A Trust Region Strategy for Nonlinear Equality Constrained Optimization[1]

by

Maria Rosa Celis

Technical Report 85-4, May 1985

[1]A Thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, Rice University.

A TRUST REGION STRATEGY

FOR

NONLINEAR EQUALITY CONSTRAINED OPTIMIZATION

MARIA ROSA CELIS

ABSTRACT

Many current algorithms for nonlinear constrained optimization problems determine a search direction by solving a quadratic programming subproblem. The global convergence properties are addressed by using a line search technique and a merit function to modify the length of the step obtained from the quadratic program.

In unconstrained optimization, trust region strategies have been very successful. In this thesis we present a new approach for equality constrained optimization problems based on a trust region strategy. The direction selected is not necessarily the solution of the standard quadratic programming subproblem.

ACKNOWLEDGEMENTS

The number of people to whom I owe thanks for this work are far too numerous to be completely mentioned here. To my advisors Richard Tapia and John Dennis I express my deepest gratitude for their advice, guidance and encouragement, not only for this dissertation, but for my entire graduate career. Special thanks to Trond Steihaug for his invaluable assistance, and to Keith Cooper, Randy Allen and Bill LeFebvre for making computers so much easier to deal with. I am extremely grateful to Daniel Woods for his friendship and support, which made my years at Rice so much better. To my parents I express my appreciation for their long and continual support throughout my years as a student. Finally I want to thank Michael Pearlman for everything he did for me.

This work was supported by DOE contract DE-AS05-82-ER13016, ARO contract DAAG-29-83-K-0035, and by the Consejo Nacional de Ciencia y Tecnología (CONACYT), México.

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1

Introduction

By the *general nonlinear programming problem* we mean the constrained optimization problem

$$\begin{aligned} &\underset{x}{\textit{minimize}} && f(x) \\ &\textit{subject to} && g_i(x) = 0, \quad i = 1,2,\ldots,m \\ &&& h_i(x) \geq 0, \quad i = 1,2,\ldots,p \end{aligned} \tag{1.1}$$

where f, g_i, h_i are smooth nonlinear functions defined from $\mathbb{R}^n$ into $\mathbb{R}$. We will refer to problem (1.1) as problem (NLP).

To simplify the study of the problem, it is common practice to consider the cases when only equality or inequality constraints are involved. We will refer to the equality constrained problem

$$\begin{aligned} &\underset{x}{\textit{minimize}} && f(x) \\ &\textit{subject to} && g_i(x) = 0 \quad i = 1,2,\ldots,m, \end{aligned} \tag{1.2}$$

as problem (EQ), and we will refer to the inequality constrained problem

$$\begin{aligned} &\underset{x}{\textit{minimize}} && f(x) \\ &\textit{subject to} && h_i(x) \geq 0 \quad i = 1,2,\ldots,p, \end{aligned} \tag{1.3}$$

as problem (INEQ).

For our research we will consider problem (EQ). We will denote by $g(x)$ the vector whose components are $g_i(x)$, $i = 1,2,\ldots,m$. The Lagrangian function associated with problem (EQ) is the function

$$l(x,\lambda) = f(x) + \lambda^T g(x) \tag{1.4}$$

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

$$L(x,\lambda,C) = f(x) + \lambda^T g(x) + \frac{1}{2} C\, g(x)^T g(x) \tag{1.5}$$

where $C \geq 0$.

It is assumed that the problem functions are at least twice continuously differentiable, that a solution x_* exists and that $\nabla g_i(x_*)$ are linearly independent. A necessary condition for x_* to be a solution to problem (EQ) is that there exists $\lambda_* \in \mathbb{R}^m$ such that (x_*,λ_*) is a solution of the nonlinear system

$$\begin{aligned} \nabla_x l(x,\lambda) &= 0 \\ g(x) &= 0. \end{aligned} \tag{1.6}$$

One of the more successful methods for solving problem (NLP) is the successive quadratic programming (SQP) algorithm. The local convergence properties for SQP have been fairly well established. The area of global convergence is currently receiving much attention. The global convergence properties have been addressed via merit functions and line searches. However it is not at all clear what a proper choice for the merit function should be. Although many functions have been suggested, they usually suffer from either

the fact that they involve unknown parameters with no clear way of choosing them; or there is no connection between the merit function and the way the step is computed.

Trust region approaches have proven to be very successful in unconstrained optimization, and compare quite favorably with existing line search techniques. The purpose of this research is to develop an effective trust region algorithm for solving problem (EQ). Our idea is based on the global convergence theory developed for trust region methods for the unconstrained problem.

At each iteration, the subproblem we solve yields a direction that is not necessarily the solution of the standard quadratic programming problem. Moreover, we will show that there is a relationship between our subproblem and various penalty function methods.

CHAPTER 2

Constrained Optimization Methods

Historically the first methods developed were the penalty function methods described in Section 2.1. Although they are simple and robust, these methods have severe computational disadvantages. These factors led to the development of the multiplier methods of Section 2.2. The latter methods were an improvement over the penalty function methods from a computational point of view, but they still were not very efficient. Because we will utilize several concepts from these methods, they will be described in some detail.

The successive quadratic programming approach of Section 2.3 has proved to be considerably more efficient, and software implementing this approach is starting to become available. Global convergence is guaranteed to some extent by using line search techniques, however, this approach has some shortcomings. Trust region approaches have been proposed for the case of linear constraints, but very little has been done for the nonlinear case.

2.1. Penalty Function Methods.

The penalty function methods attempt to transform the constrained optimization problem into an unconstrained optimization problem by adding to the objective function a term that penalizes constraint violations. The first penalty function

$$P(x,C) = f(x) + \frac{1}{2} C\, g(x)^T g(x) \qquad C \geq 0, \tag{2.1.1}$$

was probably proposed by Courant [1943] for problem (EQ). It can be shown under mild assumptions that if $x(C)$ is a minimizer of $P(x,C)$ for fixed C, then

$$\lim_{C \to \infty} x(C) = x_*,$$

where x_* is a solution to problem (EQ). Based on this fact, the Penalty Function Method can be stated as follows

ALGORITHM 2.1.1. Penalty Function Method.

1) Let x_0 be given, determine $C_0 \geq 0$

2) For $k = 0, 1, 2, \cdots$ until *convergence* do

2.1) find x_{k+1} such that $P(x_{k+1}, C_k) = \min_x P(x, C_k)$

2.2) determine $C_{k+1} > C_k$

Although Courant [1943] proposed the penalty function method, it was developed and popularized mainly by Fiacco and McCormick [1968]. Many others have also contributed to the subject.

The main convergence result for the penalty function method is due to Polyak [1971]. This result establishes that for C sufficiently large, the penalty function (2.1.1) has a locally unique minimizer $x(C)$. Furthermore, there exists a constant $M > 0$ such that

$$\| x(C) - x_* \| \leq \frac{M}{C}$$

and

$$\| C\, g(x(C)) - \lambda_* \| \leq \frac{M}{C},$$

where x_* is the solution to problem (EQ) and λ_* is its associated multiplier.

The algorithm is easy to program and the theory is quite attractive. However, in practice, a straightforward implementation gives rise to serious numerical difficulties. As $C \to \infty$, the Hessian matrix $\nabla^2 P(x,C)$ becomes increasingly ill-conditioned. Consequently the step $x_{k+1} - x_k$ becomes increasingly badly determined. Thus it is only possible to obtain low accuracy. Also, it is not at all clear how to choose the penalty constant.

It is important to point out that the penalty function method is not really an iterative procedure, i.e., x_{k+1} does not depend on x_k, unless the choice of C_{k+1} depends on x_k. The penalty constant C actually plays a role analogous to the mesh spacing in the solution of differential equations by finite differences. Specifically, by choosing a large initial penalty constant, we can get arbitrarily good accuracy. Then one might ask, why should we minimize $P(x,C)$ for various values of C, since we only need to minimize $P(x,C)$ for the largest value

of C we are interested in? The answer to this question is that it is not clear what that optimal value of C should be and the numerical conditioning of the problem enters in as it does in finite differences.

We have established that the penalty function method does not fall into the framework of a standard iterative procedure, therefore the question of local convergence has no meaning. A good initial estimate of x_* is not helpful. If the initial estimate for C is small, the minimization problem will mainly deal with the objective function f and not with satisfying the constraints. Therefore, x_1 is usually far away from x_*. In addition, from a theoretical point of view, we can obtain convergence of any order because each iterate depends only on the penalty constant and we are free to choose the penalty constant. However, this claim would be impossible to demonstrate using finite precision arithmetic. As we mentioned before, the numerical conditioning of the Hessian matrix $\nabla^2_{xx}P(x,C)$ becomes arbitrarily bad for large C.

The penalty function method can be extended from problem (EQ) to problem (NLP). Each inequality constraint $h_i(x) \geq 0$, $i = 1,2, \ldots, p$ is replaced with the equivalent equality constraint

$$\min\,(0, h_i(x)) = 0. \tag{2.1.2}$$

Notice that (2.1.2) is not differentiable. However, $[\min\,(0, h_i(x))]^2$ is reasonably smooth, so the analysis carries over for the general problem (NLP).

At this point, a natural question to ask is whether constrained optimization problems can be solved by performing a single unconstrained optimization. This is possible if we have a real function with the property that the solution x_* to the constrained problem is a local minimizer for this function. Functions with this property are called *exact penalty functions.*

For problem (INEQ), Zangwill [1967] considered the following penalty function

$$P(x,C) \;=\; f(x) - C\sum_{i=1}^{p} \min(0, h_i(x)), \tag{2.1.3}$$

and proved that $P(x,C)$ was an exact penalty function for convex problems. Later on, Pietrzykowski [1969] extended Zangwill's result to nonconvex problems.

One of the main disadvantages of the exact penalty function method of Zangwill and Pietrzykowski is that (2.1.3) is not differentiable at the solution x_*. Consequently, efficient unconstrained optimization techniques that involve derivative information can not be used directly. Another disadvantage is that although the optimal value C_* exists, in practice it can only be computed after solving the problem itself, because it depends on $f(x_*)$.

Fletcher [1970] considered the following penalty function for problem (EQ):

$$P(x,C) \;=\; f(x) + \lambda(x,C)^T g(x) \tag{2.1.4}$$

where

$$\lambda(x,C) = (\nabla g(x)^T \nabla g(x))^{-1}(g(x) - C\nabla g(x)^T \nabla f(x))$$

and proved that for C sufficiently large, x_* is a local minimizer of $P(x,C)$.

As before, the problem of finding C so that a single unconstrained optimization will yield x_* still remains. Another drawback of Fletcher's method is the fact that $P(x,C)$ has first order terms, therefore its gradient will involve second order terms and its Hessian will involve third order terms.

More recently, exact penalty functions have become of interest again, but with some slight differences. Consider a function $P:\mathbb{R}^{n+m} \to \mathbb{R}$ that has the property that one of its local minimizers, say (x_*,λ_*), is such that x_* is a solution to problem (EQ) with associated multiplier λ_*. This function is called an *exact extended penalty function* for problem (EQ).

The obvious choices for extended penalty functions are the Lagrangian function

$$l(x,\lambda) = f(x) + \lambda^T g(x)\,,$$

or the augmented Lagrangian function

$$L(x,\lambda,C) = f(x) + \lambda^T g(x) + \frac{1}{2} C g(x)^T g(x)\,.$$

Both functions have stationary points (with respect to the variables x and λ) which correspond to local solutions (and their associated multipliers of problem (EQ)). These stationary points are not local minimizers since both functions are

linear in λ, so that their Hessian matrices are not positive definite for any (x,λ).

DiPillo and Grippo [1979], and independently Boggs and Tolle [1980], and Boggs, Tolle and Wang [1982] consider the functional

$$P(x,\lambda,C,D) = l(x,\lambda) + \frac{C}{2} g(x)^T g(x) + \frac{D}{2} \nabla_x l(x,\lambda)^T Q(x) \nabla_x l(x,\lambda) \qquad (2.1.5)$$

where C and D are scalars and $Q(x)$ is a weighting matrix function. This penalty function takes the augmented Lagrangian and adds a term that is quadratic in λ. DiPillo and Grippo [1979] give general conditions which guarantee that P is an exact extended penalty function.

The penalty function method based on (2.1.5) does not take care of all of the drawbacks of previous penalty function methods. As before, it is not clear how the penalty constants should be chosen. Like Fletcher's method, the function involves first order terms, which means that its gradient will involve second order terms and its Hessian will involve third order terms. In addition, the dimension of the optimization problem is extended from n to $n+m$.

2.2. Multiplier Methods.

One of the major difficulties in the penalty function methods is that of ill-conditioning as the constants become large. Therefore it would be useful to derive methods for which the parameters need only to assume moderate values.

These concerns lead us to the multiplier method. It consists of updating the Lagrange multipliers λ_i at each iteration and sometimes the penalty

constant as well. The iteration can be stated as follows:

ALGORITHM 2.2.1. The Multiplier Method.

1) Let x_0 be given, determine λ_0, $C_0 > 0$

2) For $k = 0,1,2,\cdots$ until *convergence* do

2.1) find x_{k+1} such that $L(x_{k+1},\lambda_k,C_k) = \min_x L(x,\lambda_k,C_k)$

2.2) determine $C_{k+1} = \Pi(x_k,\lambda_k,C_k)$

$$\lambda_{k+1} = U(x_k,\lambda_k,C_k)$$

where $\Pi : \mathbb{R}^{n+m+1} \to \mathbb{R}$ is said to be the penalty constant update formula and $U : \mathbb{R}^{n+m+1} \to \mathbb{R}$ is one of the following multiplier update formulas:

$$\begin{aligned}
U_{HP}(x,\lambda,C) &= \lambda + Cg(x) \\
U_P(x,\lambda,C) &= -[g(x)^T g(x)]^{-1} \nabla g(x)^T \nabla f(x) \\
U_B(x,\lambda,C) &= \lambda + [\nabla g(x)^T B^{-1} \nabla g(x)]^{-1} g(x) \\
U_*(x,\lambda,C) &= [\nabla g(x)^T B^{-1} \nabla g(x)]^{-1} [g(x) - \nabla g(x)^T B^{-1} (\nabla f(x) + C \nabla g(x) g(x))] ,
\end{aligned}$$

where

$$B = \nabla^2_{xx} L(x,\lambda,C) ,$$

or more generally

$$U_{GUF}(x,\lambda,C) = \lambda + [\nabla g(x)^T D \nabla g(x) + A]^{-1} [g(x) - \nabla g(x)^T D \nabla_x L(x,\lambda,C)] ,$$

where A and D are $m \times m$ and $n \times n$ matrices respectively, which may depend on x, λ, and C.

The multiplier method using U_{HP} was proposed independently by Hestenes [1969] and Powell [1969]. Haarhoff and Buys [1970] proposed the multiplier method using the update formula U_P. Formula U_B was first used by Buys [1972]. Formulas U_* and U_{GUF} with $A = 0$ are special cases of a general theory developed by Tapia [1969], [1974a], [1974b], [1977] for transforming a constrained problem into an unconstrained problem of the same dimension.

Bertsekas [1976] generalized Polyak's result to include the multiplier method using U_{HP}. The rate of convergence of the multiplier method with a fixed penalty constant can be shown to be q-linear in x and in λ. Additional results show that it is possible to choose $\{C_k\}$ so that $C_k \to \infty$ and the multiplier method with penalty constants $\{C_k\}$ is convergent in x and λ. Moreover, the convergence is q-superlinear in λ if and only if $C_k \to \infty$. Tapia [1977] presents a complete analysis of the multiplier method.

Extensive computational experience with the multiplier method was reported by Miele and his coworkers, [1971a], [1971b], [1972a], [1972b].

2.3. The SQP Method and Equivalent Formulations.

To guarantee convergence of the penalty function method, the penalty constant must go to infinity and the problem becomes increasingly ill-conditioned. To guarantee fast convergence for the multiplier method, again the penalty constant must go to infinity and the problem becomes increasingly ill-conditioned. To address these problems, an algorithm which would give fast

convergence without becoming ill-conditioned is needed. In this section we present one such algorithm.

From a historical point of view, three different philosophies for extending quasi-Newton methods from unconstrained optimization to constrained optimization have been explored. These philosophies consist of the extended problem approach, the diagonalized multiplier method, and the successive quadratic programming (SQP) approach. Tapia [1978] showed that these three approaches were equivalent for problem (EQ).

