\right] = -Q_1^T(\nabla_x l + Bs_{QP})$ for $\left[\Pi^T \Delta\lambda\right]$.

     Else
     * Solve $T \ [u]_r = -Q_1^T(\nabla_x l + Bs_{QP})$ for $[u]_r$ where $[u]_r$ denotes the first r elements of $\left[Z^T \Pi^T \Delta\lambda\right]$.
     * Set the last $(m - r)$ elements of u to zero, $[u]_{(m-r)} \equiv 0$.
     * $\left[\Pi^T \Delta\lambda\right] = Z\, u$

     End if
   - "Unpivot" the multipliers: $\Delta\lambda_{QP} = \Pi \ \left[\Pi^T \Delta\lambda\right]$.

   End if

7. 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_c with multipliers λ_c, we will calculate the function information $f_c \equiv f(x_c)$, ∇f_c, B_c, h_c and ∇h_c needed to build a local model of problem ENLP (1.1). We assume that we have the current trust region radius Δ_c. (The strategy for calculating the trust region radius will be discussed in Section 7.4.) 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 $x_+ = x_c + s_c$ that is a better approximation to x_* than the current iterate.

To design an algorithm for calculating a trial step, there are some considerations to keep in mind. First, in order to retain the fast local convergence properties of the SQP method, we will want to take s_{QP} as our trial step whenever it exists and lies inside the trust region. Far from the solution, though, we will need to choose our trial step based on a globalization strategy. The globalization strategy that is the basis of this work is the 2DCTR subproblem, and so we will choose the trial step to be the solution to the two-dimensional constrained trust region subproblem,

2DCTR Subproblem: (4.1)

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

when we are far from the solution.

Using these ideas, the obvious strategy is to first solve the generalized quadratic program,

Problem GQP: (4.2)

$$\text{minimize} \quad \nabla_x {l_c}^T s + \frac{1}{2} s^T B_c s$$

$$\text{subject to} \quad \nabla {h_c}^T s + h_c = \Theta_{MIN} ,$$

for s_{QP}, if problem GQP has a solution, or d_{QP}, a descent direction of zero or negative curvature inside the null space of ∇h_c^T, otherwise. If s_{QP} exists and is inside the trust region, then we will take it as our trial step. If not, we would then solve the 2DCTR subproblem for a trial step. The advantages of this simple and straightforward approach are that it retains the fast local convergence of the SQP method, the cost of the trial step is dominated by the cost of the solution to problem GQP, and the numerical results show good global behavior.

However, there are a number of circumstances in which the constraint

$$\|\nabla {h_c}^T s + h_c\|_2 \leq \theta_c \tag{4.3}$$

can be ill-conditioned, and it is not numerically advisable to use the 2DCTR subproblem. Consider, for example, the situation when $h(x_c) = 0$ and the SQP step is not chosen as the trial step. Then, the linear feasibility constraint (4.3) becomes $\|\nabla h_c^T s\|_2 = 0$. The theory for linear least squares problems, (Golub and Van Loan, Chapter 6), tells us that we would prefer to solve $\|\nabla h_c^T s\|_2 = 0$ directly as $\nabla h_c^T s = 0$ to avoid squaring the condition number of the problem.

We will discuss other specific circumstances in which the constraint (4.3) may be ill-conditioned presently, but first, let us reconsider our design criteria for the trial step algorithm. We know that we want to choose s_{QP} as the trial step whenever it exists and is inside the trust region, and we want to use the 2DCTR subproblem when we are far from the solution. In the simple strategy outlined above, we were "far from the solution" whenever s_{QP} did not exist or was outside the trust region. Instead, suppose we use the distance to the linearized constraint manifold $\nabla {h_c}^T s + h_c = \Theta_{MIN}$ to determine if we are "far from the solution." During the solution of problem GQP, (see Chapter 3), we obtain s_{LF}, the step to the linearized constraint manifold. Then, $\|s_{LF}\|$ is the distance to this subspace since Theorem 3.2 guarantees that s_{LF} is the minimum norm solution of $\{s : \min \|\nabla h_c^T s + h_c\|\}$. In addition, the fact that s_{LF} is not inside the trust region is sufficient to conclude that s_{QP}, if it exists, will not be inside the trust region either.

Using this new notion of "far from the solution," if s_{LF} is inside the trust region, then the natural choice for a subproblem is clearly

QPTR Subproblem (4.4)

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla h_c{}^T s + h_c = \Theta_{MIN} \\ & \|s\| \le \Delta_c \,, \end{aligned}$$

since we know the constraint region is now non-empty. There are two different motivations for using the QPTR subproblem in these circumstances. First, it directly avoids using the possibly ill-conditioned 2DCTR subproblem when $h(x_c) = 0$, since $s_{LF} = 0$ when $h(x_c) = 0$. Next, the subproblem given in (4.4) is a special case of the Vardi subproblem (2.2). Recall that the difficulty with the Vardi subproblem was the choice of Θ_c in the constraint $\nabla h_c{}^T s + h_c = \Theta_c$, but in this situation, the obvious choice is $\Theta_c = \Theta_{MIN}$. Furthermore, it has the desirable property that s_{QP} is the solution to the QPTR subproblem whenever s_{QP} exists and is inside the trust region. So, we will use the QPTR subproblem whenever we know that its feasible set is non-empty. Specifically, we use $\|s_{LF}\| \le .8\Delta_c$ as the criterion for choosing the QPTR subproblem since this guarantees that the feasible set has a non-empty interior.

Now we have essentially a three-phase strategy for calculating the trial step. First, we will solve problem GQP for either a solution s_{QP} or a descent direction of zero or negative curvature, d_{QP}, and the step to linear feasibility, s_{LF}. If s_{QP} is inside the trust region, then we take it as our trial step. Otherwise, we choose our subproblem based on the distance to linear feasibility, $\|s_{LF}\|$. If we are "close" to linear feasibility, then we use the QPTR subproblem to calculate the trial step, and if we are "far" from linear feasibility, then we use the 2DCTR subproblem as our global strategy.

In the next section, we will discuss the choice of the two-dimensional subspace and the required linear feasibility constant θ_c that are needed to complete the specification of problem 2DCTR. Once we have determined the linear feasibility constraint (4.3) the following section is concerned with the circumstances in which this constraint can be ill-conditioned and how we will deal with them.

4.1 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

$$\|\nabla h_c{}^T s + h_c\| \le \theta_c \,, \tag{4.5}$$

and in this section we will discuss how to choose θ_c. The strategy we use is based on the idea that if we choose θ_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

$$\{s : \|\nabla h_c{}^T s + h_c\| \leq \theta_c \text{ and } \|s\| \leq \Delta_c\} \tag{4.6}$$

is non-empty. To preclude the possibility that the feasible set given by (4.6) consists of a single point, we choose $\hat{s}$ such that $\|\hat{s}\| \leq .8\Delta_c$. Celis, Dennis and Tapia [1985] chose θ_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 θ_c to minimize $\|\nabla h_c{}^T s + h_c\|$ inside a trust region of radius $\sigma\Delta_c$ for $0 < \sigma \leq 1$.

Our choice for θ_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 $.8\Delta_c$ as the first segment of the dogleg. The Cauchy point is determined as follows.

Calculating the Cauchy Point: (4.7)

$$s_{CP} = -\frac{h_c{}^T \nabla h_c{}^T \nabla h_c h_c}{h_c{}^T \nabla h_c{}^T \nabla h_c \nabla h_c{}^T \nabla h_c h_c} \nabla h_c h_c$$

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

Now we need the segment of the dogleg step that will play the role that the Newton step plays in the dogleg step 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 in the dogleg strategy 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

$$s_{DL} = s_{CP} + \alpha s_{LF} \text{ such that } ||s_{DL}|| = .8\Delta_c \text{ and } \alpha \geq 0. \tag{4.8}$$

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 (4.8) from $s = 0$ to $s = s_{LF}$. The calculation of the dogleg can be summarized as follows.

Calculating the Dogleg Step:

- **If** ($||s_{LF}|| \leq .8\Delta_c$), **then**

    * $s_{DL} = s_{LF}$

  Else

    * Calculate the Cauchy point from (4.7).
    * $s_{DL} = s_{CP} + \alpha s_{LF}$ such that $||s_{DL}|| = .8\Delta_c$ and $\alpha \geq 0$.

  End if

The details on how to calculate α such that $||s_{CP} + \alpha s_{LF}|| = .8\Delta_c$ can be found in Dennis and Schnabel [1983]. The dogleg step s_{DL} will be zero only when s_{LF} is zero and $s = 0$ is a linearly feasible point.

Once we have found the step which will determine the required linear feasibility, all that remains is to calculate θ_c by

$$\theta_c = ||\nabla {h_c}^T s_{DL} + h_c - \Theta_{MIN}||, \tag{4.9}$$

where the inclusion of Θ_{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 descent 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 (4.6) is non-empty. Thus, the two-dimensional subspace will be

$$s \in \text{span}\{s_{QP} \text{ or } d_{QP};\ s_{DL}\}. \tag{4.10}$$

4.2 Ill-conditioning of the Linear Feasibility Constraint

In this section, we will discuss several situations where the linear feasibility constraint (4.9) may be ill-conditioned. Previously, we used the case $h(x_c) = 0$ to motivate the use of the QPTR subproblem when s_{LF} is inside the trust region. If $h(x_c) = 0$, then (4.9) becomes $\|\nabla h_c^T s\|_2^2 = 0$, and to avoid squaring the condition number of the problem we would prefer to solve $\nabla h_c^T s = 0$ directly.

A similar situation arises when $\nabla h_c h_c = 0$. Notice that this includes the case when x_c is a nonlinearly feasible point, i.e. $h_c = 0$, but it also includes the situation when $s = 0$ lies in the linearized constraint manifold $\nabla {h_c}^T s + h_c = \Theta_{MIN}$ and $h_c \neq 0$. 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, Θ_{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 $\nabla {h_c}^T s = 0$. In both of these situations, the QPTR subproblem reduces to

$$\begin{aligned} \text{minimize} \quad & \nabla_x {l_c}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla h_c^T s = 0 \\ & \|s\| \leq \Delta_c . \end{aligned} \tag{4.11}$$

Since s_{LF} and s_{CP} are zero under these circumstances, switching to the QPTR subproblem when s_{LF} is inside the trust region avoids the possibility of ill-conditioning in the linear feasibility constraint.

Since the subproblem given in (4.11) is a special case of the Vardi subproblem, it can be reduced to a lower dimensional unconstrained trust region subproblem. This reduction will be discussed in the next section. Then, the resulting 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 (4.11). Thus, for efficiency, the algorithm will solve the subproblem in (4.11) first if it detects $\nabla h_c h_c = 0$, and in this situation, problem GQP does not need to be solved.

Next, we will discuss two more special situations that lead us to solve a Vardi-type subproblem (2.2). The first of these is when the value for θ_c from (4.9) is small, and the linear feasibility constraint becomes

$$\|\nabla h_c{}^T s + h_c - \Theta_{MIN}\| \le \theta_c \approx 0. \tag{4.12}$$

This is essentially a translated version of the previous case, and the obvious choice would be to switch to a constraint of the form

$$\nabla h_c{}^T s + h_c - \Theta_{MIN} = 0 \tag{4.13}$$

for numerical conditioning. However, the shortest distance to this linear manifold is $\|s_{LF}\|$. Since we have already dealt with the case when $\|s_{LF}\| \le .8\Delta_c$, it is very possible that s_{LF} is outside the trust region and the intersection of the constraint (4.13) and the trust region is empty. Instead, using the definition of θ_c, (4.12) becomes

$$\|\nabla h_c{}^T s + h_c - \Theta_{MIN}\| \approx \|\nabla h_c{}^T s_{DL} + h_c - \Theta_{MIN}\| .$$

If we now remove the norms, we have a constraint of the form

$$\nabla h_c{}^T s = \nabla h_c{}^T s_{DL} ,$$

which has a non-empty intersection with the trust region since $\|s_{DL}\| \le .8\Delta_c$. This gives us a subproblem of the form

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla h_c{}^T s = \nabla h_c{}^T s_{DL} \\ & \|s\| \le \Delta_c . \end{aligned} \tag{4.14}$$

We have also seen numerical ill-conditioning when

$$\|h_c - \Theta_{MIN}\| \approx \theta_c , \tag{4.15}$$

and $\|\nabla h_c{}^T s + h_c - \Theta_{MIN}\| \approx \theta_c$ over the entire feasible region of problem 2DCTR. Essentially the linearized constraints are not well-scaled in comparison with the trust region radius. Then, the linear feasibility constraint is

$$\|\nabla h_c{}^T s + h_c - \Theta_{MIN}\| \approx \theta_c \tag{4.16}$$

$$\approx \|\nabla h_c{}^T s_{DL} + h_c - \Theta_{MIN}\| \tag{4.17}$$

and with (4.15),

$$\|\nabla h_c{}^T s\| \approx \|\nabla h_c{}^T s_{DL}\|. \tag{4.18}$$

Thus, in this situation, we would prefer a subproblem with a constraint of the form $\nabla h_c{}^T s = \nabla h_c{}^T s_{DL}$ instead of the 2DCTR subproblem. Inserting this structure into the QPTR subproblem yields a subproblem of the form (4.14).

4.3 Solution of Problem QPTR and Related Subproblems

In this section we will briefly outline the solution of the QPTR subproblem and the special cases (4.11) and (4.14) developed in the last section. Each of these subproblems can be reduced to a lower dimensional unconstrained trust region subproblem. We will start with the solution of the simplest one:

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla h_c^T s = 0 \\ & \|s\| \le \Delta_c . \end{aligned}$$

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 $\nabla h_c{}^T$ from the orthogonal decomposition of ∇h_c given in (3.14). With this change of variables, the subproblem in (4.11) becomes

$$\begin{aligned} \text{minimize} \quad & \hat{q}(w) = (Q_2{}^T \nabla_x l_c)^T w + \frac{1}{2} w^T (Q_2^T B\, Q_2) w \\ \text{subject to} \quad & \|w\| \le \Delta_c , \end{aligned} \tag{4.19}$$

and the dimension of the resulting unconstrained subproblem will be the dimension of the null space of $\nabla h_c{}^T$. The trial step is then $s_c = Q_2 w$.