Following Tapia [1977], [1978], by the *extended problem* we mean the problem of finding a stationary point of the augmented Lagrangian function $L(x,\lambda,C)$ given by (1.5), i.e., solving the nonlinear system

$$\nabla L(x,\lambda,C) = 0\,, \qquad (C \geq 0) \tag{2.3.1}$$

where

$$\nabla L(x,\lambda,C) = \begin{bmatrix} \nabla_x L(x,\lambda,C) \\ \nabla_\lambda L(x,\lambda,C) \end{bmatrix} = \begin{bmatrix} \nabla f(x) + \nabla g(x)(\lambda + C\, g(x)) \\ g(x) \end{bmatrix}$$

Notice that we are allowing $C=0$, so the augmented Lagrangian includes the Lagrangian function as a special case.

From a theoretical point of view, the extended problem plays a very important role and has been in the background of the derivation of many algorithms. This formulation is widely used for the convergence analysis of its equivalent methods.

Consider applying Newton's method to solve the nonlinear system (2.3.1). The iteration is given by

$$\begin{bmatrix} x_{k+1} \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ \lambda_k \end{bmatrix} - \begin{bmatrix} B_k & \nabla g(x_k) \\ \nabla g(x_k)^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \nabla_x L(x_k,\lambda_k,C) \\ g(x_k) \end{bmatrix} \tag{2.3.2}$$

where $B_k = \nabla^2_{xx} L(x_k,\lambda_k,C)$. We will refer to the procedure based on iteration (2.3.2) as *Newton's method for the extended problem.* Fletcher [1981] refers to this method as the Solver Method. Q-quadratic convergence in (x,λ) can be proved under the standard assumptions for convergence of Newton's method, see Tapia [1977].

The straightforward quasi-Newton method applied to problem (2.3.2) would consist of approximating the entire Hessian matrix $\nabla L(x,\lambda,C)$. However, it seems inefficient to approximate first order information that has already been calculated, or even worse, to approximate the zero component of $\nabla L(x,\lambda,C)$. Therefore we approximate only the component of $\nabla L(x,\lambda,C)$ that contains second order information. Tapia [1977] refers to this method as a *structured quasi-Newton method for the extended problem*, and the iteration is given by (2.3.2) using an approximation to $\nabla^2_{xx} L(x_k,\lambda_k,C)$ as B_k. Q-superlinear convergence in (x,λ) can be demonstrated using Broyden, PSB, DFP and BFGS update formulas, see Tapia [1977].

At each iteration, the multiplier method described in Section 2.2 goes through a complete minimization step for x and only one update for λ, although

we are solving for both the minimizer x_* and its associated multiplier λ_*. It would then make sense to update the estimate of the multiplier after each update of the estimate of the minimizer. This procedure is referred to as the *diagonalized multiplier method*, and is given by

ALGORITHM 2.3.1. The Diagonalized Multiplier Method.

1) Let x_0 be given, determine λ_0, $C \geq 0$

2) For $k = 0,1,2,\cdots$ until *convergence* do

2.1) calculate $\lambda_{k+1} = U(x_k,\lambda_k,C)$

2.2) calculate $x_{k+1} = x_k - B_k^{-1}\nabla_x L(x_k,\lambda_{k+1},C)$

where U is one of the multiplier update formulas given in Section 2.2. If $B_k = \nabla^2_{xx} L(x_k,\lambda_k,C)$ we have the *diagonalized Newton multiplier method.* The *diagonalized secant multiplier methods* result from choosing B_k to be a secant update formula.

Tapia [1977] showed that the diagonalized Newton multiplier method using U_* is equivalent to Newton's method on the extended problem. Moreover, a diagonalized secant multiplier method using U_* is equivalent to a structured secant method for the extended problem.

Let us now consider the following quadratic programming problem

$$\begin{aligned} &\underset{s}{\text{minimize}} \quad q(s) = \nabla F(x)^T s + \tfrac{1}{2} s^T B s \\ &\text{subject to} \quad g(x) + \nabla g(x)^T s = 0\,, \end{aligned} \tag{2.3.3}$$

where B is the Hessian of the augmented Lagrangian $L(x,\lambda,C)$ or an

approximation to it, and

$$F(x) = f(x) + \tfrac{1}{2} C\, g(x)^T g(x)\,, \qquad (C \geq 0)\,.$$

By a *successive quadratic programming (SQP) method* for problem (EQ), we mean the iterative procedure

ALGORITHM 2.3.2. The Successive Quadratic Programming Method.

1) Let x_0 be given, determine λ_0, $C \geq 0$

2) For $k = 0,1,2,\cdots$ until *convergence* do

2.1) find a solution s_k, λ^{QP} to problem (2.3.3)

2.2) $x_{k+1} = x_k + s_k$

2.3) $\lambda_{k+1} = \lambda^{QP}$.

The successive quadratic programming method for general problems using the exact Hessian in the quadratic term was presented by Wilson [1963]. Quasi-Newton approximations to the Hessian were introduced by Garcia-Palomares and Mangasarian [1976]. The quasi-Newton SQP method was refined by Han [1976] and other refinements were later added by Powell [1977a], [1977b]. Most previous versions use $C = 0$ (hence F in (2.3.3) reduces to f).

The three philosophies described in this section are equivalent. Specifically, Tapia [1977] showed that for problem (EQ), the extended problem with L, the diagonalized multiplier method using U_*, and the successive quadratic programming method generate identical (x,λ) iterates.

Of these three equivalent formulations, the SQP philosophy is the most visible and popular. The main reason for its popularity is that it allows inclusion of inequality constraints in a straightforward manner. To do so, one merely carries them along as linearized inequalities in the quadratic program. Another reason for its popularity is that the SQP approach allows use of existing quadratic programming modules in its implementation.

Although the local convergence theory has been fairly well established, the issue of global convergence has not at all been determined. Numerous authors have addressed the global convergence properties via merit functions and line searches. Specifically, given a merit function $\Phi(x,\lambda,C)$, and a search direction s obtained as the solution of the quadratic programming problem (2.3.3), a step αs $(0<\alpha\leq 1)$ that gives sufficient decrease in $\Phi(x,\lambda,C)$ is taken. For fast local convergence the choice $\alpha = 1$ is dictated by the theory. However, if x is far from the solution, this choice will not in general guarantee convergence, see Maratos [1978].

The merit function plays an important role in determining the length of the step, and in determining whether the step chosen yields an improvement towards the solution from the current point. In the unconstrained case, a merit function is readily available, namely the function itself. For the constrained problem the choice of a merit function is not at all clear. There is no natural descent function available because of the conflicts between the desire to decrease the objective function and the desire to satisfy the constraints. Many

suggestions for Φ can be found in the literature. In particular when dealing with problem (EQ), Han [1977], Powell [1977a] and Fletcher [1981] consider the l_1 penalty function

$$\Phi(x,\lambda,C) \;=\; f(x) + \sum_{i=1}^{m} |\, g_i(x)| \;. \tag{2.3.4}$$

Bartholomew-Biggs [1980] considers the standard penalty function

$$\Phi(x,\lambda,C) \;=\; f(x) + \frac{1}{2}\, C\, g(x)^T g(x)\,, \tag{2.3.5}$$

Tapia [1977] considers the square of the 2-norm of the gradient of the Lagrangian and the constraint error

$$\Phi(x,\lambda) \;=\; ||\, \nabla f(x) + \nabla g(x)\lambda \,||_2^2 \,+\, ||\, g(x)\,||_2^2\,, \tag{2.3.6}$$

and Schittkowski [1981], [1982], and Gill, Murray, Saunders and Wright [1983] consider the augmented Lagrangian

$$\Phi(x,\lambda,C) \;=\; f(x) + \lambda^T g(x) + \frac{1}{2}\, C\, g(x)^T g(x)\,. \tag{2.3.7}$$

Finally there is the penalty function of DiPillo and Grippo [1979] and Boggs, Tolle and Wang [1980]

$$\Phi(x,\lambda,C,D) = f(x)+\lambda^T g(x)+\tfrac{1}{2}Cg(x)^T g(x)+\tfrac{1}{2}D\nabla_x l(x,\lambda)^T\nabla_x l(x,\lambda)\,. \tag{2.3.8}$$

These merit functions can all be extended to problems with inequality constraints in one of the standard ways.

A main problem with these merit functions is that there are no theoretical results that establish a connection between the descent function and the step

generated by the SQP method. Moreover, many of these merit functions involve unknown constants or parameters which affect the descent properties, and there is no clear way of choosing them.

In unconstrained optimization, trust region strategies have been very successful. Several authors including Sorensen [1982b] and Gay [1983] have extended the trust region approach to linearly constrained optimization problems. From a theoretical point of view, extending a trust region approach from unconstrained to linearly constrained problems is somewhat straightforward, one merely focuses on the subspace of interest. The choice of merit function in the case of linear constraints is also quite natural, one uses the objective function restricted to the subspace of interest. From a practical point of view, these extensions still require numerical experimentation.

For nonlinear constraints the extension is not at all clear. The main attempt in generalizing the trust region approach to nonlinear equality constraints has been Vardi [1980]. He considers adding a constraint on the size of the step to problem (2.3.3), i.e., the step is given by the solution to problem

$$\begin{aligned} \underset{s}{\textit{minimize}} \quad & \nabla_x L(x,\lambda,C)^T s + \tfrac{1}{2} s^T B s \\ \textit{subject to} \quad & g(x) + \nabla g(x)^T s = 0 \\ & \|s\|_2 \leq \Delta , \end{aligned} \tag{2.3.9}$$

where Δ is a positive number. Immediately one problem arises, the set of steps that both satisfy the linearized constraints and are inside the trust region may be empty. To overcome this problem, Vardi considers modifying the linearized

constraints to

$$\alpha\, g(x) + \nabla g(x)^T s = 0 ,$$

where $0 < \alpha \leq 1$ depends on the radius Δ. The choice of α is very important because it determines how feasible and how optimal in f the step will be. However, there seems to be no natural choice for α given by this approach, and the way in which the parameter is determined is not at all clear.

CHAPTER 3

The Model Trust Region Approach For Unconstrained Optimization

The basic idea behind trust region methods is to estimate a region in which a local model of the problem can be trusted to adequately represent the problem. Given this estimate, one takes the step that optimizes the model in this region. The step is then accepted or rejected, and the trust region is updated depending on the performance of the model in the region. The idea of defining a region of trust for the search direction was first suggested by Levenberg [1944] and Marquardt [1963] for nonlinear least-squares problems. The application of the technique to the general unconstrained optimization problem was suggested by Goldfeld, Quandt and Trotter [1966]. A detailed introduction to the trust region approach can be found in Chapter 6 of Dennis and Schnabel [1983].

The preliminary implementation of our approach will be based on one specific trust region algorithm, which we describe in this chapter. A step selection strategy is discussed in Section 3.1, and a trust region updating strategy is the topic of Section 3.2. Our presentation will follow Dennis and Schnabel [1983]. The motivation for our trust region approach for the

constrained problem is based on the global convergence theory developed for trust region methods for the unconstrained problem. We will briefly summarize these convergence results in Section 3.3.

3.1. A Trust Region Step

Suppose that we have a current point x_c and some estimate of the maximum length of a step, Δ_c that we are likely to take from x_c. The optimal step strategy, proposed by Hebden [1973], takes the solution s_* of problem

$$\begin{aligned} &\underset{s}{\textit{minimize}} \quad q_c(s) = \nabla f(x_c)^T s + \tfrac{1}{2} s^T B_c s \\ &\textit{subject to} \quad \| s \|_2 \leq \Delta_c \end{aligned} \tag{3.1.1}$$

where B_c is the Hessian of the function at x_c or some approximation to it. It then tries the step to obtain a new point $x_+ = x_c + s_*$. Discussions and modifications of this strategy have been presented by Moré [1977], Gay [1981], Sorensen [1982a], Moré and Sorensen [1983], and Schultz, Schnabel and Byrd [1985].

If B_c is positive definite and $\| B_c^{-1} \nabla f(x_c) \| \leq \Delta_c$, then $s_* = -B_c^{-1} \nabla f(x_c)$ is the solution to problem (3.1.1). Otherwise the solution to problem (3.1.1) is the solution of

$$(B_c + \mu I) s_* = -\nabla f(x_c), \tag{3.1.2}$$

with $B_c + \mu I$ positive semidefinite, $\mu \geq 0$, and $\mu(\Delta_c - \| s_* \|) = 0$. If B_c is positive definite, then so is $B_c + \mu I$, and $s_* = -(B_c + \mu I)^{-1} \nabla f(x_c)$, where μ is

uniquely determined by the solution of $||s_*|| = \Delta_c$. If B_c has negative or zero eigenvalues, s_* is still a solution of (3.1.2). If in addition, $\nabla f(x_c)$ is orthogonal to the null space of $(B_c - \lambda_1 I)$ and $||(B_c - \lambda_1 I)^+ \nabla f(x_c)|| < \Delta_c$, where the superscript $^+$ denotes the pseudoinverse and λ_1 is the most negative eigenvalue of B_c, then s_* is the particular solution of (3.1.2) given by $s_* = -(B_c + \mu I)^+ \nabla f(x_c) + \tau_c v_c$ where v_c is the eigenvector of B_c corresponding to λ_1, and τ_c is chosen so that $||s_*|| = \Delta_c$. This latter case is labeled by Moré and Sorensen [1983] as the "hard case".

Let $s(\mu) = -(B + \mu I)^{-1} \nabla f(x_c)$. To compute $\mu > 0$ so that $||s(\mu)|| = \Delta_c$, we find a solution to the scalar equation

$$\phi(\mu) = ||s(\mu)|| - \Delta_c = 0 . \tag{3.1.3}$$

This is done iteratively by forming a local model for problem (3.1.3), and finding a zero of this model. The one dimensional version of (3.1.3) suggests we use a local model of the form

$$m_c(\mu) = \frac{\alpha}{\beta + \mu} - \Delta_c \tag{3.1.4}$$

with free parameters α and β. It is reasonable to choose α and β to satisfy the two conditions:

$$\begin{aligned} m_c(\mu) &= \frac{\alpha}{\beta+\mu} - \Delta_c = \phi(\mu) = ||s(\mu)||_2 - \Delta_c \\ \text{and} \qquad & \qquad (3.1.5) \\ m_c'(\mu) &= -\frac{\alpha}{(\beta+\mu)^2} = \phi'(\mu) = -\frac{s(\mu)^T (B_c + \mu I)^{-1} s(\mu)}{||s(\mu)||_2} . \end{aligned}$$

This gives:

$$\begin{aligned} \alpha &= -\frac{(\phi(\mu) + \Delta_c)^2}{\phi'(\mu)} , \\ \beta &= -\frac{(\phi(\mu) + \Delta_c)^2}{\phi'(\mu)} - \mu . \end{aligned} \tag{3.1.6}$$

Now that we have our model, we update μ by asking it to satisfy $m(\mu) = 0$, i.e. $\mu = \frac{\alpha}{\Delta_c} - \beta$. Therefore the correction

$$\mu_+ = \mu - \left[\frac{||s(\mu)||}{\Delta_c}\right]\left[\frac{\phi(\mu)}{\phi'(\mu)}\right] \tag{3.1.7}$$

determines the iteration for solving $\phi(\mu) = 0$.

It is necessary to safeguard μ in order to transform (3.1.7) into a useful computational algorithm. For details of various schemes, see Hebden [1973], Moré [1977], Gay [1981] and Fletcher [1980]. The hard case must also be identified and a solution computed. A complete and detailed discussion can be found in Gay [1981], and Moré and Sorensen [1983].

3.2. Accepting the Step and Updating the Trust Region.

Once we have calculated a step s_c, we need to decide whether the point $x_+ = x_c + s_c$ is satisfactory as the next iterate. If x_+ is not acceptable, we reduce the size of the trust region and minimize the same quadratic model over the smaller region. When an acceptable x_+ is found, we must then decide whether the trust region should be increased, decreased or kept the same for the

next step.

The condition for accepting x_+ is quite simple, we require sufficient decrease in the function, i.e.,

$$f(x_+) \leq f(x_c) + \alpha \nabla f(x_c)^T (x_+ - x_c) \qquad (3.2.1)$$

where α is a constant in $(0, \frac{1}{2})$. If x_+ does not satisfy (3.2.1), we reduce the trust region and return to finding a smaller trust region step.

Now assume we have x_+ that satisfies (3.2.1) and that Δ_c was not halved in the process of finding s_c. If s_c is not the Newton step, then before we settle for x_+, we first ask whether we should try a larger step from x_c, using the current model. This may be useful if the trust region has become small in previous iterations or started small, since it may avoid extra evaluations of the gradient and the Hessian by allowing the trust region to grow more rapidly. In order to make this decision, we compare the actual reduction

$$\Delta f \equiv f(x_+) - f(x_c) \qquad (3.2.2)$$

to the predicted reduction

$$\Delta f_{pred} \equiv q_c(x_+ - x_c) \,. \qquad (3.2.3)$$

If the agreement is very good, i.e.,

$$| \Delta f_{pred} - \Delta f | \leq \beta | \Delta f |$$

where $\beta = 0.1$ is a typical value, or the actual reduction in f is very large, i.e.,

$$f(x_+) \leq f(x_c) + \nabla f(x_c)^T s_c \,,$$

we increase the radius of the trust region. In both these cases, we save x_+ and $f(x_+)$, but instead of moving directly to x_+, we double Δ_c, and compute a new s_c using our current model with the larger trust region. This technique is sometimes referred to as internal doubling, and in practice can save a significant number of derivative evaluations.

Now suppose that we have decided to accept x_+ as our next iterate. We then need to update Δ_c. The decision is based on whether our current quadratic model is predicting the function well. If the quadratic model predicted the actual decrease in the function sufficiently well, i.e.,

$$\Delta f \leq \sigma_2 \Delta f_{pred} \,,$$

then we increase the trust region for the next iteration. If the model greatly overestimated the decrease in the function, i.e.,

$$\Delta f > \sigma_1 \Delta f_{pred} \,,$$

where $0 < \sigma_1 < \sigma_2 < 1$, then we decrease the trust region. Otherwise we leave it the same. Dennis and Schnabel [1983] suggest using $\sigma_1 = 0.25$ and $\sigma_2 = 0.75$.

To modify the trust region radius, we follow the policy of decreasing with respect to the length of the step s_c calculated, and increasing with respect to the previous trust region radius. Specifically, to reduce the trust region we choose $\Delta \in [\tau_1 \| s_c \|, \tau_2 \| s_c \|]$, where $0 < \tau_1 < \tau_2 < 1$. To increase the trust region, we

region, we choose $\Delta = \tau_3 \Delta_c$. Typical values suggested for the constants are $\tau_1 = 0.1$, $\tau_2 = 0.5$, and $\tau_3 = 2$.

3.3. Convergence Results

Powerful convergence results for trust region approaches have been developed. Powell [1970], [1975] and Thomas [1975] discuss the convergence properties of a class of algorithms. Sorensen [1982a] proves strong convergence properties for a specific trust region algorithm which uses second order information. Related results can be found in Fletcher [1980] and Moré and Sorensen [1983]. Schultz, Schnabel and Byrd [1985] give general conditions under which a family of trust region algorithms are globally convergent.

Our discussion summarizes the Schultz, Schnabel and Byrd [1985] results. They give general conditions which the computed step must satisfy. These conditions are as follows.

(1) The step must give sufficient decrease of the quadratic model.

(2) If the Hessian is indefinite, the step should give as good a decrease of the quadratic model as a direction of sufficient negative curvature.

(3) If B is positive definite and the Newton step lies inside the trust region, then the Newton step is chosen.

Conditions (1), (2), and (3) are a generalization of the properties of the optimal step that are assumed by Fletcher [1980]. Conditions (1) and (2) can be shown to be a strict relaxation of Moré and Sorensen's condition that the step

calculated give a predicted decrease at least some fixed fraction of the optimal decrease in the quadratic model.

Given these conditions, Schultz, Schnabel and Byrd [1985] prove the following convergence results under standard assumptions.

(I) If the steps s_k satisfy Condition (1) for all k, then $\nabla f(x_k) \to 0$.

(II) If s_k satisfies Conditions (1) and (3), $B_k = \nabla^2 f(x_k)$ for all k, and the sequence $\{x_k\}$ has a limit point x_*, with $\nabla^2 f(x_*)$ positive definite, then $\{x_k\}$ converges q-quadratically.

(III) If s_k satisfies Conditions (1) and (2), $B_k = \nabla^2 f(x_k)$ for all k, and $x_k \to x_*$, then $\nabla^2 f(x_*)$ is positive semidefinite.

The first order results extend those of Powell [1970] and Thomas [1975], and are a generalization of Powell [1975]. The rest are a generalization of Sorensen [1982a].

CHAPTER 4

A Trust Region Approach for Constrained Optimization

Trust region approaches for unconstrained optimization have proven to be very successful both theoretically and practically. These approaches compare quite favorably with existing line search techniques. In particular, the theoretical results are most satisfactory and have motivated considerable experimental code.

In this chapter we develop a trust region approach for the general nonlinear equality constrained optimization problem. Section 4.1 motivates our approach and defines a trust region step for problem (EQ). The choice of merit function for deciding when to accept a step and how to update the trust region are the topics of Section 4.2.

4.1. A Constrained Trust Region Step.

One of the more successful methods for solving problem (EQ) is the successive quadratic programming (SQP) approach described in Section 2.3. At each iteration, the step is calculated as the solution of the quadratic programming problem

$$\begin{aligned} &\underset{s}{\textit{minimize}} \quad q_{QP}(s) = \nabla f(x)^T s + \tfrac{1}{2} s^T B\, s \\ &\textit{subject to} \quad g(x) + \nabla g(x)^T s \;= 0\,, \end{aligned} \tag{4.1.1}$$

where $\lambda \in \mathbb{R}^m$, and B is $\nabla^2_{xx} l(x,\lambda)$ or some approximation to it.

The most natural way to introduce the trust region idea is to add a constraint which restricts the size of the step in problem (QP), see Vardi [1980]. However, this approach may lead to inconsistent constraints, and it is not clear how to overcome this difficulty. Instead of adding the trust region constraint to the standard (QP) problem, we consider adding it to a somewhat different problem.

Suppose we want to solve $g(x)=0$ using a standard trust region method. We have a current point x_c and a bound Δ_c on the length of the step we are willing to take from x_c. At each iteration the step is calculated by solving :

$$\begin{aligned} &\underset{s}{\textit{minimize}} \quad \tfrac{1}{2} \| \; g(x_c) + \nabla g(x_c)^T s \; \|_2^2 \\ &\textit{subject to} \quad \| \; s \; \|_2 \leq \Delta_c \; . \end{aligned} \tag{4.1.2}$$

We use $\| \, g(x_c) + \nabla g(x_c)^T s \, \|_2^2$ instead of a full Newton quadratic model of $\| \, g(x_c) \, \|_2^2$, where by a full Newton quadratic model we mean

$$g(x_c)^T g(x_c) + 2 \nabla g(x_c)^T s + s^T (\nabla g(x_c) \nabla g(x_c)^T + \nabla^2 g(x_c)\, g(x_c)) s \; .$$

The two expressions differ only in the term $\nabla^2 g(x_c)\, g(x_c)$, included by the Newton model in the Hessian matrix $\nabla g(x_c) \nabla g(x_c)^T + \nabla^2 g(x_c)\, g(x_c)$, but omitted in $\| \, g(x_c) + \nabla g(x_c)^T s \, \|_2^2$. Since we are solving the problem $g(x)=0$, we have that at the solution x_*, the term $\nabla^2 g(x_*)\, g(x_*) = 0$. Therefore using

$|| g(x_c)+\nabla g(x_c)^T s ||_2^2$ we will have the same convergence properties as we would using the Newton model, and we avoid using second order information of g.

If the algorithm simply took the steepest descent step for the model, i.e. the Cauchy step s^{CP}, then under reasonable assumptions, first order convergence can be demonstrated. To formalize this result, we apply the theory of Schultz, Schnabel and Byrd [1985] to the problem of minimizing $F(x) = \frac{1}{2} g(x)^T g(x)$. Before stating the theorem, we make the following definitions. Let the step at each iteration k be $s_k = x_{k+1} - x_k$, and

$$pred_k(s) \;=\; -\nabla F(x_k)^T s - \frac{1}{2} s^T B_k s \,,$$

where $\nabla F(x_k) = \nabla g(x_k) g(x_k)$, $B_k = \nabla g(x_k) \nabla g(x_k)^T$, and $k = 0, 1, 2, \cdots$. The theorem can now be stated as follows.

THEOREM 4.1.1. *Let $g : \mathbb{R}^n \to \mathbb{R}^m$ be twice continuously differentiable, $F(x)$ as defined above, and the Hessian of $F(x)$ bounded on the whole space. Let $\{x_k\}$, $x_k \in \mathbb{R}^n$, $k = 0, 1, 2, \ldots$, be a sequence of points generated by applying a standard trust region algorithm to the problem of minimizing $F(x)$, with starting point $x_0 \in \mathbb{R}^n$. If at each iteration the step s_k satisfies*

$$pred_k(s_k) \;\geq\; pred_k(s^{CP})$$

and B_k is uniformly bounded in k, then $\nabla F(x_k)$ converges to 0.

Proof. By assumption, the steps s_k are chosen to give as much descent as the Cauchy step s^{CP}, i.e. $pred_k(s_k) \geq pred_k(s^{CP})$. So by Lemma 4.1 in Schultz, Schnabel and Byrd [1985], we can conclude that

$$pred_k(s_k) \;\geq\; \frac{1}{2} || \nabla g(x_k) g(x_k) || \min \left\{ \Delta, \frac{|| \nabla g(x_k) g(x_k) ||}{|| \nabla g(x_k) \nabla g(x_k)^T ||} \right\} .$$

Given this result, the proof of the theorem is the same as part I of Theorem 2.2 in Schultz, Schnabel and Byrd [1985]. □

The following corollary is an immediate consequence of Theorem 4.1.1.

COROLLARY 4.1.2. *Any limit point x_* of $\{x_k\}$ such that $\nabla g(x_*)$ has full rank, is a feasible point of the constraints g.*

Proof. The proof is straightforward by observing that $\nabla F(x_*) = \nabla g(x_*) g(x_*)$.

□

Theorem 4.1.1 motivates our trust region approach for constrained optimization. We will base our algorithm on choosing steps that satisfy the condition

$$|| g(x_c) + \nabla g(x_c)^T s ||_2 \;\leq\; || g(x_c) + \nabla g(x_c)^T s^{CP} ||_2 \,. \qquad (4.1.3)$$

This means that the step must be at least as linear feasible as the Cauchy step

s^{CP} for minimizing $\frac{1}{2}g(x)^T g(x)$. We will refer to Theorem 4.1.1 as the fundamental theorem for our trust region strategy.

Define the set S_{CP} as

$$S_{CP} = \{ s : \| s \|_2 \leq \Delta_c \text{ and } \| g(x_c) + \nabla g(x_c)^T s \|_2 \leq \| g(x_c) + \nabla g(x_c)^T s^{CP} \|_2 \} \quad (4.1.4)$$

That is, S_{CP} is the set of steps from x_c that are inside the trust region and give at least as much descent on the 2-norm of the residuals of the linearized constraints as the Cauchy step, (see Figure 4.1). By choosing any point in S_{CP} at each iteration, we will generate a sequence $\{x_k\}$ which under reasonable assumptions, should converge to a feasible point. We take advantage of this freedom by choosing at each iteration a step s which minimizes some quadratic model of the objective function f over S_{CP}. The step is calculated by solving the problem

$$\begin{aligned} &\textit{minimize}_{s} \quad q_c(s) \\ &\textit{subject to} \quad \| s \|_2 \leq \Delta_c \\ &\qquad\qquad \| g(x_c) + \nabla g(x_c)^T s \|_2 \leq \Theta_c, \end{aligned} \quad (4.1.5)$$

where $q_c(s)$ is a quadratic approximation to the function f and $\Theta_c = \| g(x_c) + \nabla g(x_c)^T s^{CP} \|_2$.

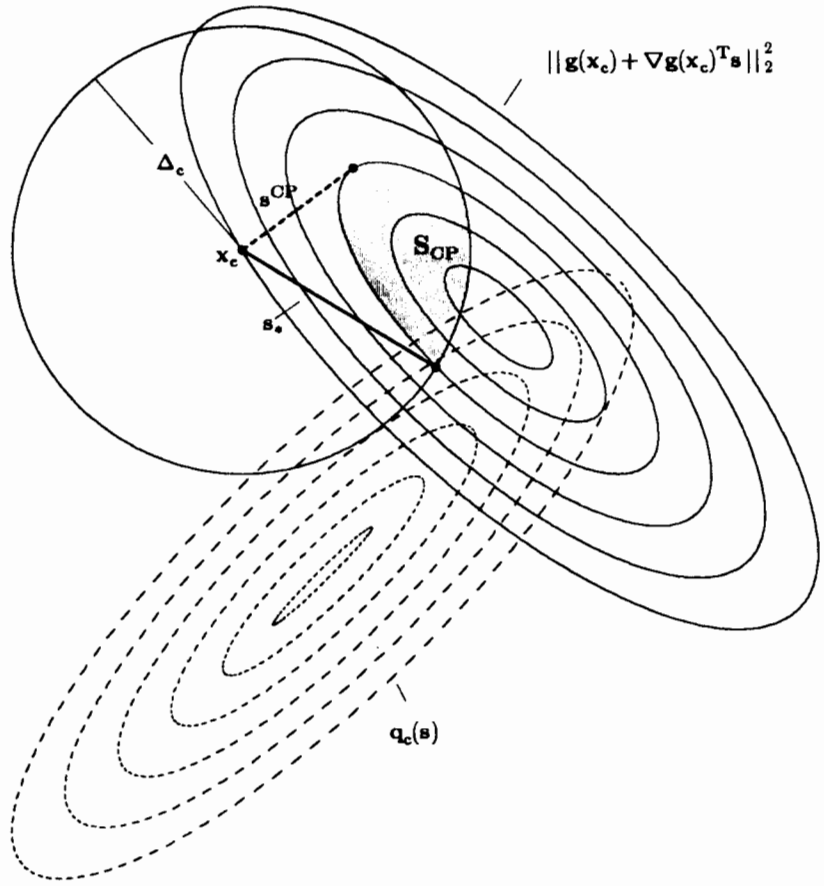

FIGURE 4.1 A Constrained Trust Region Step

In order to analyze our approach and study its relationship to other methods, we consider the problem

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & q(s) = a^T s + \tfrac{1}{2} s^T B s \\ \text{subject to} \quad & \| s \|_2 \leq \Delta \\ & \| g(x) + \nabla g(x)^T s \|_2 \leq \Theta, \end{aligned} \tag{4.1.6}$$

where $a \in \mathbb{R}^n$ and $B \in \mathbb{R}^{n \times n}$ is symmetric and nonsingular. We will refer to problem (4.1.6) as problem (QFQC). Problem (QFQC) is the basis of our trust region approach to equality constrained minimization. Its solution is given by the following lemma.

LEMMA 4.1.3 *If s_* solves problem (QFQC), then s_* is a solution to an equation of the form*

$$(B + \mu I + \eta \nabla g(x) \nabla g(x)^T) s_* = -(a + \nabla g(x) \eta g(x))$$

with $\mu, \eta \geq 0, \quad \| s_* \|_2 \leq \Delta, \quad \mu(\Delta - \| s_* \|_2) = 0,$

$\| g(x) + \nabla g(x)^T s_* \|_2 \leq \Theta$, *and* $\eta(\Theta - \| g(x) + \nabla g(x)^T s_* \|_2) = 0.$

Proof. Let us apply the method of Lagrange to the equivalent problem

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & q(s) = a^T s + \tfrac{1}{2} s^T B s \\ \text{subject to} \quad & \| s \|_2^2 \leq \Delta^2 \\ & \| g(x) + \nabla g(x)^T s \|_2^2 \leq \Theta^2 . \end{aligned} \tag{4.1.7}$$

By the first order necessary conditions of constrained optimization, s_* solves

problem (4.1.7), and therefore problem (QFQC), only if

$$\begin{aligned} a + B s_* + \mu s_* + \eta \nabla g(x) g(x) + \eta \nabla g(x) \nabla g(x)^T s_* &= 0 \\ \| s_* \|_2^2 &\leq \Delta^2 \\ \| g(x) + \nabla g(x)^T s_* \|_2^2 &\leq \Theta^2 \\ \mu(\Delta^2 - \| s_* \|_2^2) &= 0 \\ \eta(\Theta^2 - \| g(x) + \nabla g(x)^T s_* \|_2^2) &= 0, \end{aligned}$$

where μ and η are the Lagrange multipliers associated with the constraints $\| s \|_2^2 \leq \Delta^2$ and $\| g(x) + \nabla g(x)^T s \|_2^2 \leq \Theta^2$ respectively. The proof of the lemma then follows from these conditions. □

By defining a and B in various ways we can show how the solution to problem (QFQC) is related to existing theory. The following theorem shows that if the quadratic model $q(s)$ is the Taylor expansion of f, and the trust region constraint is not binding, then our step is a structured Newton step on the standard penalty function (2.1.1) with penalty constant η. It is important to note that η is not a free parameter, but is determined by the solution to problem (QFQC). The adjective structure is used in the same sense as in Section 2.3. In this case we don't use the complete Hessian matrix, but only the part that does not vanish at the solution.