Now we turn our attention to the solution of the QPTR subproblem,

QPTR Subproblem (4.20)

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \nabla h_c{}^T s + h_c = \Theta_{MIN} \\ & \|s\| \le \Delta_c . \end{aligned}$$

Using the definition of Θ_{MIN} given in (3.2), the constraint $\nabla h_c{}^T s + h_c = \Theta_{MIN}$ becomes

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

or

$$\nabla h_c{}^T s = \nabla h_c{}^T s_{LF} . \tag{4.21}$$

Using the orthogonal decomposition of ∇h_c, we can write

$$s = Q_1 w_1 + Q_2 w_2 \tag{4.22}$$

where Q_1 is an orthonormal basis for the column space of ∇h_c and Q_2 is a basis for the null space of $\nabla h_c{}^T$. For subproblem QPTR, the component of the trial step in the range space will simply be s_{LF} since s_{LF} is orthogonal to the null space, and so $s = s_{LF} + Q_2 w_2$. Substituting this into (4.20) yields an unconstrained trust region subproblem of the form

$$\begin{aligned} &\text{minimize} \quad \hat{q}(w_2) = (Q_2{}^T \nabla_x l_c + Q_2{}^T B s_{LF})^T w_2 + \frac{1}{2} w_2^T (Q_2^T B\, Q_2) w_2 && (4.23)\\ &\text{subject to} \quad \|w_2\| \le (\Delta_c - \|s_{LF}\|) . && (4.24) \end{aligned}$$

The form of the trust region constraint (4.24) is due to the fact that s_{LF} is orthogonal to the null space, and so

$$\begin{aligned} \|s\| &= \|s_{LF} + Q_2 w_2\| \\ &= \|s_{LF}\| + \|Q_2 w_2\| \\ &= \|s_{LF}\| + \|w_2\| . \end{aligned} \tag{4.25}$$

We know $(\Delta_c - \|s_{LF}\|) > 0$ since $\|s_{LF}\| \le .8\Delta_c$. Thus, subproblem (4.20) requires the solution of a standard unconstrained trust region subproblem for w_2, and then the trial step is $s_c = s_{LF} + Q_2 w_2$.

The solution of (4.14) is basically the same except that s_{DL} is not necessarily orthogonal to the null space. Since we need this orthogonality so that the trust region constraint will separate as in (4.25), we compute the portion of s_{DL} that is orthogonal to the null space, i.e.,

$$w_1^{DL} = Q_1{}^T s_{DL} . \tag{4.26}$$

Then, the solution can be written as $s = Q_1 w_1^{DL} + Q_2 w_2$. Substituting this back into subproblem (4.14) yields an unconstrained trust region subproblem which can then be solved for w_2, completing the calculation of the trial step in this instance.

In our preliminary implementation, we use the subroutine GQTPAR from the MINPACK project to solve these unconstrained trust region subproblems for our trial step. GQTPAR is based on the algorithm given in Moré and Sorensen. [1983].

4.4 Statement of the Algorithm

In this section, we will summarize our strategy for calculating a trial step. First, if $h(x_c) = 0$ or if h_c is orthogonal to the column space of ∇h_c^T, then the linearized constraint manifold contains $s = 0$. Since the feasible region for the QPTR subproblem is guaranteed to be non-empty in these circumstances, we will use this subproblem to determine a trial step. Otherwise, we will solve problem GQP for either a solution s_{QP} or a descent direction of zero or negative curvature inside the null space of ∇h_c^T. The step s_{LF} to the linearized constraints will be a by-product of the solution of problem GQP. If s_{QP} exists and is inside the trust region, then we will take it as our trial step.

If we did not take s_{QP} as our trial step, we choose a globablization strategy based on the distance to linear feasibility, $\|s_{LF}\|$. If s_{LF} is inside the "inner" trust region of radius $.8\Delta_c$, then we use the QPTR subproblem to calculate our trial step. Otherwise, we will use the 2DCTR subproblem unless we encounter one of the situations in which the constraint $\|\nabla h_c^T s + h_c - \Theta_{MIN}\| \leq \theta_c$ may be ill-conditioned, and we will handle these cases as discussed in Section 4.2.

Algorithm Trialstep:

1. **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, Δ_c, **calculate** a trial step s_c.

2. **If** $(\nabla h_c h_c = 0)$, **then**

   (a) Solve:

$$\begin{array}{ll} \text{minimize} & q_c(s) \\ \text{subject to} & \nabla h_c{}^T s = 0 \\ & \|s\| \leq \Delta_c \end{array}$$

   for s_c.

   (b) Return.

End if.

3. (a) Solve **Problem GQP:**

$$\begin{array}{ll} \text{minimize} & q_c(s) \\ \text{subject to} & \nabla h_c{}^T s + h_c = \Theta_{MIN} \end{array}$$

for

- s_{LF}, the step to the linearized constraints, $\nabla h_c{}^T s + h_c = \Theta_{MIN}$
- s_{QP}, if such a solution exists, or
- d_{QP} a descent direction of zero or negative curvature inside the null space of ∇h_c^T, otherwise.

(b) **If** (a solution to problem GQP exists), **then**

- solution = $true$

Else

- solution = $false$

End if.

4. **If** ((solution = $true$) **and** ($\|s_{QP}\| \leq \Delta_c$)), **then**

(a) $s_c = s_{QP}$

(b) Return.

End if.

5. Calculate the required Linear Feasibility.

(a) **If** ($\|s_{LF}\| \leq .8\Delta_c$), **then**

- $s_{DL} = s_{LF}$

Else

- Calculate the Cauchy point for the constraints, s_{CP}.
- Find α such that $\|s_{CP} + \alpha s_{LF}\| = .8\Delta_c$
- $s_{DL} = s_{CP} + \alpha s_{LF}$

End if

(b) $\theta_c = \|\nabla {h_c}^T s_{DL} + h_c - \Theta_{MIN}\|$

6. **If** $((\theta_c$ is too small$)$, **or** $(\theta_c \approx \|h_c - \Theta_{MIN}\|))$ **then**

   (a) Solve:

$$\begin{array}{ll} \text{minimize} & q_c(s) \\ \text{subject to} & \nabla {h_c}^T s = \nabla {h_c}^T s_{DL} \\ & \|s\| \leq \Delta_c \end{array}$$

   for s_c.

   (b) Return.

   End if

7. Choose the 2D Subspace.

   - **If** (solution= $true$), **then**
     * $\{v_1, v_2\}$ spans $\{s_{QP}, s_{DL}\}$

     Else
     * $\{v_1, v_2\}$ spans $\{d_{QP}, s_{DL}\}$

     End if.

8. Solve **Problem 2DCTR:**

$$\begin{array}{ll} \text{minimize} & q_c(s) \\ \text{subject to} & \|\nabla h_c^T s + h_c - \Theta_{MIN}\| \leq \theta_c \\ & \|s\| \leq \Delta_c \\ & s \in \text{span}\{v_1, v_2\} \end{array}$$

   for s_c.

9. Return.

10. End.

We point out that this is not the most efficient way to implement the algorithm since there will be occasions when problem GQP is solved unnecessarily. For example, suppose we solve problem GQP for s_{QP} only to discover that it is outside of the trust region, and in addition, suppose that $\|s_{LF}\| \leq .8\Delta_c$. Then we would use the QPTR subproblem to compute the trial step, and the solution of problem GQP would have been unnecessary. In our preliminary implementation, however, we are interested more in stability than speed, and we view the solution of problem GQP as a diagnostic tool. In addition, we will use the curvature information we obtain from problem GQP in choosing Lagrange multiplier estimates. Future implementations will address the issue of efficiency.

Chapter 5

Solution of the 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 θ_c, we are ready to discuss the solution of the 2DCTR subproblem. Recall that it is

2DCTR Subproblem: (5.1)

$$\begin{aligned} \text{minimize} \quad & \nabla_x {l_c}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \|\nabla {h_c}^T s + h_c\|_2 \le \theta_c \\ & \|s\|_2 \le \Delta_c \\ & s \in \text{span}\{s_{QP} \text{ or } d_{QP}, s_{DL}\}. \end{aligned} \tag{5.2}$$

This subproblem consists of the minimization of a non-convex quadratic subject to two quadratic constraints in two dimensions. Recently, Dennis, Martinez and Williamson [1991] 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 5.1 Dennis, Martinez, and Williamson [1991].

If s_c is a global solution of the CDT subproblem,

Problem CDT:

$$\begin{aligned} \text{minimize} \quad & \nabla_x {l_c}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \|\nabla {h_c}^T s + h_c\| \le \theta_c \\ & \|s\| \le \Delta_c\,, \end{aligned}$$

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: (5.3)

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \|s\| \le \Delta_c , \end{aligned}$$

or

Subproblem LF: (5.4)

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s \\ \text{subject to} \quad & \|\nabla h_c{}^T s + h_c\| \le \theta_c . \end{aligned}$$

If any global solution of either (5.3) or (5.4) is feasible for both constraints. then it is a global solution of problem CDT.

Theorem 5.1 will be our guide in developing an algorithm to solve the 2DCTR subproblem. Theorem 5.1 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 (5.3) and LF (5.4). The subproblem TR (5.3) is obviously the standard unconstrained trust region subproblem. The subproblem LF (5.4) 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 5.1 tells us that the global solution to problem 2DCTR may be a local, *non-global* solution to one of the unconstrained subproblems TR (5.3) and LF (5.4). 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 (5.3), and this work will be described in Chapter 6. For now, we will assume that we can obtain all of the solutions to the subproblems (5.3) and (5.4).

Using Theorem 5.1 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 (5.3).

2. If any global solution to subproblem TR satisfies $\|\nabla h_c^T s + h_c\| \le \theta_c$, then it is a solution to problem 2DCTR.

3. Find all local solutions to subproblem LF given in (5.4).

4. If any global solution to subproblem LF is inside the trust region, then it is a 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 (5.3), if it exists and satisfies $\|\nabla h_c^T s + h_c\| \leq \theta_c$.

   (c) The non-global solution to the subproblem in (5.4), 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 $\|\nabla h_c^T s + h_c\| \leq \theta_c$ 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

$$q(\hat{s} + \alpha d_{QP}) \;\; = \;\; q(\hat{s}) + \alpha(\nabla_x l^T d_{QP} + \hat{s}^T B d_{QP}) + \frac{1}{2}\alpha^2 d_{QP}^T B d_{QP}\,.$$

If we choose the sign of α such that $\alpha(\nabla_x l^T d_{QP} + s_{LF}^T B d_{QP}) \leq 0$, 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 $\nabla {h_c}^T$.

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

$$q(\hat{s}) \leq q(s) \text{ for } \{s : \|s\| = \Delta_c \text{ and } \|\nabla h_c^T s + h_c\|_2 \leq \theta_c\} \tag{5.5}$$

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, (5.5) 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 follows. 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 6. If a global solution to problem TR satisfies the linear feasibility constraint, then we take it as the solution. Otherwise, if second-order sufficiency holds, we convert problem LF to the standard unconstrained trust region form and use the techniques of Chapter 6 to solve it. If a global solution to problem LF is inside the trust region, then it will be a solution to Problem 2DCTR. 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.

Solution of Problem 2DCTR:

1. **Given** $h_c, \nabla h_c$, **the quadratic model,** $q_c(s) = \nabla_x l_c^T s + \frac{1}{2} s^T B_c s$, **the trust region radius,** Δ_c, **and the two-dimensional subspace** $\{v_1, v_2\}$, **calculate a solution,** s_c, **to problem 2DCTR.**

2. Solve **Problem TR:**

$$\begin{array}{ll} \text{minimize} & q_c(s) \\ \text{subject to} & \|s\| \leq \Delta_c \\ & s \in \text{span}\{v_1, v_2\} \end{array}$$

for all global solutions, s_{TRG}, **and the local, non-global solution,** s_{TRL}, **if it exists.**

3. **If** (any s_{TRG} satisfies $\|\nabla {h_c}^T s_{TRG} + h_c - \Theta_{MIN}\| \leq \theta_c$), **then**

   (a) $s_c = s_{TRG}$

   (b) Return

   End if.

4. **If** (Second-order Sufficiency holds), **then**

   (a) Solve **Problem LF**:

$$\begin{array}{ll} \text{minimize} & q_c(s) \\ \text{subject to} & \|\nabla h_c^T s + h_c - \Theta_{MIN}\| \leq \theta_c \\ & s \in \text{span}\{v_1, v_2\} \end{array}$$

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

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

   - $s_c = s_{LFG}$
   - Return

   End if.

5. Calculate the points where both of the two-dimensional constraints are binding, i. e., find s_{CB1}, s_{CB2}, s_{CB3}, and s_{CB4} such that

$$\begin{array}{l} \|\nabla {h_c}^T s + h_c\| = \theta_c \\ \|s\| = \Delta_c \\ s \in \text{span}\{v_1, v_2\} \end{array}$$

   There may be 2, 3 or 4 intersection points.

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

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

   - save s_{TRL}

End if.

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

- save s_{LFL}

End if.

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

8. Return.

9. End.

5.1 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

2DCTR Subproblem:

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

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

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

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 $z \in I\!R^2$ will be our new variable in the two-dimensional subspace. Then, writing subproblem 2DCTR with the new variables yields

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

where

$$\begin{aligned} q_{2D}(z) &= \nabla_x l_{2D}^T z + \frac{1}{2} z^T B_{2D} z. & (5.6)\\ \nabla_x l_{2D} &= V^T \nabla_x l_c, & (5.7)\\ B_{2D} &= V^T B_c V, & (5.8)\\ \text{and } \nabla h_{2D} &= V^T \nabla h_c. & (5.9) \end{aligned}$$

Then. $\nabla_x l_{2D} \in I\!R^2$, $B_{2D} \in I\!R^{2\times 2}$, and $\nabla h_{2D} \in I\!R^{2\times m}$.

5.2 Conversion of Problem LF into Standard Trust Region Form

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

$$\begin{aligned} \text{minimize} \quad & \nabla_x l_c{}^T s + \frac{1}{2} s^T B_c s & (5.10)\\ \text{subject to} \quad & \|\nabla h_c{}^T s + h_c\|_2 \le \theta_c & (5.11)\\ & s \in \text{span}\{s_{QP}, s_{DL}\}, \end{aligned}$$

into the standard trust region form:

$$\begin{aligned} \text{minimize} \quad & q_{LF}(y) \\ & \|y\| \le \theta_c. \end{aligned}$$

Recall that if we need to solve this subproblem, then we know that s_{QP} exists and satisfies $(-\nabla h_c{}^T s_{QP} = h_c - \Theta_{MIN})$. Using this relation, (5.11) becomes

$$\|\nabla h_c{}^T (s - s_{QP})\| \le \theta_c.$$

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 $\|\nabla h_{2D}{}^T (z - z_{QP})\| \le \theta_c$ where ∇h_{2D} was defined in the previous section.