THEOREM 4.1.4 *Let* $a = \nabla f(x)$ *and* $B = \nabla^2 f(x)$. *If* $\nabla^2 f(x)$ *is nonsingular and* Δ *is such that the constraint* $|| s ||_2 \leq \Delta$ *is not binding, then the solution* s_* *of problem (QFQC) is a structured Newton step for the standard penalty function*

$$P(x) = f(x) + \frac{1}{2}\eta g(x)^T g(x) .$$

Moreover, if $\nabla^2 f(x)$ *is positive definite,* s_* *is a descent direction for* $P(x)$, *independent of whether the constraint* $|| s ||_2 \leq \Delta$ *is binding.*

Proof. If $\nabla^2 f(x)$ is nonsingular and the constraint $|| s ||_2 \leq \Delta$ is not binding, the solution of problem (QFQC) with $a = \nabla f(x)$ and $B = \nabla^2 f(x)$ is given by

$$s_* = -(\nabla^2 f(x) + \eta \nabla g(x) \nabla g(x)^T)^{-1}(\nabla f(x) + \nabla g(x)\eta g(x)).$$

The Newton step s_N for minimizing $P(x)$ is

$$\nabla^2_{xx} P(x) s_N = -\nabla_x P(x),$$

where

$$\nabla P(x) = \nabla f(x) + \nabla g(x)\eta g(x)$$

and

$$\nabla^2 P(x) = \nabla^2 f(x) + \eta(\nabla g(x)^T \nabla g(x) + \nabla^2 g(x)\, g(x)).$$

At the solution x_*, $\nabla^2 g(x_*)\, g(x_*) = 0$, therefore a structured Newton step $s_{\hat{N}}$

for minimizing $P(x)$ is given by

$$(\nabla^2 f(x) + \eta \nabla g(x) \nabla g(x)^T) s_{\hat{N}} = -(\nabla f(x) + \nabla g(x) \eta \nabla g(x)).$$

Since $\nabla^2 f(x)$ is nonsingular, $s_{\hat{N}} = s_*$.

To prove that s_* is a descent direction for P, it is sufficient to show that $\nabla P(x)^T s_* < 0$. Noting that μ and η are nonnegative, and $\nabla^2 f(x)$ is positive definite, we have

$$\nabla P(x)^T s_* = -\nabla P(x)^T (\nabla^2 f(x) + \mu I + \eta \nabla g(x) \nabla g(x)^T)^{-1} \nabla P(x) < 0.$$

This completes the proof of the theorem. □

Now we show that if $q(s)$ is the Taylor expansion of the Lagrangian function, then the step that solves problem (QFQC) is a structured Newton step on the augmented Lagrangian function

$$L(x,\lambda,\eta) = f(x) + \lambda^T g(x) + \frac{1}{2}\eta\, g(x)^T g(x).$$

Again, we note that the penalty constant is determined by the solution to problem (QFQC).

THEOREM 4.1.5 *Let* $a = \nabla_x l(x,\lambda)$ *and* $B = \nabla^2_{xx} l(x,\lambda)$. *If* $\nabla^2_{xx} l(x,\lambda)$ *is nonsingular and* Δ *is such that the constraint* $|| s ||_2 \leq \Delta$ *is not binding, then the solution* s_* *of problem (QFQC) is a structured Newton step for the augmented Lagrangian. Moreover, if* $\nabla^2_{xx} l(x,\lambda)$ *is positive definite, then* s_* *is a descent direction for* $L(x,\lambda,\eta)$, *independent of whether the constraint* $|| s ||_2 \leq \Delta$ is binding.

Proof. The proof is analogous to the proof of Theorem (4.1.4). The Newton step s_N for minimizing $L(x,\lambda,\eta)$ is

$$\nabla^2_{xx} L(x,\lambda,\eta)\, s_N \;=\; -\nabla_x L(x,\lambda,\eta),$$

where

$$\nabla_x L(x,\lambda,\eta) = \nabla_x l(x,\lambda) + \nabla g(x)\eta g(x)$$

and

$$\nabla^2_{xx} L(x,\lambda,\eta) \;=\; \nabla^2_{xx} l(x,\lambda) + \eta(\nabla g(x)\nabla g(x)^T + \nabla^2 g(x) g(x)).$$

At the solution x_*, $g(x_*)=0$, so a structured Newton step for minimizing $L(x,\lambda,\eta)$ is

$$(\nabla^2_{xx} l(x,\lambda) + \eta\nabla g(x)\nabla g(x)^T)\, s_{\hat{N}} \;=\; -(\nabla_x l(x,\lambda) + \nabla g(x)\eta\nabla g(x)).$$

The Hessian $\nabla^2_{xx} l(x,\lambda)$ is nonsingular, therefore $s_{\hat{N}} = s_*$.

We now show that s_* is a descent direction for L. Noting that μ and η are nonnegative, and $\nabla^2_{xx} l(x,\lambda)$ is positive definite, we have

$$\begin{aligned}\nabla_x L(x,\lambda,\eta)^T s_* \;&=\; -\nabla_x L(x,\lambda,\eta)^T(\nabla_x l(x,\lambda) + \mu I + \eta\nabla g(x)\nabla g(x)^T)^{-1}\nabla_x L(x,\lambda,\eta)\\ &<\; 0\,.\end{aligned}$$

This completes our proof. □

We have shown how our approach relates to the standard penalty function and the augmented Lagrangian. It is also possible to relate the solution to problem (QFQC) to s_{QP}, the solution of problem (QP). From Section 2.3 we have that

$$s_{QP} \;=\; -B^{-1}\,(\nabla f(x) + \nabla g(x)\lambda)\,,$$

where

$$\lambda \;=\; (\nabla g(x)^T B^{-1}\nabla g(x))^{-1}\,(g(x) - \nabla g(x)^T B^{-1}\nabla f(x))\,.$$

The following theorem shows that one should not expect the solutions of problems (QFQC) and (QP) to be the same. It is reasonable to compare solutions of the two problems only in the case that the trust region constraint in problem (QFQC) is not binding. This is geometrically obvious in Figure 4.1.

THEOREM 4.1.6 *Let* $a = \nabla_x l(x,\lambda)$ *and* Δ *be such that the constraint* $|| s ||_2 \leq \Delta$ *is not binding. Then the solution of problem (QFQC) is the solution to problem (QP) if and only if the unconstrained minimizer of the quadratic,* $q(s)$, *satisfies linearized constraints, or the Cauchy point* s^{CP} *satisfies linearized constraints.*

Proof. Let s_* be a solution of problem (QFQC) First we will assume that $s_{QP} = s_*$ and show that either the unconstrained minimizer of $q(s)$ or the Cauchy step s^{CP} satisfies linearized constraints.

Since s_{QP} solves problem (QP), it satisfies linearized constraints, i.e.,

$$g(x) + \nabla g(x)^T s_{QP} = 0 .$$

Given that $s_* = s_{QP}$, we have

$$g(x) + \nabla g(x)^T s_* = 0 .$$

If the Cauchy step satisfies linearized constraints, then $\Theta = 0$, and problems (QP) and (QFQC) are equivalent.

Now let us consider $\Theta > 0$. Since $\Theta > 0$ the constraint

$$|| g(x) + \nabla g(x)^T s ||_2 \leq \Theta ,$$

is not binding and the multiplier η associated with this constraint is 0. Since $|| s ||_2 \leq \Delta$, is assumed not to be a binding constraint, the solution to problems (QP) and (QFQC) is given by

$$B s_* = -\nabla_x l(x,\lambda) ,$$

which gives the unconstrained minimizer of $q(s)$.

Next we must show that if the unconstrained minimizer of $q(s)$ satisfies linearized constraints, then $s_{QP} = s_*$. The result follows from the fact that in this case both problems become the same unconstrained minimization problem.□

As we progress through the iterations of our algorithm, we should expect to have Δ become large and $\Theta \to 0$. Clearly, for Δ sufficiently large, we will have $\mu = 0$. Also, from Theorem 4.1.5, we are led to conjecture that $\eta \to \infty$ as $\Theta \to 0$. Hence, we are interested in the behavior of the solution of problem (QFQC) as $\eta \to \infty$ and $\mu \to 0$. The following theorem gives us this behavior, which can be viewed as a form of consistency. Namely, while the solution of problem (QP) and problem (QFQC) are in general never the same; as $\eta \to \infty$ and $\mu \to 0$ the solution of problem (QFQC) approaches the solution of problem (QP). Thus we should expect our algorithm to eventually generate steps which are arbitrarily close to the SQP step. In practice we have found this to be the case. These comments are the subject of the following theorem.

THEOREM 4.1.7 *Let $a = \nabla_x l(x,\lambda)$, and B be positive definite. If $s(\mu,\eta)$ denotes the solution to problem (QFQC), then*

$$\lim_{(\mu,\eta)\to(0,\infty)} s(\mu,\eta) = s_{QP}.$$

Proof. To prove this theorem we obtain $(B+\mu I+\eta\nabla g(x)\nabla g(x)^T)^{-1}$. By the Sherman-Morrison-Woodbury formula (see page 50 of Ortega and Rheinboldt [1970]) we have

$$\begin{aligned}(B+\mu I+\eta\nabla g(x)\nabla g(x)^T)^{-1} &= (B+\mu I)^{-1} - \eta(B+\mu I)^{-1}\nabla g(x) \\ &\quad [I+\eta\nabla g(x)^T(B+\mu I)^{-1}\nabla g(x)]^{-1}\nabla g(x)^T(B+\mu I)^{-1}.\end{aligned}$$

Therefore,

$$
\begin{aligned}
s(\mu,\eta) &= -((B+\mu I)+\eta\nabla g(x)\nabla g(x)^T)^{-1}(\nabla_x l(x,\lambda)+\nabla g(x)\eta g(x)) \\
&= -(B+\mu I)^{-1}\{I-\nabla g(x)[\tfrac{1}{\eta}I+\nabla g(x)^T(B+\mu I)^{-1}\nabla g(x)]^{-1}\nabla g(x)^T(B+\mu I)^{-1}\} \\
&\quad (\nabla_x l(x,\lambda) + \nabla g(x)\eta g(x)) \\
&= -(B+\mu I)^{-1}\{\nabla_x l(x,\lambda)+\nabla g(x)[\tfrac{1}{\eta}I+\nabla g(x)^T(B+\mu I)^{-1}\nabla g(x)]^{-1} \\
&\quad (g(x)-\nabla g(x)^T(B+\mu I)^{-1}\nabla_x l(x,\lambda))\ \}
\end{aligned}
$$

Taking limits as $\eta\to\infty$ and $\mu\to 0$ we have

$$
\begin{aligned}
\lim_{(\mu,\eta)\to(0,\infty)} s(\mu,\eta) \;=\; & -B^{-1}(\nabla_x l(x,\lambda)+\nabla g(x)[\nabla g(x)^T B^{-1}\nabla g(x)]^{-1} \\
& (\ g(x)-\nabla g(x)^T B^{-1}\nabla_x l(x,\lambda))\ )\ .
\end{aligned}
$$

It is straightforward to see that by substituting $\nabla f(x)+\nabla g(x)\lambda$ for $\nabla_x l(x,\lambda)$ we obtain s_{QP}. □

4.2. A Merit Function.

In section (4.1), we established the relationship between the step s_* and some of the different penalty functions of chapter 2. We also proved that a penalty constant is given to us by the QFQC problem itself. It is natural then to use this relationship to determine whether or not the step calculated gives a new iterate that is an improvement over the old iterate.

Recall Theorems (4.1.4) and (4.1.5). If we minimize a quadratic approximation of the objective function f in problem (QFQC), the solution s_* is a structured Newton step for the standard penalty function (2.1.1) with penalty

constant $C=\eta$. If instead, we minimize a quadratic approximation to the Lagrangian function, the solution s_* is a structured Newton step for the augmented Lagrangian with penalty constant $C=\eta$. Moreover, if the Hessians in the quadratic approximations are positive definite, then the solutions define descent directions for the penalty function and the augmented Lagrangian respectively.

Furthermore, we observe that when $\lambda=0$, $L(x,\lambda,\eta)$ reduces to $P(x,\eta)$. Therefore a natural choice for the merit function is the augmented Lagrangian

$$
L(x,\lambda,\eta) \;=\; f(x) + \lambda^T g(x) + \tfrac{1}{2}\eta g(x)^T g(x)\,, \qquad (4.2.1)
$$

where $\eta \geq 0$ is the penalty constant and is given by the solution s_* of problem (QFQC). The multiplier λ will be chosen accordingly with the quadratic function we choose to minimize in problem (QFQC).

To decide whether to accept a step or not, and in updating the radius of the trust region, we make use of the augmented Lagrangian in the same way that the objective function f is used in the unconstrained case. A step is accepted if it gives enough descent on the augmented Lagrangian. To update the radius of the trust region, we look at the agreement between a model of the augmented Lagrangian and the augmented Lagrangian.

However, there is a special case we should pause to consider. Suppose that the Cauchy step s^{CP} for problem (4.1.4) satisfies linearized constraints. Notice that this case includes the case when x_c is feasible, but not optimal, because

then $s^{CP} = 0$ and $g(x_c) = 0$. Problem (QFQC) is then equivalent to

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & q_c(s) \\ \text{subject to} \quad & \| g(x_c) + \nabla g(x_c)^T s \|_2^2 = 0 \\ & \| s \|_2 \leq \Delta_c \,. \end{aligned} \tag{4.2.2}$$

Let s_* be the solution of (4.2.2). This implies that s_* satisfies linearized constraints, i.e. $g(x_c)+\nabla g(x_c)^T s_* = 0$. Furthermore, the gradient of the constraint $\| g(x_c)+\nabla g(x_c)^T s \|_2^2 = 0$ at the solution s_* is zero. This fact leads to numerical difficulties. To overcome these difficulties, we consider solving an equivalent problem

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & q_c(s) \\ \text{subject to} \quad & g(x_c) + \nabla g(x_c)^T s = 0 \\ & \| s \|_2 \leq \Delta_c \,. \end{aligned} \tag{4.2.3}$$

A solution of problem (4.2.3) will be a solution of problem (4.2.2), but there will not be a value for the penalty constant η to be used in the merit function. We propose the following scheme for determining a value for η.

Consider the problem of minimizing the augmented Lagrangian function $L(x,\lambda,\eta)$, with respect to the variable x and for some fixed value of λ and η, using a trust region approach. At each iteration the step is calculated as a solution to the problem

$$\begin{aligned} \underset{s}{\text{minimize}} \quad & \nabla_x L(x_c,\lambda,\eta)^T s + \tfrac{1}{2} s^T H_c s \\ \text{subject to} \quad & \| s \|_2 \leq \Delta_c \end{aligned} \tag{4.2.4}$$

where $H_c \equiv B_c + \eta \nabla g(x) \nabla g(x)^T$ is the approximation to the Hessian of the

augmented Lagrangian and B_c is the same as in problem (QFQC).

As we mentioned in Section 3.3, to obtain convergence the step should give as much descent as the Cauchy step. For now, let us assume that Δ_c in problem (4.2.4) is large enough so that the Cauchy point is inside the trust region. Let $\bar{s}^{CP}$ be the Cauchy step for problem (4.2.4), i.e.,

$$\bar{s}^{CP} = -\frac{\nabla_x L(x_c,\lambda,\eta)^T \nabla_x L(x_c,\lambda,\eta)}{\nabla_x L(x_c,\lambda,\eta)^T H_c \nabla_x L(x_c,\lambda,\eta)} \nabla_x L(x_c,\lambda,\eta) \,. \tag{4.2.5}$$

The prediction for the value of the model objective function at the Cauchy point is then given by

$$\nabla_x L(x_c,\lambda,\eta)^T \bar{s}^{CP} + \tfrac{1}{2} \bar{s}^{CP^T} H_c \bar{s}^{CP} = -\frac{1}{2} \frac{\| \nabla_x L(x_c,\lambda,\eta) \|_2^4}{\nabla_x L(x_c,\lambda,\eta)^T H_c \nabla L(x_c,\lambda,\eta)} \tag{4.2.6}$$

To obtain convergence, we require that the step s that solves problem (4.2.3) give as much descent as the Cauchy step $\bar{s}^{CP}$ for problem (4.2.4), i.e.,

$$-\frac{1}{2} \frac{\| \nabla_x L(x_c,\lambda,\eta) \|_2^4}{\nabla_x L(x_c,\lambda,\eta)^T H_c \nabla_x L(x_c,\lambda,\eta)} \geq \nabla_x L(x_c,\lambda,\eta)^T s + \tfrac{1}{2} s^T H_c s. \tag{4.2.7}$$

Therefore we will choose $\eta > 0$ for which (4.2.7) holds. If there does not exist a positive value of η for which (4.2.7) is satisfied, then the step s is not accepted, and the radius of the trust region is reduced.

The value of η for which (4.2.7) holds does not necessarily exist and is not in general unique. We can expect to have real intervals for which any value in the interval satisfies (4.2.7). The question now is which value to choose for η. We consider an optimistic strategy. If there is a value for η that satisfies (4.2.7)

and allows a very good prediction of L such that the trust region can be increased, we choose this value for η. If this value does not exist, we find a value for η for which the trust region will remain the same. If this value of η does not satisfy (4.2.7), we choose a value for η which will permit us to accept the step. It is possible that the values of η for which (4.2.7) holds do not allow us to accept the step. In this case, we reject the step, reduce the trust region and calculate a shorter step using our current model.

Now assume that the radius Δ_c in problem (4.2.4) is so small that the Cauchy point lies outside the trust region. In this case the Cauchy step $\bar{s}^{CP}$ is given by

$$\bar{s}^{CP} = -\frac{\Delta_c \nabla_z L(x_c,\lambda,\eta)}{\|\nabla_z L(x_c,\lambda,\eta)\|_2}, \tag{4.2.8}$$

and the Cauchy prediction is

$$-\Delta_c \nabla_z L(x_c,\lambda,\eta) + \tfrac{1}{2}\Delta_c \frac{\nabla_z L(x_c,\lambda,\eta)^T H_c \nabla_z L(x_c,\lambda,\eta)}{\|\nabla_z L(x_c,\lambda,\eta)\|_2^2}. \tag{4.2.9}$$

In this case, the Cauchy step $\bar{s}^{CP}$ is not the step that minimizes the quadratic model in the steepest descent direction, it is the step along the steepest descent direction that has length Δ_c. Therefore the value of the quadratic function in (4.2.4) which we obtain with (4.2.8) is greater or equal to the value we obtain

with (4.2.5). Hence

$$-\Delta_c \nabla_z L(x_c,\lambda,\eta) + \frac{1}{2}\frac{\nabla_z L(x_c,\lambda,\eta)^T H_c \nabla_z L(x_c,\lambda,\eta)}{\|\nabla_z L(x_c,\lambda,\eta)\|_2^2} \tag{4.2.10}$$

$$\geq -\frac{1}{2}\frac{\|\nabla_z L(x_c,\lambda,\eta)\|_2^4}{\nabla_z L(x_c,\lambda,\eta)^T H_c \nabla_z L(x_c,\lambda,\eta)}.$$

Combining (4.2.7) and (4.2.10), we have

$$-\Delta_c \nabla_z L(x_c,\lambda,\eta) + \frac{1}{2}\frac{\nabla_z L(x_c,\lambda,\eta)^T H_c \nabla_z L(x_c,\lambda,\eta)}{\|\nabla_z L(x_c,\lambda,\eta)\|_2^2} \tag{4.2.11}$$

$$\geq L(x_c,\lambda,\eta)^T s + \tfrac{1}{2} s^T H_c s.$$

To determine η we then proceed in the same way as before using (4.2.11) instead of (4.2.7).

Our numerical results indicate that this idea gives good results, although at the writing of this thesis we lack theoretical justification.

CHAPTER 5

A Constrained Trust Region Algorithm

The ideas of Chapter 4 are put together in this chapter to construct a trust region algorithm to find a local solution of the nonlinear equality constrained optimization problem. In Section 5.1 we discuss a way of approximating a solution to the constrained trust region problem (QFQC) using an optimal step strategy. In addition we suggest a dogleg approach and define two different dogleg curves. In Section 5.2 we present the scheme for accepting a step and updating the trust region. Section 5.3 contains the statement of the complete trust region algorithm. Numerical results are presented in Section 5.4.

5.1. Computing a Constrained Trust Region Step.

The first approach we will follow for approximating a solution s_* to problem (QFQC) is based on the idea for computing a locally constrained optimal step in the unconstrained case described in Section 3.1.

Recall problem (QFQC)

$$\begin{aligned} &\textit{minimize} \quad a^T s + \tfrac{1}{2} s^T B s \\ &\textit{subject to} \quad \| s \|_2 \leq \Delta \\ &\qquad\qquad \| g(x) + \nabla g(x)^T s \|_2 \leq \Theta \,. \end{aligned}$$

If $\| s^{QP} \| \leq \Delta$, then we take $s = s^{QP}$. Now assume $\| s^{QP} \| > \Delta$. First we must determine which constraints of problem (QFQC) are binding. This is done by considering the three different possibilities: only the Δ trust region is binding, only the Θ feasibility constraint is binding, both the constraints are binding.

To determine if only the constraint $\| s \| \leq \Delta$ is binding, we find the solution s_Δ to the unconstrained trust region problem,

$$\begin{aligned} &\textit{minimize} \quad a^T s + \tfrac{1}{2} s^T B s \\ &\textit{subject to} \quad \| s \|_2 \leq \Delta \,, \end{aligned} \tag{5.1.1}$$

using the techniques already available for approximating s_Δ, see Gay [1981], and Moré and Sorensen [1983]. If the solution s_Δ is such that

$$\| g(x) + \nabla g(x)^T s_\Delta \|_2 \;\leq\; \Theta \,, \tag{5.1.2}$$

then we know that the feasibility constraint is not binding at the solution, and s_Δ is the solution to problem (QFQC).

Now let us assume that inequality (5.1.2) does not hold, i.e., the constraint

$\| g(x)+\nabla g(x)^T s \|_2 \leq \Theta$ is binding at the solution. To determine if only this constraint is binding, we consider the problem

$$\begin{aligned} &\underset{s}{\text{minimize}} \quad a^T s + \tfrac{1}{2} s^T B s \\ &\text{subject to} \quad \| g(x) + \nabla g(x)^T s \|_2 \leq \Theta . \end{aligned} \tag{5.1.3}$$

To calculate an approximate solution s_Θ of problem (5.1.3), we proceed as in the unconstrained case and construct an algorithm to find an approximate solution η to the scalar equation

$$\phi(\eta) \equiv \| g(x) + \nabla g(x)^T s(\eta) \|_2 - \Theta = 0,$$

where $s(\eta) = -(B+\eta\nabla g(x)\nabla g(x)^T)^{-1}(a+\eta\nabla g(x)g(x))$. The local model suggested by the problem is

$$m(\eta) = \frac{\hat{\alpha}+\hat{\beta}\eta}{\hat{\gamma}+\hat{\epsilon}\eta} - \Theta ,$$

which can be simplified by dividing numerator and denominator by $\hat{\gamma}$ to obtain

$$m(\eta) = \frac{\alpha+\beta\eta}{1+\epsilon\eta} - \Theta ,$$

where $\alpha = \frac{\hat{\alpha}}{\hat{\gamma}}$, $\beta = \frac{\hat{\beta}}{\hat{\gamma}}$, and $\epsilon = \frac{\hat{\epsilon}}{\hat{\gamma}}$. The model $m(\eta)$ has three free parameters, α, β, and γ, and we choose them to satisfy the three interpolatory conditions

$$\begin{aligned} m(\eta) &= \frac{\alpha+\beta\eta}{1+\epsilon\eta} - \Theta &&= \phi(\eta) \\ m'(\eta) &= \frac{\beta-\alpha\epsilon}{(1+\epsilon\eta)^2} &&= \phi'(\eta) \\ m''(\eta) &= -\frac{2\epsilon}{1+\eta\epsilon} m'(\eta) &&= \phi''(\eta) , \end{aligned}$$

where

$$\begin{aligned} \phi(\eta) &= \| g(x) + \nabla g(x)^T s(\eta) \|_2 - \Theta , \\ \phi'(\eta) &= \frac{s'(\eta)^T \nabla g(x)(\nabla g(x)^T s(\eta)+g(x))}{\| g(x) + \nabla g(x)^T s(\eta) \|_2} \\ \phi''(\eta) &= \frac{s''(\eta)^T \nabla g(x)(g(x)+\nabla g(x)^T s(\eta)) + s'(\eta)\nabla g(x)\nabla g(x)^T s'(\eta) - \phi'(\eta)\phi'(\eta)}{\| g(x) + \nabla g(x)^T s(\eta) \|_2} , \end{aligned}$$

and

$$\begin{aligned} s'(\eta) &= -(B + \eta\nabla g(x)\nabla g(x)^T)^{-1}\nabla g(x)(\nabla g(x)^T s(\eta) + g(x)) \\ s''(\eta) &= -2(B + \eta\nabla g(x)\nabla g(x)^T)^{-1}\nabla g(x)\nabla g(x)^T s'(\eta) . \end{aligned}$$

Notice that $m''(\eta)$ completely determines ϵ, so that

$$\epsilon = -\frac{\phi''(\eta)}{\eta\phi''(\eta) + 2\phi'(\eta)} . \tag{5.1.4}$$

Then $m(\eta)$ and $m'(\eta)$ give

$$\alpha = \phi(\eta) + \Theta - \phi'(\eta)(1+\epsilon\eta)\eta \tag{5.1.5}$$

and

$$\beta = \phi'(\eta)(1+\epsilon\eta) + (\phi(\eta)+\Theta)\eta . \tag{5.1.6}$$

We choose η, to satisfy $m(\eta)=0$, i.e.,

$$\eta = \frac{\Theta-\alpha}{\beta-\Theta\epsilon} .$$

Substituting (5.1.4), (5.1.5) and (5.1.6) into the above gives the following

iteration for calculating η

$$\eta_+ = \eta - \frac{2\phi(\eta)\phi'(\eta)}{2\phi'(\eta)^2 - \phi(\eta)\phi''(\eta)} . \tag{5.1.7}$$

As in the unconstrained case, to transform (5.1.7) into a useful computational algorithm, the iteration must be properly safeguarded. The implementation we have at this time safeguards the iteration in a naive and ad hoc way. Our present research is concerned with this issue, and we hope to be able to come up with a more sophisticated safeguarding scheme.

So far, we have calculated a solution s_Δ of problem (QFQC) without the constraint $\| g(x)+\nabla g(x)^T s \|_2 \leq \Theta$ (problem (5.1.1)). If s_Δ satisfies the Θ feasibility constraint, then $s_* = s_\Delta$. If s_Δ does not satisfy the Θ constraint, then we know that it is binding at the solution s_*, and proceed to calculate a solution s_Θ of problem (QFQC) without the constraint $\| s \|_2 \leq \Delta$ (problem (5.1.3)). If $\| s_\Theta \|_2 \leq \Delta$, then $s_* = s_\Delta$.

Now let us assume that $\| s_\Theta \| > \Delta$, and that $\| g(x)+\nabla g(x)^T s_\Delta \|_2 > \Theta$. We then know that both the constraints are binding at the solution of problem (QFQC). The basic idea remains the same, to construct an algorithm for finding approximate solutions μ and η to the scalar equations

$$\begin{aligned} \phi_1(\mu,\eta) &\equiv \| s(\mu,\eta) \|_2 - \Delta &&= 0 \\ \phi_2(\mu,\eta) &\equiv \| g(x) + \nabla g(x)^T s(\mu,\eta) \|_2 - \Theta &&= 0 , \end{aligned} \tag{5.1.8}$$

where $s(\mu,\eta) = -(B+\mu I+\eta\nabla g(x)\nabla g(x)^T)^{-1}(a+\eta\nabla g(x)g(x))$. The two local

models we consider are

$$\begin{aligned} m_1(\mu,\eta) &= \frac{\alpha+\beta\eta}{\gamma+\mu+\epsilon\eta} - \Delta \\ m_2(\mu,\eta) &= \frac{\bar{\alpha}+\bar{\beta}\eta}{\gamma+\mu+\epsilon\eta} + \xi - \Theta \end{aligned} \tag{5.1.9}$$

The model $m_1(\mu,\eta)$ in (5.1.9) can be somewhat simplified by using the fact that $\lim_{(\mu,\eta)\to(0,\infty)} s(\mu,\eta) = s^{QP}$, see Theorem (4.1.7). We apply this result to obtain

$$\lim_{(\mu,\eta)\to(0,\infty)} m_1(\mu,\eta) = \frac{\beta}{\epsilon} = \| s^{QP} \|_2 ,$$

where $\| s^{QP} \|_2$ is known, since we started by solving for s^{QP} to see if it lay inside the trust region. Thus we can rewrite $m_1(\mu,\eta)$ as

$$m_1(\mu,\eta) = \frac{\alpha+\epsilon\frac{\beta}{\epsilon}\eta}{\gamma+\mu+\epsilon\eta} - \Delta = \frac{\alpha+\epsilon l\eta}{\gamma+\mu+\epsilon\eta} - \Delta ,$$

where $l = \| s^{QP} \|_2$. Our model is then given by

$$\begin{aligned} m_1(\mu,\eta) &= \frac{\alpha+\epsilon l\eta}{\gamma+\mu+\epsilon\eta} - \Delta \\ m_2(\mu,\eta) &= \frac{\bar{\alpha}+\bar{\beta}\eta}{\gamma+\mu+\epsilon\eta} + \xi - \Theta \end{aligned} \tag{5.1.10}$$

The model (5.1.10) has six free parameters α, γ, ϵ, $\bar{\alpha}$, $\bar{\beta}$, and ξ, which we choose to satisfy the conditions

$$m_1(\mu,\eta) = \frac{\alpha+\epsilon l\eta}{\gamma+\mu+\epsilon\eta} - \Delta = \phi_1(\mu,\eta)$$

$$
\begin{aligned}
m_2(\mu,\eta) &= \frac{\overline{\alpha}+\overline{\beta}\eta}{\gamma+\mu+\epsilon\eta} + \xi - \Theta = \phi_2(\mu,\eta) \\
\frac{\partial}{\partial_\mu} m_1(\mu,\eta) &= -\frac{\alpha+l\epsilon\eta}{(\gamma+\mu+\epsilon\eta)^2} = \frac{\partial}{\partial_\mu}\phi_1(\mu,\eta) \\
\frac{\partial}{\partial_\eta} m_1(\mu,\eta) &= \frac{(\gamma+\mu)l\epsilon-\alpha\epsilon}{(\gamma+\mu+\epsilon\eta)^2} = \frac{\partial}{\partial_\eta}\phi_1(\mu,\eta) \\
\frac{\partial}{\partial_\mu} m_2(\mu,\eta) &= -\frac{\overline{\alpha}+\overline{\beta}\eta}{(\gamma+\mu+\epsilon\eta)^2} = \frac{\partial}{\partial_\mu}\phi_2(\mu,\eta) \\
\frac{\partial}{\partial_\eta} m_2(\mu,\eta) &= \frac{(\gamma+\mu)\overline{\beta}-\overline{\alpha}\epsilon}{(\gamma+\mu+\epsilon\eta)^2} = \frac{\partial}{\partial_\eta}\phi_2(\mu,\eta)
\end{aligned}
$$

where

$$
\begin{aligned}
\phi_1(\mu,\eta) &= \| s(\mu,\eta) \|_2 - \Delta \\
\phi_2(\mu,\eta) &= \| g(x) + \nabla g(x)^T s(\mu,\eta) \|_2 - \Theta \\
\frac{\partial}{\partial_\mu}\phi_1(\mu,\eta) &= \frac{\left(\frac{\partial}{\partial_\mu} s(\mu,\eta)\right)^T s(\mu,\eta)}{\| s(\mu,\eta) \|_2} \\
\frac{\partial}{\partial_\eta}\phi_1(\mu,\eta) &= \frac{\left(\frac{\partial}{\partial_\eta} s(\mu,\eta)\right)^T s(\mu,\eta)}{\| s(\mu,\eta) \|_2} \\
\frac{\partial}{\partial_\mu}\phi_2(\mu,\eta) &= \frac{\left(\frac{\partial}{\partial_\mu} s(\mu,\eta)\right)^T \nabla g(x)\nabla g(x)^T s(\mu,\eta) + \left(\frac{\partial}{\partial_\mu} s(\mu,\eta)\right)^T \nabla g(x) g(x)}{\| g(x) + \nabla g(x)^T s(\mu,\eta) \|_2} \\
\frac{\partial}{\partial_\eta}\phi_2(\mu,\eta) &= \frac{\left(\frac{\partial}{\partial_\eta} s(\mu,\eta)\right)^T \nabla g(x)\nabla g(x)^T s(\mu,\eta) + \left(\frac{\partial}{\partial_\eta} s(\mu,\eta)\right)^T \nabla g(x) g(x)}{\| g(x) + \nabla g(x)^T s(\mu,\eta) \|_2}
\end{aligned}
$$

We choose μ and η to satisfy $m_1(\mu,\eta)=0=m_2(\mu,\eta)$, i.e., we solve the 2×2 linear system

$$
\begin{bmatrix} \Delta & \Delta\epsilon - l\epsilon \\ \Theta-\xi & \Theta\epsilon-\overline{\beta}-\xi\epsilon \end{bmatrix} \begin{pmatrix} \mu \\ \eta \end{pmatrix} = \begin{bmatrix} \Delta\gamma-\alpha \\ \overline{\alpha}+\xi\gamma-\Theta\gamma \end{bmatrix}. \tag{5.1.11}
$$

The system (5.1.11) determines the iteration for solving (5.1.8).