We now replace $(z - z_{QP})$ with Uy where $U \in \mathbb{R}^{2\times 2}, y \in \mathbb{R}^2$, 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

$$\begin{aligned} \nabla_x l_{LF} &= (\nabla_x l_{2D}{}^T U + z_{QP} B_{2D} U)^T && (5.12) \\ \text{and } B_{LF} &= U^T B_{2D} U. && (5.13) \end{aligned}$$

Now problem LF has been transformed into a standard unconstrained trust region subproblem. We remark, however, that if the columns of ∇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 (5.11) will not be a solid ellipse but instead will be the region between two parallel lines.

Chapter 6

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 5.1 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

Problem UTR: (6.1)

$$\begin{array}{ll} \text{minimize} & g^T s + \frac{1}{2} s^T H s \\ \text{subject to} & \|s\| \leq \Delta , \end{array}$$

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 5.1 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 (6.1) 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 (2×2) 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.

6.1 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 $g \in I\!R^2$, $H \in I\!R^{2\times 2}$, 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

$$Q^T H Q = \Lambda \equiv \mathrm{diag}(\Lambda_1, \Lambda_2) \tag{6.2}$$

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 Λ_1 by v_1 and the eigenvector corresponding to Λ_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 6.1 Sorensen [1982], Gay [1981].

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

$$(H + \mu^* I)s^* = -g \tag{6.3}$$

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 6.1 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 6.1 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 6.1 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 6.1 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 a global minimizer.

Lemma 6.2 Sorensen [1982].

Let μ and s satisfy

$$(H + \mu I)s = -g \text{ with } (H + \mu I) \text{ positive semidefinite.} \tag{6.4}$$

(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 $\psi(w) = g^T w + \frac{1}{2} w^T H w$.

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

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

Since Lemmas 6.1 and 6.2 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 6.1 Second-order Sufficiency.

If there exists a vector μ_* such that

$$(H + \mu^* I)s^* = -g \tag{6.5}$$

$$\|s^*\| \leq \Delta \tag{6.6}$$

$$\mu^*(\Delta - \|s^*\|) = 0 \tag{6.7}$$

$$\mu^* \geq 0, \tag{6.8}$$

and for every $z \neq 0$ satisfying

$$z^T s^* \geq 0 \quad \text{if} \quad \|s^*\| = \Delta \text{ and } \mu^* = 0 \tag{6.9}$$

$$z^T s^* = 0 \quad \text{if} \quad \|s^*\| = \Delta \text{ and } \mu^* > 0, \tag{6.10}$$

it follows that

$$z^T(H + \mu^* I)z > 0, \tag{6.11}$$

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.

6.2 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 6.2 Solutions to Problem UTR when $g = 0$.

Given $g \in I\!R^2$, $H \in I\!R^{2\times 2}$ with H symmetric, and $\Delta > 0$, let $\Lambda_1 \leq \Lambda_2$ be the eigenvalues and $\{v_1, v_2\}$ be 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

$$\begin{pmatrix} \Lambda_1 + \mu^* & 0 \\ 0 & \Lambda_2 + \mu^* \end{pmatrix} \begin{pmatrix} v_1^T s^* \\ v_2^T s^* \end{pmatrix} = 0. \tag{6.12}$$

1. $(\Lambda_1 > 0)$.

   Since $\Lambda_1 > 0$ and $\Lambda_1 \leq \Lambda_2$, the only solution to (6.12) with $\mu^* \geq 0$ is $s^* = 0$. Complementarity then requires $\mu^* = 0$, and $(H + \mu^* I)$ is positive definite. Thus, from Lemma 6.2, problem UTR has a single global solution $s^* = 0$ with $\mu^* = 0$.

2. $(\Lambda_1 = 0) \wedge (\Lambda_2 = 0)$.

   In this case, (6.12) reduces to the two equations

   $$(\mu^*) v_1^T s^* = 0 \tag{6.13}$$

   $$(\Lambda_2 + \mu^*) v_2^T s^* = 0. \tag{6.14}$$

   Since $(\Lambda_2 + \mu^*) > 0$ for all $\mu^* \geq 0$, equation (6.14) 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 two, s^* must be of the form av_1 for some constant a. Substituting $s^* = av_1$ into equation (6.13) shows that $\mu^* = 0$. With this μ^*, $(H + \mu^* I)$ is positive semidefinite. Then, Lemma 6.2(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 (6.12) reduces to

   $$(\mu^*) v_1^T s^* = 0$$

   $$(\mu^*) v_2^T s^* = 0.$$

   First consider $\mu^* = 0$. With this μ^*, $(H + \mu^* I)$ is positive semidefinite, and $(H + \mu^* I)s^* = -g$ is satisfied for all s. Application of Lemma 6.2(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. $(\Lambda_1 < 0) \wedge (\Lambda_2 > 0)$.

   Equation (6.12) 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 6.2(iii) yields two global solutions $s^* = \Delta v_1$ and $s^* = -\Delta v_1$ with $\mu^* = -\Lambda_1$.

Although equation (6.12) is satisfied with $\mu = -\Lambda_2$, this cannot correspond to a minimizer since $\mu < 0$.

Equation (6.12) is also satisfied with $s = 0$, and complementarity would require $\mu = 0$. Since $(H + \mu I)$ is indefinite, Lemma 6.1 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 ε is small so that εv_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. $(\Lambda_1 < 0) \wedge (\Lambda_2 = 0)$.

   Equation (6.12) reduces to

$$
\begin{aligned}
(\Lambda_1 + \mu^*)v_1^T s^* &= 0 \\
(\mu^*)v_2^T s^* &= 0,
\end{aligned}
$$

   and is satisfied by $\mu^* = -\Lambda_1 > 0$ and $s^* = av_1$ for some constant a. $(H + \mu^* I)$ is positive semidefinite. Then, Lemma 6.2(iii) yields two global solutions of the form $s^* = \Delta v_1$ and $s^* = -\Delta v_1$.

   Equation (6.12) 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 (6.12) 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 (6.12) 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 6.2(iii) yields two global solutions $s^* = \Delta v_1$ and $s^* = -\Delta v_1$ with $\mu^* = -\Lambda_1$.

   Equation (6.12) 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 6.2(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 (6.12), 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 (6.12) is satisfied for $\mu^* = -\Lambda_1 > 0$ and any s, and $(H + \mu^* I)$ is positive semidefinite. Application of Lemma 6.2(iii) yields an infinite number of global solutions of the form $s^* = \{s : \|s\| = \Delta\}$.

   The only other step which satisfies (6.12) 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 6.3 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 corresponding orthonormal eigenvectors of H. Assume that $\|g\| \neq 0$. Let

$$\mu_+ = -\Lambda_1 + \frac{\|g\|}{\Delta}. \tag{6.15}$$

If $(\Lambda_1 = \Lambda_2)$ and $(\mu_+ \geq 0)$ where μ_+ is given by (6.15), 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 μ_+ is given by (6.15), 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

$$s = -\frac{g}{(\Lambda_1 + \mu)}, \tag{6.16}$$

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

$$\mu_+ = -\Lambda_1 + \frac{\|g\|}{\Delta} \tag{6.17}$$

$$\mu_- = -\Lambda_1 - \frac{\|g\|}{\Delta}. \tag{6.18}$$

First we will consider whether or not μ_+ corresponds to a minimizer.

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

   Since $\mu_+ \geq 0$, μ_+ can correspond to a minimizer, and $\mu^* = \mu_+$. From (6.16), the step must be of the form

   $$s^* = -\frac{\Delta}{\|g\|}g,$$

   and $(H + \mu^* I)$ is positive definite. Lemma 6.2 verifies that s^* is the unique global minimizer in this case.

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

   In this case, μ_+ cannot correspond to a minimizer. However, we can show that H is positive definite since $\mu_+ < 0$ implies $\Lambda_1 > 0$. With $\mu^* = 0$, (6.16) 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 μ_- 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 μ_- cannot correspond to a minimizer. $\square$

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 6.4 Solutions to Problem UTR when $(v_1^T g = 0)$.

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

$$\mu_+ = -\Lambda_2 + \frac{|c_2|}{\Delta} \tag{6.19}$$

If $(v_1^T g = 0)$ and $(\Lambda_1 > 0) \wedge (\mu_+ \leq 0)$ where μ_+ is given by (6.19), 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 $(\Lambda_1 > 0) \wedge (\mu_+ > 0)$ where μ_+ is given by (6.19), then problem UTR has one global solution of the form $s^* = -\text{sign}(c_2)\Delta v_2$ with multiplier $\mu^*_\cdot = \mu_+$.

If $(v_1^T g = 0)$ and $(\Lambda_1 \leq 0) \wedge (\mu_+ > -\Lambda_1)$ where μ_+ is given by (6.19), 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 $(\Lambda_1 \leq 0) \wedge (\mu_+ = -\Lambda_1)$, where μ_+ is given by (6.19), then problem UTR has one global solution of the form

$$s^* = -\left(\frac{c_2}{\Lambda_2 - \Lambda_1}\right) v_2. \tag{6.20}$$

If $(v_1^T g = 0)$ and $(\Lambda_1 < 0) \wedge (\mu_+ < -\Lambda_1)$ where μ_+ is given by (6.19), then problem UTR has two global solutions of the form

$$s^* = -\left(\frac{c_2}{\Lambda_2 - \Lambda_1}\right) v_2 + \tau v_1 \tag{6.21}$$

$$\text{and } s^* = -\left(\frac{c_2}{\Lambda_2 - \Lambda_1}\right) v_2 - \tau v_1 \tag{6.22}$$

where

$$\tau = \left(\Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2}\right)^{\frac{1}{2}}. \tag{6.23}$$

If ($v_1^T g = 0$ and $(\Lambda_1 = 0) \wedge (\mu_+ < -\Lambda_1)$ where μ_+ is given by (6.19), 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 τ is given by equation (6.23).

Proof First we will show that $\mu = -\Lambda_2$ cannot correspond to a minimizer. For this case, $(H + \mu I)s = -g$ reduces to

$$(\Lambda_1 + \mu)v_1^T s^* = 0 \tag{6.24}$$

$$(\Lambda_2 + \mu)v_2^T s^* = -v_2^T g. \tag{6.25}$$

For $\mu = -\Lambda_2$, there is no finite s such that equation (6.25) can be satisfied since $v_2^T g \neq 0$.

For $\mu \neq -\Lambda_2$, equations (6.24) and (6.25) require

$$s = -\left(\frac{v_2^T g}{\Lambda_2 + \mu}\right) v_2, \tag{6.26}$$

since v_1 is orthogonal to g. Unless $\mu^* = 0$, complementarity requires $\|s^*\| = \Delta$. There are only two choices of μ that satisfy the complementarity condition and (6.26), and they are

$$\mu_+ = -\Lambda_2 + \frac{|c_2|}{\Delta}, \tag{6.27}$$

$$\mu_- = -\Lambda_2 - \frac{|c_2|}{\Delta}. \tag{6.28}$$

First we will show that μ_- cannot correspond to a minimizer. For μ_-,

$$(H + \mu_- I) = Q \begin{pmatrix} \Lambda_1 - \Lambda_2 - \frac{|c_2|}{\Delta} & 0 \\ 0 & -\frac{|c_2|}{\Delta} \end{pmatrix} Q^T. \tag{6.29}$$

Since $\Lambda_1 < \Lambda_2$, $(H + \mu_- I)$ is negative definite, and μ_- could only correspond to maximizers.

Now consider μ_+.

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 (6.26), 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 μ_+, ($\mu_+ > -\Lambda_1$) 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

   $$(H - \Lambda_1 I)p = -g \tag{6.30}$$

   and τ 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 (6.30) reduces to $(\Lambda_2 - \Lambda_1)v_1^T s = -v_1^T g$, and this gives

$$p = -\frac{c_2}{\Lambda_2 - \Lambda_1} v_2. \tag{6.31}$$

There are two values of τ that satisfy $\|p + \tau v_1\| = \Delta$, and they are

$$\tau_+ = \left(\Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2}\right)^{\frac{1}{2}}, \tag{6.32}$$

$$\tau_- = -\left(\Delta^2 - \frac{c_2^2}{(\Lambda_2 - \Lambda_1)^2}\right)^{\frac{1}{2}}. \tag{6.33}$$

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 τ given in (6.32) and (6.33) 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 (6.31) and τ_+ and τ_- are given by (6.32) and (6.33). We can show that they have the same value of the quadratic since

$$q(p + \tau v_1) = a v_2^T g + \frac{1}{2}(\Lambda_1 \tau^2 + \Lambda_2 a^2) \text{ with } a = -\frac{c_2}{\Lambda_2 - \Lambda_1}$$

does not depend on the sign of τ.

iii. $(\Lambda_1 = 0)$.
In this case, the quadratic reduces to

$$q(p + \tau v_1) = a v_2^T g + \frac{1}{2}\Lambda_2 a^2,$$

and it does not depend on τ at all. Thus, every step of the form $p + \tau v_1$ for all τ 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 (6.31) and τ_+ is given by (6.32).

□

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 6.5 Solutions to Problem UTR when $(v_2^T g = 0)$.

Given $g \in I\!R^2$, $H \in I\!R^{2\times 2}$ with H symmetric, and $\Delta > 0$, let $\Lambda_1 \leq \Lambda_2$ be the eigenvalues and $\{v_1, v_2\}$ be 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

$$\mu_- = -\Lambda_1 - \frac{|c_1|}{\Delta} \quad \text{and} \quad \mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta}. \tag{6.34}$$

If $(v_2^T g = 0)$ and $(\Lambda_1 \geq 0) \wedge (\mu_+ \leq 0)$, 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 $(v_2^T g = 0)$ and $(\mu_+ > 0)$ where μ_+ is given by (6.34), then problem UTR has one global solution of the form $s^* = -\text{sign}(c_1)\Delta v_1$ with multiplier $\mu^* = \mu_+$ given in (6.34).

If $(v_2^T g = 0)$ and $(\Lambda_1 < 0) \wedge (\mu_- > 0) \wedge (\mu_- > -\Lambda_2)$ where μ_- is given in (6.34), 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

$$(\Lambda_1 + \mu)v_1^T s^* = -v_1^T g \tag{6.35}$$

$$(\Lambda_2 + \mu)v_2^T s^* = 0. \tag{6.36}$$

For $\mu = -\Lambda_1$, there is no finite s such that equation (6.35) can be satisfied since $v_1^T g \neq 0$.