To transform (5.1.11) into a useful computational algorithm, we must safeguard μ and η. As we mentioned before, we are not satisfied with our present strategy and this is one of the topics of our present research.

We have discussed the cases we consider in order to find an approximate solution to problem (QFQC). The following algorithm summarizes these cases and defines a typical iteration. Remember that we take the SQP step if it is in the trust region.

ALGORITHM 5.1.1. Find an Approximate Solution of Problem (QFQC).

1) Determine an approximate solution s_Δ to problem (5.1.1).

2) If $\| g(x)+\nabla g(x)^T s_\Delta \|_2 \leq \Theta$ then $s = s_\Delta$; exit.

3) Determine an approximate solution s_Θ to problem (5.1.3).

4) If $\| s_\Theta \|_2 \leq \Delta$ then $s = s_\Theta$; exit.

5) Determine an approximate solution s_* to problem (QFQC); $s = s_*$.

A second approach for approximating a solution s_* of problem (QFQC) is based on the dogleg methods suggested for the unconstrained problem. A

complete description of this approach can be found in Dennis and Schnabel [1983].

Before we describe our dogleg method, let us recall some of the steps and points we have defined. The Cauchy step for our model problem of $\| g(x) \|_2^2$ (problem (4.1.2)) is denoted by s^{CP}, and we will refer to the corresponding Cauchy point as CP. The solution of problem (QP) is denoted by s^{QP}, and we define $x^{QP} = x_c + s^{QP}$. We assume that we have already computed s^{QP} and that $\| s^{QP} \|_2 > \Delta_c$.

We have considered two dogleg curves. The first one is a piecewise linear function connecting the Cauchy point, CP, to the direction s^{QP}, as indicated in Figure 5.1. We now choose our next iterate x_+ to be the point along this polygonal arc such that $\| x_+ - x_c \|_2 = \Delta_c$. This particular dogleg is analogous to the dogleg introduced by Powell [1970a] for the unconstrained case. The Cauchy point that minimizes $\| g(x_c) + \nabla g(x_c)^T s \|_2$ within the trust region is used instead of the Cauchy point that minimizes the quadratic model of the objective function, and the SQP direction acts as the Newton direction.

The second dogleg curve we propose introduces a third direction. From the Cauchy point CP, we determine the step $s^{\hat{CP}}$ in the steepest descent direction that minimizes the quadratic model of f within the region s^{CP} (4.1.4). The step $s^{\hat{CP}}$ gives a second Cauchy point which we will denote by $\hat{CP}$, see Figure 5.2.

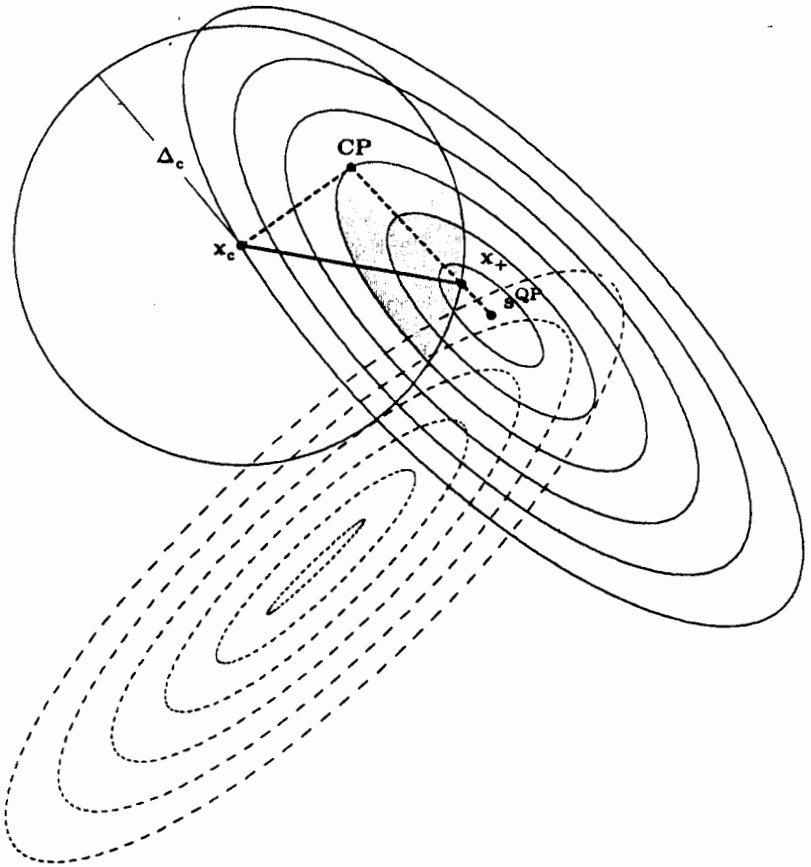

FIGURE 5.1 Dogleg Step I

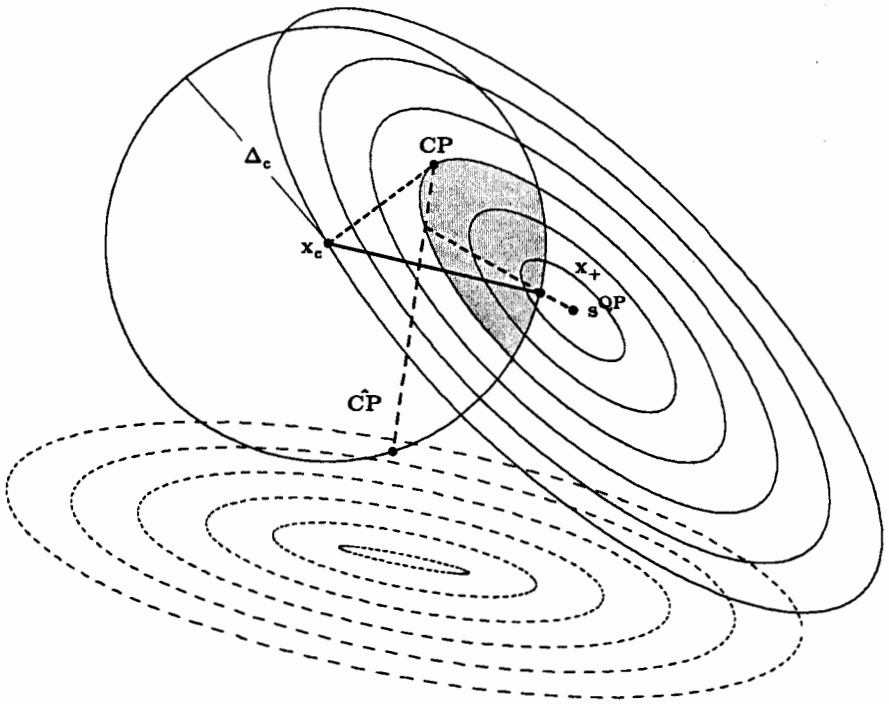

FIGURE 5.2 Dogleg Step II

We then define the dogleg curve as the piecewise linear function connecting x_c, CP, $\hat{CP}$, and the SQP point x^{QP} as indicated in Figure 5.2., and choose x_+ as the point on this polygonal arc that satisfies $\| x_+ - x_c \|_2 = \Delta_c$. Observe that this dogleg curve reduces to the dogleg curve for the unconstrained case when no constraints are present.

Certainly other definitions for a dogleg curve are possible. However, we will not discuss more possibilities until we learn more about these particular ones from our numerical results.

5.2. Accepting a Step and Updating the Trust Region

As we mentioned in Section 4.2, we will use the augmented Lagrangian in the same way as the objective function is used in the unconstrained case, to decide when to accept a step and how to update the trust region. However, since in some cases the choice of the penalty constant is not unique, we will incorporate the optimistic strategy described in Section 4.2.

The condition for accepting a step is the same one developed for the unconstrained case in Section 3.2. Let x_c be the current iterate, s_c the step calculated, $x_+ = x_c + s_c$, and Δ_c the current trust region radius. We accept x_+ as the next iterate if

$$L(x_+,\lambda_+,\eta_+) \leq L(x_c,\lambda_+,\eta_+) + \alpha \nabla_x L(x_c,\lambda_+,\eta_+)^T s_c \ , \qquad (5.2.1)$$

where λ_+ and η_+ are the multiplier and penalty constant associated with the step s_c, and α is a positive constant in $(0,\frac{1}{2})$. Our implementation uses

$\alpha = 10^{-4}$. If x_+ does not satisfy (5.2.1), we reduce the trust region by a factor between $\frac{1}{10}$ and $\frac{1}{2}$, and return to compute a shorter step.

Once we have a new iterate x_+ that satisfies (5.2.1), we decide whether or not to try a larger step from x_c. For this, we compare the actual reduction

$$\Delta L \equiv L(x_+,\lambda_+,\eta_+) - L(x_c,\lambda_+,\eta_+) , \tag{5.2.2}$$

to the predicted reduction

$$\Delta L_{pred} = \bar{q}_c(x_+ - x_c,\lambda_+,\eta_+) , \tag{5.2.3}$$

where $\bar{q}_c(x_+ - x_c,\lambda_+,\eta_+)$ is the model of the augmented Lagrangian defined by

$$\bar{q}_c(x_+ - x_c,\lambda_+,\eta_+) = \nabla_x L(x_c,\lambda_+,\eta_+)^T(x_+ - x_c) + \frac{1}{2}(x_+ - x_c)^T H_c(x_+ - x_c) ,$$

where $H_c = B_c + \eta \nabla g(x) \nabla g(x)^T$, and $B_c = \nabla^2_{xx} l(x_c,\lambda_c)$ or some approximation to it. If the agreement is so good that

$$| \Delta L_{pred} - \Delta L | \leq \beta | \Delta L | , \tag{5.2.4}$$

where we choose $\beta = 0.1$, or the actual reduction in L is very large, i.e.,

$$L(x_+,\lambda_+,\eta_+) \leq L(x_c,\lambda_+,\eta_+) + \nabla_x L(x_c,\lambda_+,\eta_+)^T s_c , \tag{5.2.5}$$

then we increase the radius of the trust region and compute a new x_+ using the current model. In both these cases we save s_c, so if the new s_c does not satisfy (5.2.1), we drop back to the last step accepted.

Now suppose that we have decided to accept s_c as our step, so x_+ is now our next iterate. Following the same procedure as in the unconstrained case, we

need to update Δ_c. Again the decision is based on whether the quadratic model of L is predicting the actual decrease in the merit function well. Let $0 < \sigma_1 < \sigma_2 < 1$ and $0 < \tau_1 < \tau_2 < 1 < \tau_3$ be specified constants. If

$$\Delta L \leq \sigma_2 \Delta L_{pred} , \tag{5.2.6}$$

then we believe that the model has predicted the merit function sufficiently well, so we increase the trust region. We will follow the principal of increasing the trust radius in terms of the previous trust radius, so that $\Delta_+ = \tau_3 \Delta_c$. Our implementation uses the values $\sigma_2 = 0.75$ and $\tau_3 = 2$. On the other hand, if

$$\Delta L > \sigma_1 \Delta L_{pred} , \tag{5.2.7}$$

then we believe that the model greatly overestimated the decrease in the merit function, therefore we decrease the trust region. The radius of the trust region is decreased in terms of the length of the step, hence $\Delta_+ \in [\tau_1 || s ||_2, \tau_2 || s ||_2]$. Our implementation uses $\tau_1 = \frac{1}{10}$ and $\tau_2 = \frac{1}{2}$. If neither inequality (5.2.6) or (5.2.7) holds, we leave the trust region the same, i.e., $\Delta_+ = \Delta_c$.

When the step s_c satisfies linearized constraints, it may be possible to have a choice of values for the penalty constant. Let N_c be the set of possible values, specifically

$$N_c = \left\{ \eta : -\frac{1}{2} \frac{|| \nabla_x L(x_c,\lambda_+,\eta_+) ||^4}{\nabla_x L(x_c,\lambda_+,\eta_+)^T H_c \nabla_x L(x_c,\lambda_+,\eta_+)} \geq \nabla_x L(x_c,\lambda_+,\eta_+)^T s_c + \frac{1}{2} s_c^T H_c s_c \right\} \tag{5.2.8}$$

or

$$N_c = \left\{\eta: \; -\Delta_c \nabla_x L(x_c,\lambda_+,\eta_+) + \frac{1}{2}\frac{||\nabla_x L(x_c,\lambda_+,\eta_+)||^4}{\nabla_x L(x_c,\lambda_+,\eta_+)^T H_c \nabla_x L(x_c,\lambda_+,\eta_+)} \right. \tag{5.2.9}$$
$$\left. \geq \nabla_x L(x_c,\lambda_+,\eta_+)^T s_c + \frac{1}{2} s_c^T H_c \, s_c \right\}.$$

If the set N_c is empty, we reject the step, decrease the trust region and return to calculating a shorter step from x_c.

Let us now assume that the set N_c is nonempty. Following our optimistic strategy, we find values for the penalty constant that will allow us to do internal doubling. Specifically, we determine the set

$$N_{dbl} = \left\{\eta: \mid \Delta L_{pred} - \Delta L \mid \leq \beta \mid \Delta L \mid \; or \; \Delta L \leq \sigma_2 \Delta L_{pred}\right\}. \tag{5.2.10}$$

If $N_c \bigcap N_{dbl} \neq \emptyset$, we choose $\eta_+ \in N_c \bigcap N_{dbl}$, and go on to double the trust region and compute a new x_+ using the current model.

If $N_c \bigcap N_{dbl} = \emptyset$, internal doubling is not done. Instead we find the values of η that will allow our model to predict the merit function sufficiently well. Let

$$N_{inc} = \left\{\eta: \Delta L < \sigma_2 \Delta L_{pred}\right\}. \tag{5.2.11}$$

As before, if $N_c \bigcap N_{inc} \neq \emptyset$, we choose $\eta_+ \in N_c \bigcap N_{inc}$, accept the step and increase the trust radius for the next iteration.

If the sets N_c and N_{inc} do not intersect, we determine if there is a value for η that will allow us to accept the step and leave the trust region the same. For

this we find the values of the penalty constant that are in the set

$$N_{sme} = \left\{\eta: \sigma_1 \Delta L_{pred} < \Delta L < \sigma_2 \Delta L_{pred}\right\}. \tag{5.2.12}$$

We choose $\eta_+ \in N_c \bigcap N_{sme}$, accept the step, and let the next trust region radius be the same as the current one. If such η does not exist, we will choose an η for which the step is accepted. Therefore, we determine the set

$$N_{dec} = \left\{\eta: L(x_+,\lambda_+,\eta_+) < L(x_c,\lambda_+,\eta_+) + \alpha \nabla_x L(x_c,\lambda_+,\eta_+)^T s_c\right\}, \tag{5.2.13}$$

and choose $\eta_+ \in N_c \bigcap N_{dec}$. If the two sets N_c and N_{dec} do not intersect, we reject the step, decrease the trust region and return to compute a shorter step.

To complete our algorithm for accepting a step and updating the trust region, we need to specify the choice for the multiplier λ_+ to be used in the merit function. If the step s_c calculated by our method satisfies linearized constraints, then the step s_c is the solution s^{QP} to problem (QP) or probably is very close to it. For s^{QP} we have an associated multiplier λ^{QP}. We choose $\lambda_+ = \lambda^{QP}$, because this is the only multiplier given by the update formulas of Section 2.2 that satisfies linearized constraints, see Tapia [1977], [1978]. If the step s_c does not satisfy linearized constraints, then $s_c = s_*$ is very likely not close to s^{QP}. In this case we will use $\lambda_+ = 0$, and the merit function becomes the standard penalty function (2.1.1) with $C = \eta$.

We have now discussed all the ingredients of our scheme for accepting a

step and updating the trust region. The following algorithm summarizes our ideas.

ALGORITHM 5.2.1. Accepting a Step and Updating the Trust Region.

Let $0 < \sigma_1 < \sigma_2 < 1$ and $0 < \tau_1 < \tau_2 < 1 < \tau_3$ be specified constants.