For $\mu \neq -\Lambda_1$, equations (6.35) and (6.36) require

$$s = -\left(\frac{v_1^T g}{\Lambda_1 + \mu}\right) v_1, \tag{6.37}$$

since v_2 is orthogonal to g. Unless $\mu^* = 0$, complementarity requires $\|s^*\| = \Delta$. There are only two choices of μ that satisfy the complementarity condition and (6.37), and they are

$$\mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta}, \tag{6.38}$$

$$\mu_- = -\Lambda_1 - \frac{|c_1|}{\Delta}. \tag{6.39}$$

Now we will consider if μ_+ and μ_- 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

   $$\frac{|c_1|}{\Delta} \leq \Lambda_1. \tag{6.40}$$

   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 μ_+ into equation (6.37) 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 μ_-.

Since $\Lambda_1 \geq 0$, $\mu_- < 0$, and so μ_- cannot correspond to a minimizer.

2. ($\Lambda_1 < 0$).

(a) Consider μ_+.

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 μ_-.

Clearly, we are only interested in $\mu_- \geq 0$. Using μ_- in equation (6.37) yields $s = \text{sign}(c_1)\Delta v_1$.

i. Suppose $\mu_- > 0$.

Then. for Theorem 6.1, all $z \neq 0$ satisfying $z^T s = 0$ can be written as $z = av_2$ for all $a \neq 0$. Then,

$$z^T(H + \mu_- I)z = a^2(\Lambda_2 + \mu_-). \tag{6.41}$$

Thus, from Theorem 6.1, 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 6.1, 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

$$q(\hat{s}) = |c_1|\Delta + \frac{1}{2}\Delta^2\Lambda_1. \tag{6.42}$$

Now consider a step $\tilde{s}$ which is slightly inside the trust region from $\hat{s}$ along v_1 of the form

$$\tilde{s} = \text{sign}(c_1)(\Delta - \varepsilon)v_1 \tag{6.43}$$

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

$$q(\tilde{s}) = |c_1|(\Delta - \varepsilon) + \frac{1}{2}(\Delta - \varepsilon)^2\Lambda_1. \tag{6.44}$$

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

$$\begin{aligned} q(\tilde{s}) - q(\hat{s}) &= -|c_1|\varepsilon + \frac{1}{2}\Lambda_1\left((\Delta - \varepsilon)^2 - \Delta^2\right) && (6.45) \\ &= -|c_1|\varepsilon + \Lambda_1\Delta\varepsilon + \frac{1}{2}\Lambda_1\varepsilon^2. && (6.46) \end{aligned}$$

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

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

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 are 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.

6.3 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 μ^* 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

$$(H + \mu I)s = -g \text{ such that } \|s\| = \Delta \text{ and } \mu \in (-\Lambda_1, \infty). \tag{6.47}$$

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

$$s(\mu) = -Q(\Lambda + \mu I)^{-1} Q^T g. \tag{6.48}$$

With the notation $c_1 = v_1^T g$ and $c_2 = v_2^T g$, (6.48) becomes

$$s(\mu) = -\left(\frac{c_1}{\Lambda_1 + \mu}\right) v_1 - \left(\frac{c_2}{\Lambda_2 + \mu}\right) v_2. \tag{6.49}$$

Adding the trust region constraint yields

$$\|s(\mu)\|^2 = \frac{c_1^2}{(\Lambda_1 + \mu)^2} + \frac{c_2^2}{(\Lambda_2 + \mu)^2} = \Delta^2. \tag{6.50}$$

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

$$\frac{1}{\Delta} - \frac{1}{\|(H + \mu I)^{-1} g\|_2} = 0. \tag{6.51}$$

Newton's method is very efficient when applied to (6.51) 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

$$\phi(\mu) = \frac{1}{\|(H+\mu I)^{-1}g\|_2} - \frac{1}{\Delta} = 0, \tag{6.52}$$

and we will consider applying Newton's method to it. Given a starting point μ_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 6.3 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 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

$$\phi(\mu) = \frac{1}{\|(H+\mu I)^{-1}g\|_2} - \frac{1}{\Delta}. \tag{6.53}$$

Then,

$$\lim_{\mu \to -\Lambda_1^-} \phi'(\mu) = -\frac{1}{|c_1|} \tag{6.54}$$

and,

$$\lim_{\mu \to -\Lambda_1^+} \phi'(\mu) = \frac{1}{|c_1|}. \tag{6.55}$$

Proof First,

$$\phi(\mu) = \left(\frac{c_1^2}{(\Lambda_1+\mu)^2} + \frac{c_2^2}{(\Lambda_2+\mu)^2}\right)^{-\frac{1}{2}} - \frac{1}{\Delta}, \tag{6.56}$$

and

$$\phi'(\mu) = \left(\frac{c_1^2}{(\Lambda_1+\mu)^3} + \frac{c_2^2}{(\Lambda_2+\mu)^3}\right)\left(\frac{c_1^2}{(\Lambda_1+\mu)^2} + \frac{c_2^2}{(\Lambda_2+\mu)^2}\right)^{-\frac{3}{2}}. \tag{6.57}$$

Since

$$\lim_{\mu \to -\Lambda_1^-} \phi'(\mu) = \lim_{\varepsilon \to 0^-} \phi'(-\Lambda_1 + \varepsilon),$$

we will consider $\phi'(-\Lambda_1 + \varepsilon)$.

$$\begin{aligned}
\phi'(-\Lambda_1+\varepsilon) &= \left(\frac{c_1^2}{\varepsilon^3} + \frac{c_2^2}{(\Lambda_2-\Lambda_1+\varepsilon)^3}\right)\left(\frac{c_1^2}{\varepsilon^2} + \frac{c_2^2}{(\Lambda_2-\Lambda_1+\varepsilon)^2}\right)^{-\frac{3}{2}} \\
&= \left(\frac{c_1^2(\Lambda_2-\Lambda_1+\varepsilon)^3 + c_2^2\varepsilon^3}{\varepsilon^3(\Lambda_2-\Lambda_1+\varepsilon)^3}\right)\left(\frac{c_1^2(\Lambda_2-\Lambda_1+\varepsilon)^2 + c_2^2\varepsilon^2}{\varepsilon^2(\Lambda_2-\Lambda_1+\varepsilon)^2}\right)^{-\frac{3}{2}} \\
&= \operatorname{sign}(\varepsilon)\ \operatorname{sign}(\Lambda_2-\Lambda_1+\varepsilon)\left(\frac{c_1^2(\Lambda_2-\Lambda_1+\varepsilon)^3 + c_2^2\varepsilon^3}{c_1^2(\Lambda_2-\Lambda_1+\varepsilon)^2 + c_2^2\varepsilon^2}\right)^{\frac{3}{2}}. \qquad (6.58)
\end{aligned}$$

Thus,

$$\lim_{\varepsilon\to 0^-}\phi'(-\Lambda_1+\varepsilon) = (-1)\ \frac{c_1^2(\Lambda_2-\Lambda_1)^3}{(c_1^2(\Lambda_2-\Lambda_1)^2)^{\frac{3}{2}}},$$

and so,

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

Similarly, from (6.58),

$$\lim_{\varepsilon\to 0^+}\phi'(-\Lambda_1+\varepsilon) = \frac{c_1^2(\Lambda_2-\Lambda_1)^3}{(c_1^2(\Lambda_2-\Lambda_1)^2)^{\frac{3}{2}}},$$

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

$$\lim_{\mu\to -\Lambda_2^-}\phi'(\mu) = -\frac{1}{|c_2|} \qquad (6.59)$$

and,

$$\lim_{\mu\to -\Lambda_2^+}\phi'(\mu) = \frac{1}{|c_2|}. \qquad (6.60)$$

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

Lemma 6.4 Given $g \in I\!R^2$, $H \in I\!R^{2\times 2}$ where H is symmetric, and $\Delta > 0$, let $\Lambda_1 \le \Lambda_2$ denote the eigenvalues of H, and let v_1 and v_2 denote corresponding orthonormal eigenvectors. Let

$$\bar{\phi}(\mu) = \frac{1}{\|(H + \mu I)^{-1} g\|_2}. \tag{6.61}$$

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 $\bar{\phi}(\mu)$ as $\mu \to -\Lambda_1^-$. Then,

$$\bar{\phi}(\mu) \leq l^-(\mu) \text{ for } \mu \in (-\infty, -\Lambda_1]. \tag{6.62}$$

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

$$\bar{\phi}(\mu) \leq l^+(\mu) \text{ for } \mu \in [-\Lambda_1, \infty). \tag{6.63}$$

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

$$l^-(\mu) = -\frac{1}{|c_1|}\,(\mu + \Lambda_1). \tag{6.64}$$

Consider $\bar{\phi}(\mu)^2$:

$$\begin{aligned}
\bar{\phi}(\mu)^2 &= \left(\frac{c_1^2}{(\Lambda_1+\mu)^2} + \frac{c_2^2}{(\Lambda_2+\mu)^2}\right)^{-1} \\
&= (\Lambda_1+\mu)^2 \left(\frac{(\Lambda_2+\mu)^2}{c_1^2(\Lambda_2+\mu)^2 + c_2^2(\Lambda_1+\mu)^2}\right) \\
&= (\Lambda_1+\mu)^2 \left(c_1^2 + c_2^2\left(\frac{\Lambda_1+\mu}{\Lambda_2+\mu}\right)^2\right)^{-1}.
\end{aligned}$$

Thus,

$$\bar{\phi}(\mu)^2 \leq (\Lambda_1+\mu)^2 \left(\frac{1}{c_1^2}\right),$$

and so,

$$\bar{\phi}(\mu) \leq \frac{|\Lambda_1+\mu|}{|c_1|}. \tag{6.65}$$

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

$$\bar{\phi}(\mu) \leq -\frac{(\Lambda_1+\mu)}{|c_1|}. \tag{6.66}$$

Thus,

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

for $\mu \in (-\infty, -\Lambda_1]$.

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

$$l^+(\mu) = \frac{1}{|c_1|}\ (\mu + \Lambda_1). \tag{6.67}$$

For $\mu \in [-\Lambda_1, \infty)$, $(\mu + \Lambda_1) \geq 0$, and from (6.65),

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

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

$$\mu_+ = -\Lambda_1 + \frac{|c_1|}{\Delta}. \tag{6.68}$$

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

$$\mu_0 = \max\{0, \mu_+\} \tag{6.69}$$

as our starting point. From Lemma 6.4, 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 μ_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,

$$\|s_c\| \leq \Delta \text{ and } \mu_c = 0, \tag{6.70}$$

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

$$|\ \Delta - \|s_c\|\ | \leq \sigma\Delta \tag{6.71}$$

for some tolerance σ. Thus, using Newton's method from μ_0 given in (6.69), we can find the global solution in the non-degenerate case.

6.4 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 μ^* 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

$$(H + \mu I)s = -g \text{ such that } \|s\| = \Delta, \mu \in (-\Lambda_2, -\Lambda_1) \text{ and } \mu \geq 0. \tag{6.72}$$

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

Consider the function

$$\varphi(\mu) = \|(H + \mu I)^{-1} g\|^2 \tag{6.73}$$

on the interval $(-\Lambda_2, -\Lambda_1)$. Dennis, Martinez and Williamson [1991] 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 6.6 Given $g \in I\!R^2$, $H \in I\!R^{2\times 2}$ 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 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

$$\mu_0 = \frac{\alpha \Lambda_2 - \Lambda_1}{1 - \alpha} \text{ where } \alpha = -\left(\frac{c_1^2}{c_2^2}\right)^{\frac{1}{3}}. \tag{6.74}$$

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 $\varphi'(\mu) = 0$.

$$\varphi'(\mu) = \frac{c_1^2}{(\Lambda_1+\mu)^3} + \frac{c_2^2}{(\Lambda_2+\mu)^3}. \tag{6.75}$$

So, $\varphi'(\mu) = 0$ is equivalent to

$$\begin{aligned} \frac{(\Lambda_1+\mu)^3}{(\Lambda_2+\mu)^3} &= -\frac{c_1^2}{c_2^2} \\ \frac{(\Lambda_1+\mu)}{(\Lambda_2+\mu)} &= -\left(\frac{c_1^2}{c_2^2}\right)^{\frac{1}{3}}. \end{aligned} \tag{6.76}$$

Let

$$a = -\left(\frac{c_1^2}{c_2^2}\right)^{\frac{1}{3}}. \tag{6.77}$$

Then, equation (6.76) becomes $\Lambda_1 + \mu = a(\Lambda_2 + \mu)$, and it is easy to see that

$$\mu_0 = \frac{a\Lambda_2 - \Lambda_1}{1-a} \tag{6.78}$$

where a is given by (6.77). Notice that μ_0 given by (6.78) is well-defined since

$$(1-a) > 1.$$

This follows from (6.77) and the fact that $\Lambda_1 < \Lambda_2$. Clearly μ_0 minimizes $\varphi(\mu)$ since $\varphi''(\mu_0) > 0$.

Thus, we have established that μ_0 is the unique global minimizer of $\varphi(\mu)$ for $\mu \in (-\Lambda_2, -\Lambda_1)$, and consequently,

$$\|(H+\mu_0 I)^{-1}g\|^2 < \|(H+\mu I)^{-1}g\|^2 \text{ for all } \mu \in (-\Lambda_2, -\Lambda_1) \text{ with } \mu \neq \mu_0. \tag{6.79}$$

From this, it is obvious that $\varphi(\mu)$ will intersect the horizontal line Δ^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 6.7

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 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 μ_0 be given by (6.74), and let

$$\mu_- = -\Lambda_1 - \frac{|\, c_1 \,|}{\Delta} \text{ and } \mu_l = \max(0, \mu_0)\,. \tag{6.80}$$

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

$$\Lambda_1 < 0 \text{ and } \|(H+\mu_l I)^{-1}g\|_2 = \|H^{-1}g\| \leq \Delta\,. \tag{6.81}$$

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

$$\Lambda_1 < 0 \text{ and } \|(H+\mu_0 I)^{-1}g\|_2 < \Delta\,. \tag{6.82}$$

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

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

1. ONLY IF: Show μ^* exists implies either condition (6.81) or condition (6.82).

   From Theorem 6.6 and the fact that μ_0 is not a minimizer, we know $\mu^* \in (\mu_0, -\Lambda_1)$. Since μ^* 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 (6.81) and (6.82) imply μ^* exists.