1) if $g(x_c) + \nabla g(x_c)^T s_c = 0$ then

1.1) set $\lambda_+ = \lambda^{QP}$

1.2) determine N_c as in (5.2.8) or (5.2.9)

else

1.3) set $\lambda_+ = 0$

1.4) $N_c = \{\eta_c\}$

2) if $N_c = \emptyset$ then

2.1) choose $\Delta_c \in [\tau_1 || s_c ||, \tau_2 || s_c ||]$

2.2) compute a new s_c

3) Determine N_{dec} as defined in (5.2.1)

4) if $N_c \bigcap N_{dec} = \emptyset$ then

4.1) choose $\Delta_c \in [\tau_1 || s_c ||, \tau_2 || s_c ||]$

4.2) compute a new s_c.

5) Accept $x_+ = x_c + s_c$, choose $\eta_+ \in N_c \bigcap N_{dec}$.

6) Determine N_{inc} as defined in (5.2.11)

7) IF $N_c \bigcap N_{inc} \neq \emptyset$ then

7.1) choose $\eta_+ \in N_c \bigcap N_{inc}$

7.2) $\Delta_+ \in [\Delta_c, \tau_3 \Delta_c]$

7.3) exit.

8) Determine N_{sme} as defined in (5.2.12)

9) if $N_c \bigcap N_{sme} \neq \emptyset$ then

9.1) choose $\eta_+ \in N_c \bigcap N_{sme}$,

9.2) $\Delta_+ = \Delta_c$,

else

9.3) choose $\Delta_+ \in [\tau_1 || s_c ||, \tau_2 || s_c ||]$.

5.3. The SQPQC Algorithm

Before the statement of the algorithm, we will point out the guiding principals of our method. A full SQP step should be taken whenever $|| s^{QP} ||_2 \leq \Delta$ in order to maintain fast local convergence. When a full SQP step cannot be taken, the step should be determined by the solution of problem (QFQC). The augmented Lagrangian function should be used to determine the acceptability of the step, and also to update the trust region, as described in Section 5.2. The penalty constant to be used in the augmented Lagrangian should be determined from the model subproblem or from the optimistic scheme we developed in Section 5.2.

For the preliminary implementation of the algorithm, we use second order information. At each iteration k, problem (QP) is defined to be

$$\begin{aligned} \underset{s}{\textit{minimize}} \quad & \nabla f(x_k)^T s + \tfrac{1}{2} s^T \nabla^2_{xx} l(x_k,\lambda_k) s \\ \textit{subject to} \quad & g(x_k) + \nabla g(x_k)^T s = 0 \,, \end{aligned}$$

and problem (QFQC) is defined as

$$\begin{aligned} \underset{s}{\textit{minimize}} \quad & \nabla f(x_k)^T + \tfrac{1}{2} s^T \nabla^2 B_k s \\ \textit{subject to} \quad & \| s \|_2 \leq \Delta_k \\ & \| g(x_k) + \nabla g(x_k)^T s \|_2 \leq \Theta_k \,. \end{aligned}$$

We will use $B_k = \nabla^2_{xx} l(x_k,\lambda_k)$ if $\Theta_k = 0$, and $B_k = \nabla^2 f(x_k)$ otherwise. This will allow us to choose a step as close as possible to the SQP step if linearized constraints are satisfied.

The choice of multiplier used in the Hessian matrix $\nabla^2_{xx} l(x_k,\lambda_k)$ depends on the previous step taken. If s_{k-1} satisfies linearized constraints, we choose λ_k to be the multiplier associated with the SQP step, i.e., $\lambda_k = \lambda^{QP}$. As before, this choice is motivated by the fact that λ^{QP} is the only multiplier that allows linearized constraints to be satisfied. If the previous step s_{k-1} does not satisfy linearized constraints, we choose the multiplier to be a particular case of the multiplier update formula U_{GUF}, namely

$$\lambda_k \;=\; (\nabla g(x_k)^T \nabla g(x_k))^{-1} (g(x_k) - \nabla g(x_k)^T \nabla f(x_k)) .$$

In practice, this particular formula has proven to be very successful, see Miele [1972a], [1972b]. The estimate for the initial multiplier λ_0, is also chosen in this

way.

Finally, the initial radius of the trust region Δ_0 is chosen to be a percentage of the length of the Cauchy step for problem (4.1.2). This choice follows the strategy suggested by Powell [1970] for the unconstrained case.

We now give a general statement of our algorithm for solving equality constrained minimization problems.

ALGORITHM 5.3.1 The SQPQC Method.

1) Let $x_0 \in \mathbf{R}^n$, determine λ_0, Δ_0.

2) for $k = 0, 1, 2, \cdots$ until *convergence* do

2.1) Compute $\nabla f(x_k)$, $\nabla^2 f(x_k)$, $\nabla g(x_k)$, $\nabla^2_{xx} l(x_k,\lambda_k)$.

2.2) Compute the solution s^{QP} (and λ^{QP}) to problem (QP).

2.3) If $\| s^{QP} \| \leq \Delta_k$ then

$s_k = s^{QP}$,

go to 2.5).

2.4) Determine an approximate solution s_k to problem (QFQC) by Algorithm (5.1.1).

2.5) $x_+ = x_k + s_k$

2.6) Decide whether x_+ is acceptable, and calculate a new value of Δ_k by Algorithm (5.2.1)

2.7) If x_+ is not acceptable then go to 2.4);

otherwise $x_{k+1} = x_+$ and $\Delta_{k+1} = \Delta_k$.

5.4. Numerical Results

In order to study the effectiveness of our approach from arbitrary starting points, we produced a preliminary implementation of algorithm SQPQC. Although research is still being done to make this algorithm more efficient, we wanted to obtain a feel for the robustness of the approach. For this we compared our method SQPQC with two SQP approaches: VF02AD by Powell [1977a], which is available in the Harwell Subroutine Library, and NPSOL by Gill, Murray, Saunders and Wright [1983]. All tests were performed in double precision on a VAX 11/780.

We now list a subset of our test problems and the results obtained. Most of these problems are referenced and can be found in Hock and Schittkowski [1981]. The number in parenthesis denotes the number given to this problem in Hock and Schittkowski, n is the number of variables in the problem, and m is the number of equality constraints.

The results from this subset of test problems are reported in tables following the statement of each problem. The starting point x_0 precedes each table. The column labeled CONV indicates whether or not convergence was obtained, and the column labeled NO. ITER indicates the number of iterations the algorithm took to converge. This number does not give meaningful

comparisons for many reasons, including the fact that the algorithm is only in a preliminary stage. We have, however, included it for completeness. The column labeled $\mathbf{x_F}$ indicates to which solution the method converged. Comments are included in some cases to provide more information about the particular case.

At the end of this section, we summarize the convergence results in Table 1. The number in parentheses in the CONV column indicates the number of iterations the algorithm took to converge.

PROBLEM 1 (60) - $n = 3,\ m = 1$

$f(x) = (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^4$

$g_1(x) = x_1(1 + x_2^2) + x_3^4 - 4 - 3\sqrt{2}$

$x_{*,1} \approx (1.1048, 1.1966, 1.5352)$ $\qquad f(x_{*,1}) \approx 3.2568 \times 10^{-2}$

$\mathbf{x_0} = (2, 2, 2)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	7	$x_{*,1}$
NPSOL	Y	8	$x_{*,1}$
SQPQC	Y	12	$x_{*,1}$

$\mathbf{x_0} = (2.7, 2.9, 3.8)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	10	$x_{*,1}$
NPSOL	Y	13	$x_{*,1}$
SQPQC	Y	19	$x_{*,1}$

$\mathbf{x_0} = (27, 29, 38)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	36	$x_{*,1}$
NPSOL	Y	26	$x_{*,1}$
SQPQC	Y	27	$x_{*,1}$

$\mathbf{x_0} = (10, 10, 10)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	17	$x_{*,1}$
NPSOL	Y	21	$x_{*,1}$
SQPQC	Y	23	$x_{*,1}$

$\mathbf{x_0} = (11, 12, 15)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	19	$x_{*,1}$
NPSOL	Y	26	$x_{*,1}$
SQPQC	Y	24	$x_{*,1}$

$\mathbf{x_0} = (1, 2, 3)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	10	$x_{*,1}$
NPSOL	Y	13	$x_{*,1}$
SQPQC	Y	17	$x_{*,1}$

$\mathbf{x_0} = (1.5, 1.5, 1.5)$

METHOD	CONV	NO. ITER	$\mathbf{x_F}$
VF02AD	Y	6	$x_{*,1}$
NPSOL	Y	8	$x_{*,1}$
SQPQC	Y	15	$x_{*,1}$

PROBLEM 2 (77) - $n = 5$, $m = 2$

$$f(x) = (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + (x_4 - 1)^4 + (x_5 - 1)^6$$

$$g_1(x) = x_1^2 x_4 + \sin(x_4 - x_5) - 2\sqrt{2}$$
$$g_2(x) = x_2 + x_3^4 x_4^2 - 8 - \sqrt{2}$$

$x_{*,1} \approx$ (1.1661, 1.1821, 1.3802, 1.5060, 0.6109) $\quad f(x_{*,1}) \approx 2.4150 \times 10^{-1}$
$x_{*,2} \approx$ (-1.0287, -1.1017, 1.3545, 1.7603, 0.4531) $\quad f(x_{*,2}) \approx 4.6025$
$x_{*,3} \approx$ (1.0889, 1.1777, -1.2814, 1.7477, 0.8910) $\quad f(x_{*,3}) \approx 5.5333$
$x_{*,4} \approx$ (-0.9896, -0.9142, -1.3028, 1.8932, 0.4975) $\quad f(x_{*,4}) \approx 9.9087$

$x_0 = (2, 2, 2, 2, 2)$

METHOD	CONV	NO. ITER	x_F
VF02AD	Y	15	$x_{*,1}$
NPSOL	Y	16	$x_{*,1}$
SQPQC	Y	12	$x_{*,1}$

$x_0 = (1, 1, 1, 1, 1)$

METHOD	CONV	NO. ITER	x_F
VF02AD	Y	13	$x_{*,1}$
NPSOL	Y	13	$x_{*,1}$
SQPQC	Y	12	$x_{*,1}$

$x_0 = (10, 10, 10, 10, 10)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	N	*	*	line search fails at iter. 1
NPSOL	N	*	*	overflow in line search iter. 1
SQPQC	Y	21	$x_{*,1}$	

$x_0 = (-3, -3, -3, 9, 0)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	Y	21	$x_{*,2}$	$x_{*,2}$ is a saddle point
NPSOL	N	*	*	overflow in line search iter. 1
SQPQC	Y	21	$x_{*,2}$	$x_{*,2}$ is a saddle point

$x_0 = (-1, 8, 3, 3, 0)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	Y	18	$x_{*,2}$	$x_{*,2}$ is a saddle point
NPSOL	Y	17	$x_{*,2}$	
SQPQC	Y	17	$x_{*,2}$	

$x_0 = (4, 3, 7, -5, -3)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	N	*	*	inconsistent constraints at iter. 18
NPSOL	N	*	*	overflow in line search iter. 1
SQPQC	Y	21	$x_{*,2}$	$x_{*,2}$ is a saddle point

$x_0 = (-1, 3, -0.5, -2, -3)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	N	*	*	inconsistent constraints at iter. 12
NPSOL	Y	75	$x_{*,4}$	
SQPQC	Y	18	$x_{*,4}$	

$\mathbf{x}_0 = (12, 13, 14, 15, 7)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$	COMMENTS
VF02AD	Y	50	$x_{*,1}$	
NPSOL	N	*	*	overflow in line search at iter. 1
SQPQC	Y	22	$x_{*,1}$	

$\mathbf{x}_0 = (-2, -2, -2, -2, -2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$	COMMENTS
VF02AD	Y	39	$x_{*,3}$	
NPSOL	N	*	*	overflow in update routine at iter. 2
SQPQC	N	*	*	failure due to safeguarding step

PROBLEM 3 (79) - $n = 5$, $m = 3$

$$f(x) = (x_1 - 1)^2 + (x_1 - x_2)^2 + (x_2 - x_3)^2 + (x_4 - 1)^4 + (x_5 - 1)^4$$

$$g_1(x) = x_2 + x_2^2 + x_3^3 - 2 - 3\sqrt{2}$$
$$g_2(x) = x_2 + x_3^2 + x_4 - 2 - 2\sqrt{2}$$
$$g_3(x) = x_1 x_5 - 2$$

$x_{*,1} \approx (1.1911,\ 1.3626,\ 1.4728,\ 1.6350,\ 1.6790)$ $\quad f(x_{*,1}) \approx 7.8776 \times 10^{-2}$
$x_{*,2} \approx (-0.7661,\ 2.6667,\ -0.4681,\ -1.6191,\ -2.6103)$ $\quad f(x_{*,2}) \approx 27.4520$
$x_{*,3} \approx (-2.7022,\ -2.9899,\ 0.1719,\ 3.8479,\ -0.7401)$ $\quad f(x*3) \approx 649.5049$

$\mathbf{x}_0 = (2, 2, 2, 2, 2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	9	$x_{*,1}$
NPSOL	Y	12	$x_{*,1}$
SQPQC	Y	4	$x_{*,1}$

$\mathbf{x}_0 = (1, 1, 1, 1, 1)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	8	$x_{*,1}$
NPSOL	Y	10	$x_{*,1}$
SQPQC	Y	4	$x_{*,1}$

$\mathbf{x}_0 = (10, 10, 10, 10, 10)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	18	$x_{*,1}$
NPSOL	Y	22	$x_{*,1}$
SQPQC	Y	14	$x_{*,1}$

$\mathbf{x}_0 = (-2, -2, -2, -2, -2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$	COMMENTS
VF02AD	Y	16	$x_{*,2}$	each method converged to
NPSOL	Y	20	$x_{*,3}$	a different solution
SQPQC	Y	11	$x_{*,1}$	

$\mathbf{x}_0 = (-1, 3, -0.5, -2, -3)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	10	$x_{*,1}$
NPSOL	Y	8	$x_{*,1}$
SQPQC	Y	4	$x_{*,1}$

$\mathbf{x}_0 = (-1, 2, 1, -2, -2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$	COMMENTS
VF02AD	Y	16	$x_{*,2}$	
NPSOL	Y	21	$x_{*,1}$	different solution than the rest
SQPQC	Y	5	$x_{*,2}$	

PROBLEM 4 (79) - $n = 5,\ m = 3$

$$f(x) = x_1 x_2 x_3 x_4 x_5$$

$$g_1(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10$$
$$g_2(x) = x_2 x_3 - 5 x_4 x_5$$
$$g_3(x) = x_1^3 + x_3^2 + 1$$

$x_{*,1} \approx (-1.7171, 1.5957, 1.8272, -0.7636, -0.7636)$ $\quad f(x_{*,1}) \approx -2.9197$

$x_{*,2} \approx (-0.6990, -0.8699, -2.7899, -0.6967, -0.6967)$ $\quad f(x_{*,2}) \approx -8.2359 \times 10^{-1}$

$\mathbf{x}_0 = (-1, 1.5, 2, -1, -2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	10	$x_{*,1}$
NPSOL	Y	9	$x_{*,1}$
SQPQC	Y	5	$x_{*,1}$

$\mathbf{x}_0 = (-10, 10, 10, -10, -10)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$	COMMENTS
VF02AD	Y	21	$x_{*,1}$	
NPSOL	Y	86	$x_{*,2}$	different solution than others
SQPQC	Y	9	$x_{*,1}$	

$\mathbf{x}_0 = (-1, 2, 1, -2, -2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	9	$x_{*,1}$
NPSOL	Y	12	$x_{*,1}$
SQPQC	Y	6	$x_{*,1}$

$\mathbf{x}_0 = (-1, -1, -1, -1, -1)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	9	$x_{*,2}$
NPSOL	Y	11	$x_{*,2}$
SQPQC	Y	5	$x_{*,2}$

$\mathbf{x}_0 = (-2, 2, 2, 2, 2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	7	$x_{*,1}$
NPSOL	Y	17	$x_{*,1}$
SQPQC	Y	5	$x_{*,1}$

PROBLEM 7 (Boggs and Tolle [1981]) - $n = 2,\ m = 2$

$f(x) = -x_1$

$g_1(x) = x_2 - x_1^3$
$g_2(x) = x_1^2 - x_2$

$x_{*,1} = (1.0, 1.0) \qquad f(x_{*,1}) = -1.0$

$\mathbf{x}_0 = (2, 2)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	7	$x_{*,1}$
NPSOL	Y	8	$x_{*,1}$
SQPQC	Y	8	$x_{*,1}$

$\mathbf{x}_0 = (20, 20)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	13	$x_{*,1}$
NPSOL	Y	15	$x_{*,1}$
SQPQC	Y	13	$x_{*,1}$

$\mathbf{x}_0 = (50, 50)$

METHOD	CONV	NO. ITER	$\mathbf{x}_F$
VF02AD	Y	16	$x_{*,1}$
NPSOL	Y	18	$x_{*,1}$
SQPQC	Y	14	$x_{*,1}$

PROBLEM 8 (Vardi [1980]) - $n = 3,\ m = 1$

$f(x) = (x_1 - x_2)^2 + (x_2 - x_3)^4$

$g_1(x) = x_1 + x_1x_2^2 + x_3^4 - 3$

$x_{*,1} = (1.0, 1.0, 1.0) \qquad f(x_{*,1}) = 0.0$

$\mathbf{x_0} = (2.4, 0.5, 0.0)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	Y	16	$x_{*,1}$	Low accuracy obtained
NPSOL	Y	50	$x_{*,1}$	because the Hessian is
SQPQC	Y	50	$x_{*,1}$	singular at x_*

$\mathbf{x_0} = (10, -10, 10)$

METHOD	CONV	NO. ITER	x_F	COMMENTS
VF02AD	Y	26	$x_{*,1}$	Low accuracy obtained
NPSOL	Y	50	$x_{*,1}$	
SQPQC	Y	50	$x_{*,1}$	

The Hessian of the Lagrangian $\nabla^2_{xx} l(x_*, \lambda_*)$ is singular. VF02AD stops because it requires lower accuracy than the other two subroutines. NPSOL and SQPQC stop because the maximum number of iterations permitted is 50.