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

$$\|(H+\mu I)^{-1}g\|^2 = \Delta^2. \tag{6.83}$$

From Theorem 6.6 we know that (6.83) will have roots in the interval $(-\Lambda_2, -\Lambda_1)$ if and only if

$$\|(H+\mu_0 I)^{-1}g\|_2 \leq \Delta \tag{6.84}$$

where μ_0 is given by (6.74).

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

Since $(\mu_0 < 0)$ and $\Lambda_1 < 0$, we have $\mu_0 < 0 < -\Lambda_1$. Theorem 6.6, 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 6.6, together with $\|(H+\mu_0 I)^{-1}g\|^2 < \Delta^2$, 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 μ_l and the above arguments, we know $\mu_l \leq \mu^*$. The constant μ_- given in (6.80) is the point where the tangent line $l^-(\mu)$ intersects the trust region constraint. From Lemma 6.4, we know $l^-(\mu) \geq \|(H+\mu I)^{-1}g\|^{-1}$ for $\mu \in (-\lambda_2, -\lambda_1)$, and this implies $\mu^* \leq \mu_-$. Therefore, $\mu^* \in [\mu_l, \mu_-]$, if it exists.

□

From Theorem 6.7, we can use the following logic to determine whether or not a non-global solution exists.

Existence of a Non-global Solution:

If ($\lambda_1 < 0$), **then**

  If ($\mu_0 < 0$), **then**

    If ($\|H^{-1}g\|^2 \leq \Delta^2$), **then**

      A non-global solution exists on (μ_l, μ_-).

    Else

      No non-global solution exists.

    End if

  Else

    If ($\|(H + \mu_0 I)^{-1}g\|^2 < \Delta^2$), **then**

      A non-global solution exists on (μ_l, μ_-).

    Else

      No non-global solution exists.

    End if

  End if

Else

  No non-global solution exists.

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 6.3. 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,

$$|\,\Delta - \|s_c\|\,| \le \sigma\Delta \tag{6.85}$$

for some tolerance σ. Thus, using Newton's method from μ_-, we can find the non-global solution in the non-degenerate case if we have determined that it exists.

6.5 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:

1. **Given** $g \in I\!R^2$, $H \in I\!R^{2\times 2}$ 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.

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

3. **If** $(g = 0)$, **then**

   (a) **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

   $s_g^* = \{s : \|s\| \le \Delta\}$

   End if

Else

If $(\Lambda_1 = \Lambda_2)$, **then**

$s_g^* = \{s : \|s\| = \Delta\}$

Else

$s_g^* = \Delta\, v_1$

$s_g^{**} = -\Delta\, v_1$

$\mu^* = -\Lambda_1$

End if

End if

End if

(b) Return.

End if

4. **If** $(\Lambda_1 = \Lambda_2)$, **then**

(a) $\mu_+ = -\Lambda_1 + (\|g\|/\Delta)$

(b) **If** $(\mu_+ \geq 0)$, **then**

$a = \Delta/\|g\|$

$s_g^* = -a\, g$

$\mu^* = \mu_+$

Else

$s_g^* = -(1/\Lambda_1)\, g$

$\mu^* = 0$

End if

(c) Return.

End if

5. Calculate $c_1 = v_1^T g$ and $c_2 = v_2^T g$.

6. **If** $(v_1^T g = 0)$, **then**

(a) $\mu_+ = -\Lambda_2 + (|\, c_2 \,| /\Delta)$

(b) **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**

$a = -c_2/(\Lambda_2 - \Lambda_1)$

$s_g^* = a v_2$

End if

If ($\mu_+ < -\Lambda_1$), **then**

If ($\Lambda_1 \neq 0$), **then**

$a = -c_2/(\Lambda_2 - \Lambda_1)$

$p = a\, v_2$

$\tau = \sqrt{\Delta^2 - a^2}$

$s_g^* = p + \tau\, v_1$

$s_g^{**} = p - \tau\, v_1$

Else

$b = -c_2/\Lambda_2$

$\tau = \sqrt{\Delta^2 - b^2}$

$s_g^* = b\, v_2 + (2\gamma - 1)\tau v_1$ for $\gamma \in [0, 1]$

End if

End if

End if

(c) Return.

End if

7. **If** $(v_2^T g = 0)$, **then**

(a) $\mu_+ = -\Lambda_1 + (\mid c_1 \mid /\Delta)$

(b) **If** $(\Lambda_1 \geq 0)$, **then**

If $(\mu_+ > 0)$, **then**

$s_g^* = -\text{sign}(c_1)\Delta v_1$

$\mu^* = \mu_+$

Else

$s_g^* = -(1/\Lambda_1)\, g$

$\mu^* = 0$

End if

Else

$s_g^* = -\text{sign}(c_1)\Delta v_1$

$\mu^* = \mu_+$

$\mu_- = -\Lambda_1 - (\mid c_1 \mid /\Delta)$

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

$s_l^* = \text{sign}(c_1)\Delta v_1$

$\mu_l^* = \mu_-$

End if

End if

(c) Return.

End if

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

$\mu_+ = -\Lambda_1 + (\mid c_1 \mid /\Delta)$

$\mu_0 = \max(0, \mu_+)$

$k = 0$

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

(b) Check Convergence Criteria:

If ($(\mid \Delta - \|p\| \mid \leq \sigma\Delta)$ **or** $(\|p\| \leq \Delta$ **and** $\mu_k = 0)$), **then**

$s_g^* = p$

$\mu^* = \mu_k$

GoTo 9.

End if

(c) Take a Newton step:

$\alpha = c_1^2/(\Lambda_1 + \mu_k)^3 + c_2^2/(\Lambda_2 + \mu_k)^3$

$\mu_{k+1} = \mu_k + (\|p\|^2/\alpha)\ ((\Delta - \|p\|)/\Delta)$

(d) $k = k + 1$

(e) GoTo 8a.

9. Determine if there is a local solution with $\mu_l^* \in (-\Lambda_2, -\Lambda_1)$.

If $(\Lambda_1 < 0)$, **then**

$\alpha = -(c_1^2/c_2^2)^{\frac{1}{3}}$

$\mu_0 = (\alpha\Lambda_2 - \Lambda_1)/(1 - \alpha)$

$\mu_l = \max(0, \mu_0)$

$\mu_- = -\Lambda_1 - (\mid c_1 \mid / \Delta)$

If $(\mu_0 < 0)$, **then**

If $(\|H^{-1}g\|^2 \leq \Delta^2)$, **then**

A non-global solution exists on (μ_l, μ_-).

Else

A non-global solution does not exist.

Return.

End if

Else

If $(\|(H + \mu_0 I)^{-1}g\|^2 < \Delta^2)$, **then**

A non-global solution exists on (μ_l, μ_-).

Else

A non-global solution does not exist.

Return.

End if

End if

Else

A non-global solution does not exist.

Return.

End if

10. Iterative Method to determine the local solution with $\mu_l^* \in (\mu_l, \mu_-)$.

    $\mu_0 = \mu_-$

    $k = 0$

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

    (b) Check Convergence Criteria:

        If $(| \Delta - \|p\| | \leq \sigma\Delta)$, **then**

        $s_l^* = p$

        $\mu_l^* = \mu_k$

        Return.

        End if

    (c) Take a Newton step:

        $\alpha = c_1^2/(\Lambda_1 + \mu_k)^3 + c_2^2/(\Lambda_2 + \mu_k)^3$

        $\mu_{k+1} = \mu_k + (\|p\|^2/\alpha)\ ((\Delta - \|p\|)/\Delta)$

    (d) $k = k + 1$

    (e) GoTo 10a.

11. **End.**

6.5.1 Accuracy in the Trust Region Subproblems

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

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

and

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

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

$$|\ \Delta_c - \|s(\mu)\|\ | \leq \sigma_\Delta\, \Delta_c \tag{6.86}$$

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 σ_Δ. In the course of calculating a trial step, there are four situations where we will need to use a test like (6.86). 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 σ_Δ 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

$$\|\nabla {h_c}^T s_{TR} + h_c - \Theta_{MIN}\| \leq (1 + \sigma_\theta)\theta_c \tag{6.87}$$

where $\sigma_\theta = \sigma_\Delta$. However, our strategy for updating the penalty constant requires

$$\|\nabla {h_c}^T s_c + h_c - \Theta_{MIN}\| < \|h_c\| . \tag{6.88}$$

Condition (6.88) can be enforced by choosing σ_θ in (6.87) as

$$\sigma_\theta = \min\left(\sigma_\Delta, \gamma\left(\frac{\|h_c\|}{\theta_c} - 1\right)\right) , \tag{6.89}$$

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

We must also enforce (6.88) 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 σ_θ given by (6.89) 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 (6.86) to determine if this solution also satisfies the trust region constraint.

Chapter 7

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.

7.1 The Choice of a Merit Function

The merit function plays an important rôle 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 ℓ_1 penalty function

$$\phi_1(x) = f(x) + \sum_{i=1}^{m} \rho_i \mid h_i(x) \mid$$

for the merit function. Both of these applications require that the penalty constants (weights) ρ_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 λ and the penalty constant ρ.

Powell and Yuan choose for the multipliers in the augmented Lagrangian the *least squares* multipliers

$$\lambda = -\left(\nabla h(x)^T \nabla h(x)\right)^{-1} \nabla h(x)^T \nabla f(x) \tag{7.1}$$

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 ρ 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:

$$\Delta\lambda_c = -\left(\nabla h_c{}^T \nabla h_c\right)^{-1} \left(B_c s_c + \nabla_x l(x_c, \lambda_c)\right) . \tag{7.2}$$

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 (7.1) and the multipliers given by (7.2), and they can be interpreted as an efficient implementation of the least squares multipliers.

7.2 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 $\widehat{\Delta\lambda}_c$, and we will use the *trial* multiplier $\hat{\lambda}_+ = \lambda_c + \widehat{\Delta\lambda}_c$ 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

$$\nabla h_c \Delta\lambda = -(B_c s_c + \nabla_x l(x_c, \lambda_c)), \tag{7.3}$$

and the resulting $\hat{\lambda}_+$ is then the least squares solution to $\nabla_x l(x_c, \lambda_c + \widehat{\Delta\lambda}_c) = 0$. 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 $\widehat{\Delta\lambda}_M$ 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 $\widehat{\Delta\lambda}_c$ is the SQP multiplier $\Delta\lambda_{QP}$ that we obtained during the solution of problem GQP. If $s_c = 0$, then the multiplier update (7.3) is equivalent to the multiplier update given by (7.1) evaluated at x_c. Thus, the multiplier update (7.3) varies smoothly between the multiplier update given by (7.1) and the QP multipliers.

Numerical experience indicates that using the trial multipliers determined from (7.3) 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 GQP when second-order sufficiency does not hold. When we have a descent direction of negative or zero curvature for the quadratic model inside the null space of ∇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 $\widehat{\Delta\lambda}_c = 0$ when second-order sufficiency does not hold for problem GQP.

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 λ_*. 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 $\widehat{\Delta\lambda}_c = 0$, and the correct multipliers λ_* 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, λ_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 λ_+ to use in the convergence test.

The second multiplier update $\Delta\lambda_c$ is chosen to be the least squares solution to (7.1),

$$\nabla h(x_+)\Delta\lambda = -\left[\nabla f(x_+) + \nabla h(x_+)\left(\lambda_c + \widehat{\Delta\lambda}_c\right)\right], \tag{7.4}$$

and then the new multipliers are

$$\lambda_+ = \lambda_c + \widehat{\Delta\lambda}_c + \Delta\lambda_c. \tag{7.5}$$

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.

Tables 7.1 and 7.2 give 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 7.7 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. Tables 7.1 and 7.2 do not include the test problems for which second-order sufficiency held at every iteration, and for which all versions of the algorithm behaved identically.

The first set of columns in the tables are the results for the versions of the algorithm using a single multiplier update at each iteration, i. e., $\lambda_+ = \hat{\lambda}_+ = \lambda_c + \widehat{\Delta\lambda}_c$. The column labelled $\widehat{\Delta\lambda}_c = \widehat{\Delta\lambda}_M$ always took the multiplier update to be the least squares solution to (7.3). The column labelled $\widehat{\Delta\lambda}_c = 0$ used $\widehat{\Delta\lambda}_M$ except when second-order sufficiency did not hold, and then $\widehat{\Delta\lambda}_c$ was set to 0. In several cases, this version of the algorithm failed to find a solution.

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 $\widehat{\Delta\lambda}_c = 0$ and $\widehat{\Delta\lambda}_c = \widehat{\Delta\lambda}_M$ 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 (7.1), and the new multipliers are $\lambda_+ = \lambda_c + \widehat{\Delta\lambda}_c + \Delta\lambda_c$. These strategies for computing the Lagrange multipliers are detailed in following outline where the choices for the first update are (a, b) and to update a second time or not using the least squares multipliers is determined by choices c or d.

Calculating Lagrange Multiplier Estimates:

1. **Given** λ_c, s_c, $\nabla_x l(x_c, \lambda_c)$, ∇h_c, and B_c, **calculate** λ_+.

2. **If** $(s_c = s_{QP})$, **Then**

   $\widehat{\Delta\lambda}_c = \Delta\lambda_{QP}$

   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 $\widehat{\Delta\lambda}_c$

   Else

   a. $\widehat{\Delta\lambda}_c = 0$, or

   b. $\widehat{\Delta\lambda}_c$ solves $\nabla h_c \Delta\lambda = -(B_c s_c + \nabla_x l(x_c, \lambda_c))$

   End if

   End if

3. $\hat{\lambda}_+ = \lambda_c + \widehat{\Delta\lambda}_c$

4. Evaluate the step.

5. **If** (Step is accepted), **Then**

   c. $\lambda_+ = \hat{\lambda}_+$, or

   d. $\lambda_+ = \hat{\lambda}_+ + \Delta\lambda_c$ where $\Delta\lambda_c$ solves $\nabla h(x_+)\Delta\lambda = -\left[\nabla f(x_+) + \nabla h(x_+)\hat{\lambda}_+\right]$

   End if

Notice from Tables 7.1 and 7.2 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, iii or iv in Table 7.3, 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 only multiplier update 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. Both of the two-step approaches have only four failures but the two single-update strategies each have many more failures. Interestingly, the algorithms did not fail for the same problems, and in fact, at least one of the methods successfully solved each problem listed. To consider efficiency, we can compute the average number of iterations and function evaluations per problem. The average number of iterations per problem is 30.1 for the single update with $\widehat{\Delta\lambda}_c = \widehat{\Delta\lambda}_M$, 25.6 for the two-step multiplier strategy using $\widehat{\Delta\lambda}_c = 0$ and 31.7 for the two-step strategy using $\widehat{\Delta\lambda}_c = \widehat{\Delta\lambda}_M$. Similarly, the average number of function evaluations per problem is 40.1 for the single update with $\widehat{\Delta\lambda}_c = \widehat{\Delta\lambda}_M$, 35.3 for the two-step multiplier strategy using $\widehat{\Delta\lambda}_c = 0$ and 41.7 for the two-step strategy using $\widehat{\Delta\lambda}_c = \widehat{\Delta\lambda}_M$. Given these considerations, we slightly prefer the two-step multiplier strategy using $\widehat{\Delta\lambda}_c = 0$, and we will use it to state the algorithm in the remainder of this work.

Table 7.1 Multiplier Test Results

Problem	Starting Point	# Iterations (# Function Evaluations)			
		$\lambda_+ = \lambda_c + \widehat{\Delta\lambda_c}$		$\lambda_+ = \lambda_c + \widehat{\Delta\lambda_c} + \Delta\lambda_c$	
		$\widehat{\Delta\lambda_c} = 0$	$\widehat{\Delta\lambda_c} = \Delta\lambda_M$	$\widehat{\Delta\lambda_c} = 0$	$\widehat{\Delta\lambda_c} = \Delta\lambda_M$
6	-1.2 1	F	14(26)	14(26)	14(26)
	-2 4	F	17(27)	17(28)	17(28)
	-6 5	F	8(9)	8(11)	8(11)
	-12 10	F	16(22)	8(10)	8(10)
	-10 10	F	15(23)	7(9)	7(9)
	-10 0	F	12(18)	14(21)	14(21)
	20 20	7(9)	7(9)	12(19)	12(19)
7	-35 -40	22(28)	23(31)	26(31)	26(31)
26	5 -5 5	28(32)	26(31)	27(31)	29(35)
	50 -50 50	34(41)	28(29)	28(29)	28(29)
	450 -370 645	39(52)	38(52)	34(35)	34(35)
	-0.3 2.1 -2.1	21(25)	21(24)	21(25)	21(24)
27	2 2 2	F	17(26)	16(26)	16(26)
	1 4 2	F	10(15)	9(13)	9(11)
	1 3 4	F	11(14)	23(31)	25(38)
	1 4 4	F	11(18)	20(29)	29(44)
	-4 -2 -1	11(17)	13(22)	13(17)	13(17)
	20 20 20	131(154)	105(149)	105(143)	105(143)
	5 -10 8	41(59)	31(51)	28(42)	33(53)
	15 -9 3	F	F	53(82)	185(246)
	-2 6 -11	13(19)	10(15)	12(17)	12(17)
	10 5 7	F	F	54(82)	108(143)
	23 -19 38	122(155)	104(142)	98(125)	98(125)
39	2 2 2 2	F	13(16)	15(17)	15(17)
	-10 -10 10 10	16(24)	16(26)	16(21)	16(21)
	20 20 20 20	F	32(41)	38(54)	38(54)
	-3 2 3 7	13(19)	13(19)	10(14)	10(14)
	5 0 -5 0	F	14(17)	13(19)	13(19)
	-3 -5 7 9	16(25)	F	15(20)	15(20)
	-2 -4 6 2	14(20)	18(29)	14(21)	14(21)
	20 49 63 -9	46(76)	40(59)	36(49)	36(49)
	4 -5 9 36	31(43)	31(43)	41(51)	41(51)
	40 2 4 -5	38(52)	66(82)	78(113)	78(113)
	0 0 0 0	F	8(14)	11(18)	11(18)

F Failed to converge.

Table 7.2 Multiplier Test Results

Problem	Starting Point	# Iterations (# Function Evaluations)			
		$\lambda_+ = \lambda_c + \widehat{\Delta\lambda_c}$		$\lambda_+ = \lambda_c + \widehat{\Delta\lambda_c} + \Delta\lambda_c$	
		$\widehat{\Delta\lambda_c} = 0$	$\widehat{\Delta\lambda_c} = \Delta\lambda_M$	$\widehat{\Delta\lambda_c} = 0$	$\widehat{\Delta\lambda_c} = \Delta\lambda_M$
40	.8 .8 .8 .8	F	F	3(4)	3(4)
	-1 -1 -1 -1	9(12)	16(22)	9(13)	10(15)
	-2 -4 -4 -2	F	156(222)	14(20)	14(20)
	1 0 -1 0	7(10)	7(11)	7(10)	6(7)
	-2 -4 6 2	F	F	F	17(25)
	0 -0.5 1 0	11(14)	10(13)	8(10)	7(9)
	8 8 8 8	F	13(20)	12(19)	12(19)
	3 2 4 7	9(13)	9(13)	8(9)	8(9)
	30 29 -39 3	F	96(113)	116(172)	66(96)
	5 3 -100 -10	101(129)	F	77(120)	69(90)
42	10 10 10 10	8(9)	8(9)	7(8)	8(9)
	100 -100 30 -70	F	11(14)	12(16)	12(16)
60	-10 40 9	18(24)	18(24)	15(17)	15(17)
	100 100 -100	F	18(19)	18(19)	18(19)
77	2 2 2 2 2	8(10)	8(10)	9(11)	9(11)
	10 10 10 10 10	21(23)	21(23)	23(25)	23(25)
	20 20 20 20 20	26(27)	26(27)	24(26)	24(26)
	4 3 7 -5 -3	F	33(43)	F	F
	-1 3 -0.5 -2 -3	F	35(51)	F	F
	12 13 14 15 7	19(22)	19(22)	20(23)	20(23)
	-2 -2 -2 -2 -2	F	F	108(143)	310(370)
	-5 -5 -5 -5 -5	F	F	50(80)	F
	3 2 7 1 9	18(26)	17(25)	F	F
	-1 2 5 0 6	F	269(295)	19(27)	97(117)
	0 0 0 0 0	F	10(18)	9(16)	9(16)
78	-20 15 20 -10 -10	F	F	13(19)	13(20)
	50 50 50 50 50	F	43(64)	24(41)	23(38)
	-10 10 10 -10 -10	F	16(29)	8(9)	8(9)
	0 0 1 1 1	7(11)	6(10)	6(10)	6(10)
79	10 10 10 10 10	11(12)	11(12)	10(11)	10(11)
	-2 -2 -2 -2 -2	28(35)	16(23)	10(13)	10(13)
	-3 2 6 -7 9	15(16)	15(16)	19(24)	18(21)
	40 -30 50 -80 20	F	51(80)	20(29)	16(21)

F **Failed to converge.**

Table 7.3 Multiplier Test Results

Problem	Starting Point	Solution Found			
		$\lambda_+ = \lambda_c + \widehat{\Delta\lambda_c}$		$\lambda_+ = \lambda_c + \widehat{\Delta\lambda_c} + \Delta\lambda_c$	
		$\widehat{\Delta\lambda_c} = 0$	$\widehat{\Delta\lambda_c} = \Delta\lambda_M$	$\widehat{\Delta\lambda_c} = 0$	$\widehat{\Delta\lambda_c} = \Delta\lambda_M$
26	5 -5 5	ii	ii	ii	ii
	50 -50 50	i	i	i	i
	450 -370 645	i	i	i	i
	-0.3 2.1 -2.1	i	ii	i	i
40	.8 .8 .8 .8	F	F	i	i
	-1 -1 -1 -1	i	i	i	i
	-2 -4 -4 -2	F	ii	ii	ii
	1 0 -1 0	ii	ii	ii	ii
	-2 -4 6 2	F	F	F	i
	0 -0.5 1 0	i	i	ii	ii
	8 8 8 8	F	i	i	i
	3 2 4 7	i	i	i	i
	30 29 -39 3	F	iii	ii	ii
	5 3 -100 -10	iii	F	ii	ii
60	-10 40 9	i	i	i	i
	100 100 -100	F	ii	ii	ii
77	2 2 2 2 2	i	i	i	i
	10 10 10 10 10	i	i	i	i
	20 20 20 20 20	ii	i	ii	ii
	4 3 7 -5 -3	F	ii	F	F
	-1 3 -0.5 -2 -3	F	ii	F	F
	12 13 14 15 7	i	i	i	i
	-2 -2 -2 -2 -2	F	F	iii	iii
	-5 -5 -5 -5 -5	F	F	i	F
	3 2 7 1 9	ii	ii	F	F
	-1 2 5 0 6	F	ii	ii	ii
	0 0 0 0 0	F	i	i	i
78	-20 15 20 -10 -10	F	F	iii	iii
	50 50 50 50 50	F	iii	ii	ii
	-10 10 10 -10 -10	F	iii	i	i
	0 0 1 1 1	ii	ii	ii	ii
79	10 10 10 10 10	i	i	i	i
	-2 -2 -2 -2 -2	iii	iii	iii	iii
	-3 2 6 -7 9	i	i	ii	ii
	40 -30 50 -80 20	F	iv	iv	iv

F Failed to converge.

i,ii,iii,iv Solution found: See Appendix A.

7.3 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 $\|\nabla {h_c}^T s + h_c\|$. 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 ρ_c to $\hat{\rho}_c$ before we evaluate each step, and, if we accept the step, then we 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 (7.8). We use the following strategy for updating the penalty constant.

Updating the Penalty Constant:

Given a small, fixed constant $\beta > 0$ and $\rho_0 > 0$, at each iteration:

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

$$\hat{\rho}_c = 2\frac{\nabla_x {l_c}^T s_c + \frac{1}{2}{s_c}^T B_c s_c + \Delta{\lambda_c}^T(h_c + \nabla {h_c}^T s_c)}{\|h_c\|^2 - \|h_c + \nabla {h_c}^T s_c\|^2} + \beta$$

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,

$$\|h_c\|^2 - \|h_c + \nabla h_c{}^T s_c\|^2 > 0 \ . \tag{7.6}$$

to insure that the sequence of penalty constants is nondecreasing. If condition (7.6) does not hold, then it is possible that the updated penalty constant $\hat{\rho}_c$ could be negative. If the trust region subproblem was solved exactly, then (7.6) will hold. However, in practice, the subproblem is only solved approximately, and care must be taken to ensure (7.6) holds. See Section 6.5.1 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 ρ_0 if $\rho_+ > \rho_{MAX}$, and for this implementation, we set $\rho_{MAX} = 1 \times 10^6$.

The values of the constants that we use in our implementation of this penalty constant strategy are $\rho_0 = 1.1$, $\beta = 0.1$, and $\rho_{MAX} = 1 \times 10^6$.

7.4 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, λ_c) to the new iterate $(x_+, \widehat{\lambda}_+)$,

$$\begin{aligned} ared_c &= L(x_c, \lambda_c) - L(x_+, \widehat{\lambda}_+) \\ &= l(x_c, \lambda_c) - l(x_+, \lambda_c) - \left(\widehat{\lambda}_+ - \lambda_c\right)^T h_+ \\ &\quad + \hat{\rho}_c \left(\|h_c\|^2 - \|h_+\|^2\right) \end{aligned} \tag{7.7}$$

to the *predicted reduction*,

$$\begin{aligned} pred_c &= -\nabla_x l(x_c, \lambda_c)^T s_c - \frac{1}{2} s_c{}^T B_c s_c - \left(\widehat{\lambda}_+ - \lambda_c\right)^T \left(h_c + \nabla h_c{}^T s_c\right) \\ &\quad + \hat{\rho}_c \left[\|h_c\|^2 - \|h_c + \nabla h_c{}^T s_c\|^2\right] \ . \end{aligned} \tag{7.8}$$

If the agreement between the actual and predicted reduction is reasonable and the step gives at least a small amount of decrease, i. e.,

$$0 < \eta_1 \leq \frac{ared_c}{pred_c}$$

where $\eta_1 \in (0,1)$ is a small. fixed constant, then the point $x_+ = x_c + s_c$ is accepted and λ_+ is computed from (7.4). 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 $0 < \alpha_2 < 1$. Consider, for example, $\alpha_2 = 0.5$. This has the effect of halving the trust region, or if the step lies strictly in the interior of the trust region, then Δ_+ would be set to half the length of this step.

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

$$\eta_1 \leq \frac{ared_c}{pred_c} < \eta_2 \ ,$$

where $\eta_2 \in (\eta_1, 1)$ is a fixed constant, then the radius of the trust region is decreased by the rule $\Delta_+ = \alpha_2 \|s_c\|$ where $0 < \alpha_2 < 1$. However, if the agreement is very good,

$$\eta_3 \leq \frac{ared_c}{pred_c} \ ,$$

where $\eta_3 \in (\eta_2, 1)$ is a fixed constant, then we possibly increase the radius of the trust region by

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

where Δ_{MAX} is the maximum allowable trust region radius and $\alpha_3 > 1$.

The values of the trust region constants that we use in our implementation are $\eta_1 = 0.001$, $\eta_2 = 0.25$, $\eta_3 = .75$, $\alpha_2 = 0.5$, $\alpha_3 = 2.0$, and $\Delta_{MAX} = 20\Delta_0$. We experimented with several choices for Δ_0; the length of the Cauchy step for the constraints, the distance to the linearized constraint manifold, $\|s_{LF}\|$, and $1.5\|s_{LF}\|$. These results can be found in Tables 7.4 and 7.5, and they are summarized at the end of Table 7.5. Surprisely, the most conservative choice, $\|s_{CP}\|$, was the most efficient. Unfortunately, as in unconstrained optimization, the behavior of the algorithm is sensitive to the choice of the initial trust region radius. 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].