PROBLEM	STARTING POINT	CONV (NO. ITER) VF02AD	NPSOL	SQPQC
1	(2, 2, 2)	Y (7)	Y (8)	Y (12)
1	(2.7, 2.9, 3.8)	Y (10)	Y (13)	Y (19)
1	(27, 29, 38)	Y (36)	Y (27)	Y (27)
1	(10, 10, 10)	Y (17)	Y (21)	Y (23)
1	(11, 12, 15)	Y (19)	Y (26)	Y (24)
1	(1, 2, 3)	Y (10)	Y (13)	Y (17)
1	(1.5, 1.5, 1.5)	Y (6)	Y (8)	Y (15)
2	(2, 2, 2, 2, 2)	Y (15)	Y (16)	Y (12)
2	(1, 1, 1, 1, 1)	Y (13)	Y (13)	Y (12)
2	(10, 10, 10, 10, 10)	N (*)	N (*)	Y (21)
2	(-3, -3, 3, 9, 0)	Y (21)	N (*)	Y (21)
2	(-1, 8, 3, 3, 0)	Y (18)	Y (17)	Y (17)
2	(4, 3, 7, -5, -3)	N (*)	N (*)	Y (21)
2	(-1, 3, -0.5, -2, -3)	N (*)	Y (75)	Y (18)
2	(12, 13, 14, 15, 7)	Y (50)	N (*)	Y (21)
2	(-2, -2, -2, -2, -2)	Y (39)	N (*)	N (*)
3	(2, 2, 2, 2, 2)	Y (9)	Y (12)	Y (4)
3	(1, 1, 1, 1, 1)	Y (8)	Y (10)	Y (4)
3	(10, 10, 10, 10, 10)	Y (18)	Y (22)	Y (14)
3	(-2, -2, -2, -2, -2)	Y (18)	Y (20)	Y (11)
3	(-1, 3, -0.5, -2, -3)	Y (10)	Y (8)	Y (4)
3	(-1, 2, 1, -2, -2)	Y (16)	Y (21)	Y (5)
4	(-1, 1.5, 2, -1, -2)	Y (10)	Y (9)	Y (5)
4	(-10, 10, 10, -10, -10)	Y (21)	Y (86)	Y (9)
4	(-1, 2, 1, -2, -2)	Y (9)	Y (12)	Y (6)
4	(-1, -1, -1, -1, -1)	Y (9)	Y (11)	Y (5)
4	(-2, 2, 2, 2, 2)	Y (7)	Y (17)	Y (5)
7	(2, 2)	Y (7)	Y (8)	Y (8)
7	(20, 20)	Y (13)	Y (15)	Y (13)
7	(50, 50)	Y (16)	Y (18)	Y (14)
8	(2.4, 0.5, 0.0)	Y (16)	Y (50)	Y (50)
8	(20, 20)	Y (26)	Y (50)	Y (50)

TABLE 1

CHAPTER 6

Concluding Remarks

We have presented a framework for a trust region approach for solving equality constrained optimization problems. At each iteration the subproblem we solve is not in general the successive quadratic programming (SQP) subproblem. We have motivated the conjecture that asymptotically our step is the same as the step produced by solving the SQP subproblem.

The theoretical results presented in this thesis, although preliminary, have established important links between the step selection process and several widely used merit functions. We have shown that the step we obtain is a descent direction on either the standard penalty function or the augmented Lagrangian function, where each penalty constant is provided by the solution of the associated subproblem.

The augmented Lagrangian was suggested as the choice for the merit function. A strategy for choosing the penalty constant when it is not given by the solution to the subproblem was discussed. This choice is not arbitrary, it is the choice dictated by minimizing the augmented Lagrangian function. An updating scheme for the trust region was included.

We have produced a preliminary implementation of our approach. This implementation, although not polished, has produced good numerical results. These numerical results support the robustness of the algorithm, and have lead us to believe that our approach is worthy of continued research.

There are many questions left unanswered at the writing of this thesis. Considerable theoretical and numerical investigations must be performed before we make any conclusions about the global behavior of the algorithm.

The implementation of the algorithm must be refined. In particular, an efficient safeguarding routine for the optimal step is needed. This will require a closer look at problem (QFQC) and the characteristics of its solution.

We proposed a dogleg approach for approximating the solution of problem (QFQC). Two different dogleg curves were defined. Their definitions involve the Cauchy step for the 2-norm of the linearized constraints, the Cauchy step for the quadratic model of the objective function (not necessarily from the current point), as well as the SQP step. Numerical testing remains to be done for our dogleg approach.

For the SQP approach as well as for our SQPQC approach, an approximation to the Hessian of the Lagrangian function must be used for an efficient algorithm. Currently, this is the topic of much research, e.g. Tapia [1984], but the problem has not yet been solved. This a certainly an important missing piece in the development of our global trust region algorithm.

BIBLIOGRAPHY

BARTHOLOMEW-BIGGS, M.C. [1980]. Recursive quadratic programming based on penalty functions for constrained minimization. *Nonlinear Optimization, Theory and Algorithms,* L.C.W. Dixon, E. Spedicato, G.P. Szego, eds., Birkhauser.

BERTSEKAS, D.P. [1976] On penalty and multiplier methods for constrained optimization. *SIAM J. Control and Optimization,* 14, pp. 216-135.

BOGGS, P.T., TOLLE, J.W. [1980]. Augmented Lagrangians that are quadratic in the multiplier. *Journal of Optimization Theory and Applications,* 3, pp. 17-26.

BOGGS, P.T., TOLLE, J.W. [1981]. An implementation of a quasi-Newton method for constrained optimization. Technical report 81-3, Dept. of Mathematics and Curriculum in Operations Research and Systems Analysis, University of North Carolina, Chapel Hill, N.C.

BOGGS, P.T., TOLLE, J.W., WANG. P. [1982]. On the local convergence of quasi-Newton methods for constrained optimization. *SIAM J. Control and Optimization,* 20, pp. 161-171.

BUYS, J.D. [1972]. *Dual algorithms for constrained optimization problems.* Ph.D. thesis, University of Leiden, Netherlands.

COURANT, R. [1943]. Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc.,* 49, pp. 1-23.

DENNIS, J.E., JR., SCHNABEL, R.B. [1983]. *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice-Hall, Englewood Cliffs, N.J.

DIPILLO, G., GRIPPO, L. [1979]. A new class of augmented Lagrangians in nonlinear programming. *SIAM J. Control and Optimization,* 17, pp. 618-628.

FIACCO, A.V., MCCORMICK, G.P. [1968]. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* John Wiley & Sons, New York.

FLETCHER, R. [1970]. A class of methods for nonlinear programming with termination and convergence properties. *Integer and Nonlinear Programming,* J. Abadie, ed., North Holland, Amsterdam.

FLETCHER, R. [1980]. *Practical Methods of Optimization.* Vol. 1, John Wiley, New York.

FLETCHER, R. [1981]. *Practical Methods of Optimization.* Vol. 2, John Wiley, New York.

FLETCHER, R. [1981]. Methods for nonlinear constraints. *Nonlinear Optimization 1981,* M.J.D. Powell, ed., Academic Press, New York, NY, pp. 185-212.

GARCIA-PALOMARES, U.M., MANGASARIAN, O.L. [1976]. Superlinearly convergent quasi-Newton algorithms for nonlinearly constrained optimization problems. *Math. Prog.,* 11, pp. 1-13.

GAY, D.M. [1981]. Computing optimal locally constrained steps. *SIAM J. Scientific and Statistical Computing,* 2, pp. 186-197.

GAY, D.M. [1983]. A trust region approach to linearly constrained optimization. *Proceedings of the Dundee Biennial Conference on Numerical Analysis.*

GILL, P.E., MURRAY, W., WRIGHT, M.H. [1981]. *Practical Optimization.* Academic Press, London, New York, Toronto, Sydney and San Francisco.

GILL, P.E., MURRAY, W., SAUNDERS, M.A., WRIGHT, M.H. [1983]. User's guide for SOL/NPSOL: a Fortran package for nonlinear programming. Department of Operations Research, Stanford University, California.

GLAD, S.T. [1979]. Properties of updating methods for the multipliers in augmented Lagrangians. *Journal of Optimization Theory and Applications,* 28, pp 135-156.

GOLDFELD, S.M., QUANDT, R.E., TROTTER, H.F. [1966]. Maximization by quadratic hill-climbing, *Econometrica,* 34, pp. 541-551.

HAARHOFF, P.C., BUYS, J.D. [1970]. A new method for the optimization of a nonlinear function subject to nonlinear constraints. *Computer Journal,* 13, pp. 178-184.

HAN, S.-P. [1976]. Superlinearly convergent variable metric algorithms for general nonlinear programming problems. *Mathematical Programming,* 11, pp 263-283.

HAN, S.-P. [1977]. Dual variable metric algorithms for constrained optimization. *SIAM J. Control and Optimization,* 15, pp. 546-565.

HEBDEN, M.D. [1973]. An algorithm for minimization using exact second derivatives. Technical report TP515, Atomic Energy Research Establishment, Harwell, England.

HESTENES, M.R., [1969]. Multiplier and gradient methods, *J. Optimization Theory and Applications,* 4, pp. 303-320.

HOCK, W., SCHITTKOWSKI, K. [1981]. *Test Examples for Nonlinear Programming Codes.* Lecture Notes in Economics and Mathematical Systems 187, Springer-Verlag, Berlin, Heidelberg and New York.

LEVENBERG, K. [1944]. A method for the solution of certain problems in least squares. *Quart. Appl. Math.,* 2, pp. 164-168.

MARATOS, N. [1978]. *Exact penalty function algorithms for finite dimensional and control problems.* Thesis, University of London.

MARQUARDT, D. [1963]. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Applied Mathematics,* 11, pp. 431-441.

MIELE, A., CRAGG, E.E., IYER, R.R., LEVY, A.V. [1971a]. Use of the augmented penalty function in mathematical programming, part I. *J. Optimization Theory and Applications,* 8, pp. 115-130.

MIELE, A., CRAGG, E.E., LEVY, A.V. [1971b]. Use of the augmented penalty function in mathematical programming, part II. *J. Optimization Theory and Applications,* 8, pp. 131-153.

MIELE, A., MOSELEY, P.E., CRAGG, E.E. [1972a]. A modification of the method of multipliers for mathematical programming problems. *Techniques of Optimization,* A.V. Balakrishnan, ed., Academic Press, New York, NY.

MIELE, A., MOESELEY, P.E., LEVY, A.V., COGGINS, G.M. [1972b]. On the method of multipliers for mathematical programming problems. *J. Optimization Theory and Applications,* 10, pp. 1-33.

MORÉ, J.J. [1977]. The Levenberg-Marquardt algorithm: implementation and theory. *Numerical Analysis,* G.A. Watson, ed., pp. 105-116, Lecture Notes in Mathematics 630, Springer-Verlag, Berlin, Heidelberg and New York.

MORÉ, J.J., SORENSEN, D.C. [1983]. Computing a trust region step. *SIAM J. Scientific and Statistical Computing,* 4, pp. 553-572.

ORTEGA, J.M., RHEINBLODT, W.C. [1970]. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, London and New York.

PIETRZKOWSKI, T. [1969]. An exact potential method for constrained maxima. *SIAM J. Numerical Analysis,* 6, pp. 299-304.

POLYAK, B.T. [1971]. The convergence rate of the penalty function method. *Zh. vychisl. Mat. mat. fiz.,* 11, pp. 1-12.

POWELL, M.J.D., [1969]. A method for nonlinear constraints in minimization problems. *Optimization,* R. Fletcher, ed., Academic Press, New York, pp. 283-298.

POWELL, M.J.D. [1970]. A new algorithm for unconstrained optimization. *Nonlinear Programming,* J.B. Rosen, O.L. Mangasarian, K. Ritter, eds., Academic Press, New York, pp. 31-65.

POWELL, M.J.D. [1975]. Convergence properties of a class of minimization algorithms. *Nonlinear Programming 2,* O.L. Mangasarian, R.R. Meyer, S.M. Robinson, eds. Academic Press, New York, NY., pp. 1-27.

POWELL, M.J.D. [1977a]. A fast algorithm for nonlinearly constrained optimization calculations. *Numerical Analysis, Dundee 1977, Lecture notes in mathematics* G.A. Watson, ed., Springer Verlag, Berlin, 1978, pp. 144-157.

POWELL, M.J.D. [1977b]. Constrained optimization by a variable metric method. DAMTP report 77/NA6, Cambridge University.

POWELL, M.J.D. [1978]. The convergence of variable metric methods for nonlinearly constrained optimization problems. *Nonlinear Programming 3,* O. Mangasarian, R. Meyer, S. Robinson, eds., Academic Press, New York, NY., pp. 27-63.

SCHITTKOWSKI, K. [1981]. The nonlinear programming method of Wilson, Han and Powell with an augmented Lagrangian type line search function, Part 2: An efficient implementation with linear least squares subproblems. *Numerische Mathematik*, 38, pp. 115-127

SCHITTKOWSKI, K. [1982]. On the convergence of a sequential quadratic programming method with an augmented Lagrangian line search function. Technical report SOL 82-4, Department of Operations Research, Stanford University, California.

SCHULTZ, G.A., SCHNABEL, R.B., BYRD, R.H. [1985]. A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties. *SIAM J. Numerical Analysis*, 22, pp. 47-67.

SORENSEN, D.C. [1982a]. Newton's method with a model trust region modification. *SIAM J. Numerical Analysis*, 19, pp. 409-426.

SORENSEN, D.C. [1982b]. Private communication.

TAPIA, R.A. [1974a]. Newton's method for problems with equality constraints. *SIAM J. Numerical Analysis*, 11, pp. 174-196.

TAPIA, R.A. [1974b]. Newton's method for optimization problems with equality constraints. *SIAM J. Numerical Analysis*, 11, pp. 874-886.

TAPIA, R.A. [1977]. Diagonalized multiplier methods and quasi-Newton methods for constrained optimization. *Journal of Optimization Theory and Applications*, 22, pp. 135-194.

TAPIA, R.A. [1978]. Quasi-Newton methods for equality constrained optimization: equivalence of existing methods and a new implementation. *Nonlinear Programming 3*, O. Mangasarian, R. Meyer, S. Robinson, eds., Academic Press, New York, NY., pp. 125-164.

TAPIA, R.A. [1984]. On secant updates for use in general constrained optimization. Technical report 84-3, Department of Mathematical Sciences, Rice University, Houston, Tx.

THOMAS, S.W. [1975]. Sequential estimation techniques for quasi-Newton algorithms. Technical report TR75-227, Department of Computer Science, Cornell University.

VARDI, A. [1980]. *Trust region strategies for unconstrained and constrained minimization.* Ph.D. thesis, School of Operations Research and Industrial Engineering, Cornell University.

WILSON, R.B. [1963]. *A simplicial algorithm for concave programming.* Ph.D. thesis, Harvard University.

ZANGWILL, W.I. [1967]. Nonlinear programming via penalty functions, *Management Science*, vol. 13, pp. 344-358.