Table 7.4 Effect of the Initial Trust Region Radius

Problem	Starting Point	# Iterations (# Func Evaluations) $\lVert s_{CP} \rVert$	$\lVert s_{LF} \rVert$	$1.5\lVert s_{LF} \rVert$
6	-6 5	8(11)	8(11)	11(15)
	-12 10	8(10)	8(10)	17(26)
	-10 10	7(9)	7(9)	9(12)
	-10 0	14(21)	14(21)	124(170)
	20 20	12(19)	12(19)	478(843)
7	20 20	18(22)	18(23)	18(23)
	-15 6	17(21)	17(21)	15(17)
	23 -10	17(20)	17(20)	22(27)
	-35 -40	26(31)	26(31)	18(21)
9	-10 10	6(7)	6(7)	3(4)
26	5 -5 5	27(31)	27(31)	26(33)
	-26 20 20	25(26)	25(26)	17(18)
	450 -370 645	34(35)	33(34)	33(34)
27	2 2 2	16(26)	16(26)	18(23)
	1 4 2	9(13)	9(13)	14(27)
	1 3 4	23(31)	23(31)	26(41)
	1 4 4	20(29)	20(28)	24(32)
	1 1 1	11(15)	11(15)	12(16)
	-10 -10 -10	28(37)	28(37)	24(33)
	20 20 20	105(143)	107(146)	80(111)
	5 -10 8	28(42)	26(42)	29(44)
	15 -9 3	53(82)	53(82)	29(44)
	-2 6 -11	12(17)	12(17)	16(21)
	10 5 7	54(82)	55(78)	48(78)
	23 -19 38	98(125)	98(125)	73(99)
39	2 2 2 2	15(17)	15(17)	10(14)
	0 2 0 0	5(6)	5(6)	5(7)
	-10 -10 10 10	16(21)	20(27)	23(37)
	20 20 20 20	38(54)	62(85)	39(58)
	-3 2 3 7	10(14)	16(20)	10(13)
	5 0 -5 0	13(19)	16(21)	17(19)
	-11 8 -2 9	13(15)	347(666)	28(38)
	-3 -5 7 9	15(20)	14(22)	13(21)
	-2 -4 6 2	14(21)	31(35)	13(20)
	20 49 63 -9	36(49)	F	171(284)
	4 -5 9 36	41(51)	38(51)	29(42)
	40 2 4 -5	78(113)	210(335)	180(283)

F Failed to converge.

Table 7.5 Effect of the Initial Trust Region Radius

Problem	Starting Point	# Iterations (# Func Evaluations) $\|s_{CP}\|$	$\|s_{LF}\|$	$1.5\|s_{LF}\|$
40	-1 -1 -1 -1	9(13)	F	8(13)
	-2 -4 -4 -2	14(20)	24(38)	12(19)
	-2 -4 6 2	F	16(24)	24(34)
	0 -0.5 1 0	8(10)	8(10)	11(16)
	8 8 8 8	12(19)	10(11)	10(14)
	3 2 4 7	8(9)	16(23)	9(14)
	30 29 -39 3	116(172)	13(18)	17(33)
	5 3 -100 -10	77(120)	58(88)	69(109)
42	1 2 3 4	6(7)	5(6)	5(6)
	25 -30 -10 9	9(10)	8(10)	10(14)
	100 -100 30 -70	12(16)	12(14)	12(15)
60	0 0 0	11(17)	11(17)	9(14)
	10 10 10	11(12)	11(12)	12(13)
	-20 -20 -20	13(14)	13(14)	12(13)
	-10 40 9	15(17)	15(17)	17(21)
77	1 1 1 1 1	6(9)	6(9)	8(11)
	10 10 10 10 10	23(25)	22(27)	17(18)
	20 20 20 20 20	24(26)	24(27)	23(29)
	4 3 7 -5 -3	F	93(113)	F
	-1 3 -0.5 -2 -3	F	134(165)	F
	12 13 14 15 7	20(23)	18(20)	21(22)
	-2 -2 -2 -2 -2	108(143)	108(143)	161(192)
	-5 -5 -5 -5 -5	50(80)	F	75(93)
	-1 2 5 0 6	19(27)	35(47)	31(43)
	0 0 0 0 0	9(16)	9(16)	10(18)
78	-20 15 20 -10 -10	13(19)	10(18)	10(18)
	50 50 50 50 50	24(41)	18(33)	25(39)
	-10 10 10 -10 -10	8(9)	7(8)	7(8)
	0 0 1 1 1	6(10)	7(12)	6(11)
79	10 10 10 10 10	10(11)	9(10)	17(23)
	-2 -2 -2 -2 -2	10(13)	11(15)	9(12)
	3 5 5 5 5	9(10)	10(11)	14(17)
	-3 2 6 -7 9	19(24)	12(13)	12(13)
	40 -30 50 -80 20	20(29)	61(102)	45(78)
Totals		1769(2276)	2254(3261)	2458(3620)
Averages		25.3(32.5)	32.7(47.3)	35.1(51.7)
Failures		**3F**	**3F**	**2F**

F Failed to converge.

7.5 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:

1. Initialization:

   (a) Given x_0, obtain h_0, ∇h_0, f_0, ∇f_0, and the constants ρ_0, ρ_{MAX}, β, ε, $0 < \alpha_2 < 1, \alpha_3 > 1$, and $0 < \eta_1 < \eta_2 < \eta_3 < 1$.

   (b) Calculate λ_0 from $\nabla h_0 \lambda = -\nabla f_0$.

   (c) Calculate $\Delta_0 = \max\{\|s_{CP}\|, 1.5\}$.

   (d) $\Delta_{MAX} = \max\{20\Delta_0, 10.0\}$.

2. Calculate a trial step s_c.

3. $x_+ = x_c + s_c$.

4. Calculate the Lagrange multiplier update.

   If $(s_c = s_{QP})$, **then**

   $\widehat{\Delta\lambda}_c = \Delta\lambda_{QP}$

   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 $\widehat{\Delta\lambda}_c$

   Else

   $\widehat{\Delta\lambda}_c = 0$

   End if

   End if

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

6. Calculate the Predicted Reduction:

$$\begin{aligned} pred_c &= -\nabla_x l_c{}^T s_c - \frac{1}{2} s_c{}^T B_c s_c - \widehat{\Delta\lambda}_c^T (h_c + \nabla h_c{}^T s_c) \\ &\quad + \rho_c \left(\|h_c\|^2 - \|h_c + \nabla h_c{}^T s_c\|^2 \right). \end{aligned}$$

7. 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**

$\hat{\rho}_c = \rho_c$

Else

$$\hat{\rho}_c = 2\frac{\nabla_x l_c{}^T s_c + \frac{1}{2}s_c{}^T B_c s_c + \Delta\lambda_c{}^T(h_c + \nabla h_c{}^T s_c)}{\|h_c\|^2 - \|h_c + \nabla h_c{}^T s_c\|^2} + \beta$$

$$\begin{aligned} pred_c &= -\nabla_x l_c{}^T s_c - \frac{1}{2}s_c{}^T B_c s_c - \widehat{\Delta\lambda}_c^T(h_c + \nabla h_c{}^T s_c) \\ &\quad + \hat{\rho}_c\left(\|h_c\|^2 - \|h_c + \nabla h_c{}^T s_c\|^2\right). \end{aligned}$$

End if

8. Calculate the Actual Reduction:

$$ared_c = l_c - l(x_+, \lambda_c) - \widehat{\Delta\lambda}_c^T h(x_+) + \hat{\rho}_c\,(\|h_c\|^2 - \|h(x_+)\|^2).$$

9. Evaluate the Step and Update the Trust Region:

(a) **Given** constants $0 < \alpha_2 < 1, \alpha_3 > 1$, and $0 < \eta_1 < \eta_2 < 1$,

(b) **If** $(\frac{ared_c}{pred_c} < \eta_1)$, **then**

Do not accept the step:

$x_+ = x_c$

$\lambda_+ = \lambda_c$

Reduce the trust region radius:

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

End if

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

Accept the step.

Reduce the trust region radius:

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

End if

(d) **If** $(\eta_3 \leq \frac{ared_c}{pred_c})$, **then**

Accept the step.

Possibly increase the trust region radius:

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

End if

(e) **If** (Step Not Accepted), **then**

$\rho_+ = \rho_c$

$f_+ = f_c$, $h_+ = h_c$, $\nabla f_+ = \nabla f_c$, $\nabla h_+ = \nabla h_c$, and $B_+ = B_c$

Go To 2.

Else

$\rho_+ = \hat{\rho}_c$

If $(\rho_+ > \rho_{MAX})$, $\rho_+ = \rho_0$

End if

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

11. Update the Lagrange Multipliers:

    (a) Solve $\nabla h_+ \Delta\lambda = -\nabla f_+ - \nabla h_+ \left(\lambda_c + \widehat{\Delta\lambda_c}\right)$ for $\Delta\lambda_c$

    (b) $\lambda_+ = \lambda_c + \widehat{\Delta\lambda}_c + \Delta\lambda_c$

12. Test for Convergence:

If $(\|\nabla_x l(x_+, \lambda_+)\| + \|h(x_+)\| \leq \varepsilon)$, **then**

Solution found, **Stop.**

Else

Get $B_+ \equiv B(x_+, \lambda_+)$.

Go To 2.

End if

7.6 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_o, we use the least squares solution to

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

for the initial multipliers λ_0.

As discussed in Section 7.4, we choose the initial trust region radius to be the length of the Cauchy step for the Gauss-Newton model of the constraints. Since the Cauchy step from x_o could be zero, we use

$$\Delta_0 = \max\{\|s_{CP}\|, 1.5\}.$$

We would prefer a choice of Δ_0 that comes from the problem instead of some absolute constant like 1.5, and in the future, we will perhaps consider some fraction of the length of the initial SQP step.

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

$$\|\nabla_x l(x_+, \lambda_+)\| + \|h(x_+)\| \leq \varepsilon$$

where $\varepsilon = 1 \times 10^{-6}$. 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 1×10^{-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.

7.7 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 two SQP approaches: `NPSOL` by Gill, Murray, Saunders, and Wright [1986], and `DNCONG` by Schittkowski [1986], which is available in the IMSL MATH/LIBRARY. 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, and all tests were performed in double precision on a Sun 4.

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. The first starting point listed is the standard starting point from Hock and Schittkowski [1981].

In order to study the robustness of the algorithm, we tested each problem from several starting points. The results are reported in Tables 7.6, 7.7 and 7.8 with a summary at the end of Table 7.8. The numbers in the columns labeled NPSOL, DNCONG and NLPTR indicate the number of iterations that each algorithm required for convergence when convergence was achieved, and the letter F indicates that the algorithm failed to converge. The numbers in parenthesis indicates the number of function evaluations required, i. e.,the number of objective function evaluations or the number of constraint evaluations. The number of iterations provides some insight into the difficulty of the problem. Also, the difference between the number of iterations and the number of function evaluations indicates how difficult it was for the algorithm to find acceptable steps for that problem. The gradient evaluations that both NLPTR and DNCONG required is equivalent to the number of iterations, while NPSOL required one gradient evaluation for each function evaluation.

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 uses exact Hessian information while 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 show that our algorithm is significantly more robust than DNCONG and NPSOL. We point out that DNCONG failed on a number of problems because the line search could not find an acceptable step with the allowed number of function calls. The default for the function evaluations allowed during the line search is 5, and the IMSL routine will not let the user override this default. 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 7.9.

Table 7.6 Convergence Results

Problem	Starting Point	# Iterations (# Func Evaluations)		
		NPSOL	DNCONG	NLPTR
6	-1.2 1	13(20)	9(10)	14(26)
	-12 10	8(11)	12(12)	8(10)
	-10 10	9(12)	16(24)	7(9)
	-10 0	11(30)	12(14)	14(21)
	20 20	11(14)	11(12)	12(19)
7	2 2	11(17)	10(11)	7(8)
	20 20	22(34)	F†	18(22)
	-15 6	31(59)	16(19)	17(21)
	23 -10	24(39)	22(28)	17(20)
	-35 -40	26(41)	28(37)	26(31)
8	2 1	6(10)	5(5)	5(7)
	20 10	6(8)	6(6)	6(7)
	-50 -50	F	F	F
9	0 0	5(9)	6(6)	4(5)
	-10 10	5(9)	8(9)	6(7)
26	-2.6 2 2	40(44)	20(22)	17(18)
	0 0 0	42(49)	10(14)	18(20)
	-1 -1 -1	41(44)	19(20)	19(20)
	1 -1 1	57(78)	17(19)	19(21)
	5 -5 5	77(105)	31(32)	27(31)
	-26 20 20	46(56)	30(30)	25(26)
	50 -50 50	95(150)	62(71)	28(29)
	30 35 40	51(70)	30(32)	20(21)
	450 -370 645	104(150)	F‡	34(35)
	-0.3 2.1 -2.1	43(51)	22(24)	21(25)
27	2 2 2	18(29)	21(27)	16(26)
	1 4 2	22(33)	23(30)	9(13)
	1 3 4	21(29)	20(22)	23(31)
	1 4 4	27(44)	25(30)	20(29)
	-4 -2 -1	22(28)	37(61)	13(17)
	1 1 1	25(35)	21(28)	11(15)
	-10 -10 -10	64(85)	65(108)	28(37)
	20 20 20	20(33)	441(1254)	105(143)
	5 -10 8	28(48)	24(25)	28(42)
	15 -9 3	47(74)	273(740)	53(82)
	-2 6 -11	29(51)	28(36)	12(17)
	10 5 7	18(31)	110(266)	54(82)
	23 -19 38	27(45)	70(101)	98(125)

F Failed to converge.

† Too many function calls in Line Search.

‡ Reached maximum number of Iterations.

Table 7.7 Convergence Results

Problem	Starting Point	# Iterations (# Func Evaluations) NPSOL	DNCONG	NLPTR
39	2 2 2 2	12(16)	12(14)	15(17)
	0 2 0 0	2(4)	3(3)	5(6)
	-10 -10 10 10	37(65)	37(44)	16(21)
	20 20 20 20	20(25)	20(22)	38(54)
	-3 2 3 7	29(52)	16(17)	10(14)
	-11 8 -2 9	25(36)	49(81)	13(15)
	-3 -5 7 9	32(54)	44(67)	15(20)
	-2 -4 6 2	94(233)	31(39)	14(21)
	20 49 63 -9	21(26)	26(33)	36(49)
	4 -5 9 36	25(44)	24(33)	41(51)
	40 2 4 -5	119(320)	23(28)	78(113)
	0 0 0 0	2(4)	3(3)	11(18)
40	.8 .8 .8 .8	6(10)	6(6)	3(4)
	-1 -1 -1 -1	46(135)	36(59)	9(13)
	-2 -4 -4 -2	30(56)	16(23)	14(20)
	1 0 -1 0	12(15)	26(33)	7(10)
	-2 -4 6 2	27(38)	24(29)	F‡
	0 -0.5 1 0	F	F†	8(10)
	8 8 8 8	23(33)	40(64)	12(19)
	3 2 4 7	28(48)	30(49)	8(9)
	30 29 -39 3	26(40)	F^{max}	116(172)
	5 3 -100 -10	51(120)	38(60)	77(120)
42	1 1 1 1	7(12)	9(11)	3(4)
	10 10 10 10	12(18)	10(11)	7(8)
	25 -30 -10 9	15(26)	17(18)	9(10)
	100 -100 30 -70	18(28)	23(25)	12(16)
	-50 -75 40 100	16(23)	29(36)	11(12)
60	2 2 2	8(12)	10(11)	7(8)
	11 12 13	22(29)	18(23)	11(12)
	0 0 0	20(42)	20(29)	11(17)
	10 10 10	18(32)	19(23)	11(12)
	20 20 20	21(31)	17(20)	13(14)
	27 29 38	22(30)	23(24)	15(16)
	-20 -20 -20	24(36)	20(22)	13(14)
	1 1 1	8(14)	8(9)	6(8)
	-10 40 9	66(124)	42(59)	15(17)
	-45 11 87	25(34)	49(60)	18(19)
	100 100 -100	29(39)	F	18(19)

F Failed to converge.

† Too many function calls in Line Search.

‡ Reached maximum number of Iterations.

Table 7.8 Convergence Results

Problem	Starting Point	# Iterations (# Func Evaluations) NPSOL	DNCONG	NLPTR
77	2 2 2 2 2	14(21)	15(16)	9(11)
	1 1 1 1 1	13(20)	13(17)	6(9)
	10 10 10 10 10	F	61(73)	23(25)
	20 20 20 20 20	F	120(178)	24(26)
	-3 -3 3 9 0	F	26(27)	10(11)
	-1 8 3 3 0	15(23)	20(25)	8(9)
	4 3 7 -5 -3	F	39(40)	F
	12 13 14 15 7	F	70(94)	20(23)
	-2 -2 -2 -2 -2	F	41(55)	108(143)
	-5 -5 -5 -5 -5	F	F†	50(80)
	3 2 7 1 9	F	56(72)	F
	-1 2 5 0 6	192(366)	F†	19(27)
	0 0 0 0 0	24(44)	31(47)	9(16)
78	-2 1.5 2 -1 -1	10(15)	8(8)	4(5)
	-20 15 20 -10 -10	39(55)	38(60)	13(19)
	50 50 50 50 50	108(273)	F^{max}	24(41)
	-1 1.5 2 -1 -2	14(23)	11(13)	5(6)
	-10 10 10 -10 -10	13(18)	13(13)	8(9)
	0 0 1 1 1	F	F^{max}	6(10)
79	2 2 2 2 2	10(13)	9(10)	4(5)
	1 1 1 1 1	9(11)	8(8)	4(5)
	10 10 10 10 10	21(32)	19(21)	10(11)
	-2 -2 -2 -2 -2	20(39)	25(37)	10(13)
	3 5 5 5 5	17(24)	15(18)	9(10)
	-3 2 6 -7 9	22(39)	32(33)	19(24)
	40 -30 50 -80 20	77(115)	44(48)	20(29)
Totals		2768(4710)	3092(5186)	2007(2667)
Averages		30.4(51.8)	33.6(56.4)	20.5(27.2)
Failures		11F	10F	4F

F Failed to converge.

† Too many function calls in Line Search.

‡ Reached maximum number of Iterations.

max Converged to a maximum.

Table 7.9 Solutions Found

Problem	Starting Point	Solution Found		
		NPSOL	DNCONG	NLPTR
8	45 60	ii	ii	ii
9	-10 10	i	i	ii
26	-1 -1 -1	ii	ii	ii
	5 -5 5	i	i	ii
	50 -50 50	i	ii	i
	450 -370 645	i	F	i
	-0.3 2.1 -2.1	ii	ii	i
40	-1 -1 -1 -1	ii	ii	i
	-2 -4 -4 -2	ii	ii	ii
	1 0 -1 0	ii	ii	ii
	-2 -4 6 2	iv	iii	F
	0 -0.5 1 0	F	F	ii
	8 8 8 8	ii	iii	i
	3 2 4 7	ii	i	i
	30 29 -39 3	i	F	ii
	5 3 -100 -10	i	ii	ii
60	-20 -20 -20	ii	ii	ii
	100 100 -100	ii	F	ii
77	10 10 10 10 10	F	i	i
	20 20 20 20 20	F	iii	ii
	-3 -3 3 9 0	F	ii	ii
	-1 8 3 3 0	ii	ii	ii
	4 3 7 -5 -3	F	i	F
	12 13 14 15 7	F	i	i
	-2 -2 -2 -2 -2	F	iii	iii
	-5 -5 -5 -5 -5	F	F	i
	3 2 7 1 9	F	i	F
	-1 2 5 0 6	i	F	ii
78	-20 15 20 -10 -10	iii	i	iii
	50 50 50 50 50	i	F	ii
	0 0 1 1 1	F	F	ii
79	-2 -2 -2 -2 -2	iii	ii	iii
	-3 2 6 -7 9	v	i	ii
	40 -30 50 -80 20	ii	iii	iv

F Failed to converge.

Chapter 8

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 $\nabla h(x)^T$ 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]. Second-order sufficiency holds at the points x_* given below unless otherwise noted.

- Hock and Schittkowski 6

$$\begin{array}{rl} \text{minimize} & (1-x_1)^2 \\ \text{subject to} & 10(x_2 - x_1^2) = 0 \end{array}$$

$x_* = (1.0, 1.0)^T$; $f(x_*) = 0.0$

- Hock and Schittkowski 7

$$\begin{array}{rl} \text{minimize} & \ln(1+x_1^2) - x_2 \\ \text{subject to} & (1+x_1^2)^2 + x_2^2 - 4 = 0 \end{array}$$

$x_* = (0.0, \sqrt{3.0})^T$; $f(x_*) = -\sqrt{3.0}$

- Hock and Schittkowski 8

$$\begin{array}{rl} \text{minimize} & -1.0 \\ \text{subject to} & x_1^2 + x_2^2 - 25.0 = 0 \\ & x_1 x_2 - 9.0 = 0 \end{array}$$

i. $x_* = (4.60159, 1.95584)^T$; $f(x_*) = -1.0$
ii. $x_* = (1.95584, 4.60159)^T$; $f(x_*) = -1.0$
iii. $x_* = (-4.60159, -1.95584)^T$; $f(x_*) = -1.0$
iv. $x_* = (-1.95584, -4.60159)^T$; $f(x_*) = -1.0$

- Hock and Schittkowski 9

$$\begin{array}{rl} \text{minimize} & \sin(\Pi x_1/12)\ \cos(\Pi x_2/16) \\ \text{subject to} & 4x_1 - 3x_2 = 0 \end{array}$$

i. $x_* = (-3.0, -4.0)^T$; $f(x_*) = -0.5$

ii. $x_* = (9.0, 12.0)^T$; $f(x_*) = -0.5$

iii. $x_* = (-15.0, -20.0)^T$; $f(x_*) = -0.5$

- Hock and Schittkowski 26

$$\begin{array}{ll} \text{minimize} & (x_1 - x_2)^2 + (x_2 - x_3)^4 \\ \text{subject to} & (x_2^2 + 1)x_1 + x_3^4 - 3 = 0 \end{array}$$

i. $x_* = (1.0, 1.0, 1.0)^T$; $f(x_*) = 0.0$

ii. $x_* = (-1.809, -1.809, -1.810)^T$; $f(x_*) = 0.0$

- Hock and Schittkowski 27

$$\begin{array}{ll} \text{minimize} & 0.01(x_1 - 1)^2 + (x_2 - x_1^2)^2 \\ \text{subject to} & x_1 + x_3^2 + 1 = 0 \end{array}$$

$x_* = (-1.0, 1.0, 0.0)^T$; $f(x_*) = 0.04$

- Hock and Schittkowski 39

$$\begin{array}{ll} \text{minimize} & -x_1 \\ \text{subject to} & x_2 - x_1^3 - x_3^2 = 0 \\ & x_1^2 - x_2 - x_4^2 = 0 \end{array}$$

$x_* = (1.0, 1.0, 0.0, 0.0)^T$; $f(x_*) = -1.0$

- Hock and Schittkowski 40

$$\begin{array}{ll} \text{minimize} & -x_1x_2x_3x_4 \\ \text{subject to} & x_1^3 + x_2^2 - 1 = 0 \\ & x_1^2x_4 - x_3 = 0 \\ & x_4^2 - x_2 = 0 \end{array}$$

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. $x_* = (0, 1.0, 0, 1.0)^T$; $f(x_*) = 0.0$; (Reduced Hessian is zero).

iv. $x_* = (0, 1.0, 0, -1.0)^T; f(x_*) = 0.0$; (Reduced Hessian is zero).

- Hock and Schittkowski 42

$$\begin{array}{rl} \text{minimize} & (x_1-1)^2+(x_2-2)^2+(x_3-3)^2+(x_4-4)^2 \\ \text{subject to} & x_1-2=0 \\ & x_3^2+x_4^2-2=0 \end{array}$$

$x_* = (2, 2, 0.848528, 1.13137)^T$; $f(x_*) = 13.8579$

- Hock and Schittkowski 60

$$\begin{array}{rl} \text{minimize} & (x_1-1)^2+(x_1-x_2)^2+(x_2-x_3)^4 \\ \text{subject to} & x_1(1+x_2^2)+x_3^4-4-3\sqrt{2}=0 \end{array}$$

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

ii. $x_* = (0.0986, -0.895, -1.6519)^T$; $f(x_*) = 2.189$

- Hock and Schittkowski 77

$$\begin{array}{rl} \text{minimize} & (x_1-1)^2+(x_1-x_2)^2+(x_3-1)^2+(x_4-1)^4+(x_5-1)^6 \\ \text{subject to} & x_1^2x_4+\sin(x_4-x_5)-2\sqrt{2}=0 \\ & x_2+x_3^4x_4^2-8-\sqrt{2}=0 \end{array}$$

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

$$\begin{array}{rl} \text{minimize} & x_1x_2x_3x_4x_5 \\ \text{subject to} & x_1^2+x_2^2+x_3^2+x_4^2+x_5^2-10=0 \\ & x_2x_3-5x_4x_5=0 \\ & x_1^3+x_2^3+1=0 \end{array}$$

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. $x_* = (-0.6991, -0.8700, -2.790, -0.6967, -0.6967)^T$; $f(x_*) = -0.8236$

- Hock and Schittkowski 79

$$
\begin{array}{ll}
\text{minimize} & (x_1-1)^2 + (x_1-x_2)^2 + (x_2-x_3)^2 + (x_3-x_4)^4 + (x_4-x_5)^4 \\
\text{subject to} & x_1 + x_2^2 + x_3^3 - 2 - 3\sqrt{2} = 0 \\
 & x_2 - x_3^2 + x_4 + 2 - 2\sqrt{2} = 0 \\
 & x_1 x_5 - 2 = 0
\end{array}
$$

i. $x_* = (1.191, 1.362, 1.473, 1.635, 1.679)^T$; $f(x_*) = 0.0788$

ii. $x_* = (-0.7662, 2.667, -0.4682, -1.619, -2.610)^T$; $f(x_*) = 27.45$

iii. $x_* = (-2.702, -2.990, 0.1719, 3.848, -0.7401)^T$; $f(x_*) = 649.1$

iv. $x_* = (2.718, 2.033, -0.848, -0.486, 0.736)^T$; $f(x_*) = 13.96$

v. $x_* = (-1.247, 2.422, 1.175, -0.2132, -1.604)^T$; $f(x_*) = 27.52$

Bibliography

[1] Mordecai Avriel. *Nonlinear Programming: Analysis and Methods.* Prentice-Hall. Inc.. 1976.

[2] R. H. Byrd, R. B. Schnabel. and G. A. Schultz. A trust region algorithm for nonlinearly constrained optimization. *SIAM J. Numer. Anal.*, 24, 1987.

[3] R. H. Byrd, R. B. Schnabel, and G. A. Schultz. Approximate solution of the trust region problem by minimization over two-dimensional subspaces. *Math. Prog.*, 40(3), 1988.

[4] M. R. Celis, J. E. Dennis, and R. A. Tapia. A trust region strategy for equality constrained optimization. In *Numerical Optimization.* SIAM, 1985.

[5] Maria Rosa Celis. *A Trust Region Strategy for Nonlinear Equality Constrained Optimization.* PhD thesis, Rice University, 1985.

[6] J. E. Dennis, Mahmoud El-Alem, and R. A. Tapia. Numerical experience with a polyhedral norm version of the Celis-Dennis-Tapia trust-region algorithm. Technical report, Rice University. In preparation.

[7] J. E. Dennis, J. M. Martinez, and K. A. Williamson. An algorithm based on a convenient trust-region subproblem for nonlinear programming. Technical report, Rice University, 1991.

[8] J. E. Dennis, Jr. and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Inc., 1983.

[9] Mahmoud El-Alem. *A Global Convergence Theory for the Celis-Dennis-Tapia Trust Region Algorithm for Constrained Optimization.* PhD thesis, Rice University, 1988.

[10] D. M. Gay. Computing optimal locally constrained steps. *SIAM J. Sci. Stat. Comput.*, 2, 1981.

[11] Philip E. Gill, Walter Murray, Michael A. Sanders, and Margaret H. Wright. User's guide for NPSOL (version 4.0): A Fortran package for nonlinear programming. Technical report, Stanford University, 1986.

[12] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Practical Optimization.* Academic Press, 1981.

[13] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, 1983.

[14] Willi Hock and Klaus Schittkowski. *TEST EXAMPLES FOR NONLINEAR PROGRAMMING CODES.* volume 187 of *Lecture Notes in Economics and Mathematical Systems.* Springer-Verlag, 1981.

[15] Jorge J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. Stat. Comput.*, 4(3), 1983.

[16] M. J. D. Powell and Y. Yuan. A trust region algorithm for equality constrained optimization. Technical report, University of Cambridge, 1986.

[17] D. C. Sorensen. Newton's method with a model trust region modification. *SIAM J. Numer. Anal.*, 19(2), 1982.

[18] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis.* Springer-Verlag, 1980.

[19] R. A. Tapia. An introduction to the algorithms and theory of constrained optimization, 1983. Unpublished notes.

[20] A. Vardi. *Trust region strategies for unconstrained and constrained minimization.* PhD thesis, Cornell University, 1980.

[21] A. Vardi. A trust region algorithm for equality constrained minimization: Convergence properties and implementation. *SIAM J. Numer. Anal.*, 22(3), 1985.

[22] Y. Yuan. A dual algorithm for minimizing a quadratic function with two quadratic constraints. Technical Report DAMTP-NA3, University of Cambridge, 1988.

[23] Y. Yuan. On a subproblem of trust region algorithms for constrained optimization. *Math. Prog.*, 47, 1990.

[24] Yin Zhang. Computing a Celis-Dennis-Tapia trust region step for equality constrained optimization. Technical report, Rice University, 1988.